SemEval 2022

The 16th International Workshop on Semantic Evaluation (SemEval-2022)

Proceedings of the Workshop

July 14-15, 2022

Order copies of this and other ACL proceedings from:

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

ISBN 978-1-955917-80-3

Introduction

Welcome to SemEval-2022!

The Semantic Evaluation (SemEval) series of workshops focuses on the evaluation and comparison of systems that can analyze diverse semantic phenomena in text, with the aims of extending the current state of the art in semantic analysis and creating high quality annotated datasets in a range of increasingly challenging problems in natural language semantics. SemEval provides an exciting forum for researchers to propose challenging research problems in semantics and to build systems/techniques to address such research problems.

SemEval-2022 is the sixteenth workshop in the series of International Workshops on Semantic Evaluation. The first three workshops, SensEval-1 (1998), SensEval-2 (2001), and SensEval-3 (2004), focused on word sense disambiguation, each time expanding in the number of languages offered, the number of tasks, and also the number of teams participating. In 2007, the workshop was renamed to SemEval, and the subsequent SemEval workshops evolved to include semantic analysis tasks beyond word sense disambiguation. In 2012, SemEval became a yearly event. It currently takes place every year, on a two-year cycle. The tasks for SemEval-2022 were proposed in 2021, and next year's tasks have already been selected and are underway.

SemEval-2022 is co-located (hybrid) with The 2022 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2022) on July 14 - 15. This year's SemEval included the following 12 tasks:

- Lexical semantics
  - Task 1: CODWOE - COmparing Dictionaries and WOrd Embeddings
  - Task 2: Multilingual Idiomaticity Detection and Sentence Embedding
  - Task 3: Presupposed Taxonomies - Evaluating Neural-network Semantics (PreTENS)
- Social factors & attitudes
  - Task 4: Patronizing and Condescending Language Detection
  - Task 5: MAMI - Multimedia Automatic Misogyny Identification
  - Task 6: iSarcasmEval - Intended Sarcasm Detection in English and Arabic
- Discourse, documents, and multimodality
  - Task 7: Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts
  - Task 8: Multilingual news article similarity
  - Task 9: R2VQ - Competence-based Multimodal Question Answering
- Information extraction
  - Task 10: Structured Sentiment Analysis
  - Task 11: MultiCoNER - Multilingual Complex Named Entity Recognition
  - Task 12: Symlink - Linking Mathematical Symbols to their Descriptions

This volume contains both task description papers that describe each of the above tasks and system description papers that present the systems that participated in the tasks. A total of 12 task description papers and 221 system description papers are included in this volume.

SemEval-2022 features two awards, one for organizers of a task and one for a team participating in a task. The Best Task award recognizes a task that stands out for making an important intellectual contribution to empirical computational semantics, as demonstrated by a creative, interesting, and scientifically rigorous dataset and evaluation design, and a well-written task overview paper. The Best Paper award recognizes a system description paper (written by a team participating in one of the tasks) that advances our understanding of a problem and available solutions with respect to a task. It need not be the highest-scoring system in the task, but it must have a strong analysis component in the evaluation, as well as a clear and reproducible description of the problem, algorithms, and methodology.

2022 has been another particularly challenging year across the globe. We are immensely grateful to the task organizers for their perseverance through many ups, downs, and uncertainties, as well as to the large number of participants whose enthusiastic participation has made SemEval once again a successful event! Thanks also to the task organizers who served as area chairs for their tasks, and to both task organizers and participants who reviewed paper submissions. These proceedings have greatly benefited from their detailed and thoughtful feedback. Thousands of thanks to our assistant organizers Siddharth Singh and Shyam Ratan for their extensive, detailed, and dedicated work on the production of these proceedings! We also thank the members of the program committee who reviewed the submitted task proposals and helped us to select this exciting set of tasks, and we thank the NAACL 2022 conference organizers for their support. Finally, we most gratefully acknowledge the support of our sponsor: the ACL Special Interest Group on the Lexicon (SIGLEX).

The SemEval-2022 organizers: Guy Emerson, Natalie Schluter, Gabriel Stanovsky, Ritesh Kumar, Alexis Palmer, and Nathan Schneider

Organizing Committee

SemEval Organizers:

Guy Emerson, Cambridge University
Natalie Schluter, IT University Copenhagen
Gabriel Stanovsky, The Hebrew University of Jerusalem
Ritesh Kumar, Dr. Bhimrao Ambedkar University
Alexis Palmer, University of Colorado Boulder
Nathan Schneider, Georgetown University

Assistant Organizers:

Siddharth Singh, Dr. Bhimrao Ambedkar University
Shyam Ratan, Dr. Bhimrao Ambedkar University

Task Organizers:

Task 1: Timothee Mickus, Denis Paperno, Mathieu Constant, Kees van Deemter
Task 2: Harish Tayyar Madabushi, Marcos Garcia, Carolina Scarton, Marco Idiart, Aline Villavicencio
Task 3: Dominique Brunato, Cristiano Chesi, Shammur Absar Chowdhury, Felice Dell'Orletta, Simonetta Montemagni, Giulia Venturi, Roberto Zamparelli
Task 4: Carla Perez-Almendros, Luis Espinosa-Anke, Steven Schockaert
Task 5: Elisabetta Fersini, Paolo Rosso, Francesca Gasparini, Alyssa Lees, Jeffrey Sorensen
Task 6: Ibrahim Abu Farha, Silviu Oprea, Steve Wilson, Walid Magdy
Task 7: Michael Roth, Talita Kloppenburg-Anthonio, Anna Sauer
Task 8: Xi Chen, Ali Zeynali, Chico Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, Mattia Samory
Task 9: James Pustejovsky, Jingxuan Tu, Marco Maru, Simone Conia, Roberto Navigli, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Eben Holderness
Task 10: Jeremy Barnes, Andrey Kutuzov, Jan Buchmann, Laura Ana Maria Oberländer, Enrica Troiano, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, Stephan Oepen
Task 11: Shervin Malmasi, Besnik Fetahu, Anjie Fang, Sudipta Kar, Oleg Rokhlenko
Task 12: Viet Dac Lai, Amir Ben Veyseh, Thien Huu Nguyen, Franck Dernoncourt

Invited Speakers:

Allyson Ettinger, University of Chicago (shared speaker with *SEM)
Alane Suhr, Cornell University

Program Committee

Task Selection Committee:

Carlos Santos Armendariz
Pepa Atanasova
Alberto Barrón-Cedeño
Steven Bethard
Luis Chiruzzo
Giovanni Da San Martino
Ron Daniel Jr.
Marina Danilevsky
Amitava Das
Leon Derczynski
Franck Dernoncourt
Jennifer D'Souza
Richard Evans
Hamed Firooz
Tommaso Fornaciari
Michael Gamon
Goran Glavaš
Paul Groth
Corey Harper
Nabil Hossain
Najla Kalach
Georgi Karadzhov
Henry Kautz
Seokhwan Kim
Anna Korhonen
Egoitz Laparra
Shuailong Liang
Zhenhua Ling
Walid Magdy
Diwakar Mahajan
Federico Martelli
Jonathan May
J. A. Meaney
Timothy Miller
Preslav Nakov
Roberto Navigli
Gustavo Henrique Paetzold
John Pavlopoulos
Mohammad Taher Pilehvar
Massimo Poesio
Senja Pollak
Simone Paolo Ponzetto
Soujanya Poria
Matthew Purver
Marko Robnik-Šikonja
Sara Rosenthal

Antony Scerri
Dominik Schlechtweg
Matthew Shardlow
Fabrizio Silvestri
Edwin Simpson
Thamar Solorio
Jeffrey Sorensen
Sasha Spala
Alexandra Uma
Ivan Vulić
Cunxiang Wang
Steven Wilson
Marcos Zampieri
Yue Zhang
Boyuan Zheng
Xiaodan Zhu

Table of Contents

Program

Thursday, July 14, 2022

08:30 - 10:00	*Invited Talk I: Alane Suhr*
10:00 - 10:30	*Coffee Break*
10:30 - 12:00	*Oral Session I: Task Description Papers*
12:00 - 13:30	*Lunch Break*
13:30 - 15:00	*Poster Session I: System Description Papers*
15:00 - 15:30	*Coffee Break*
15:30 - 17:00	*Poster Session II: System Description Papers*

Friday, July 15, 2022

08:30 - 10:00	*Oral Session II: Task Description Papers*
10:00 - 10:30	*Coffee Break*
10:30 - 12:00	*Poster Session III: System Description Papers*
12:00 - 13:30	*Lunch Break*
13:30 - 15:00	*Invited Talk II: Allyson Ettinger Understanding" and prediction: Controlled examinations of meaning sensitivity in pre-trained models [Shared with *SEM]*
15:00 - 15:30	*Coffee Break*
15:30 - 16:45	*Poster Session IV: System Description Papers*
16:45 - 17:30	*Oral Session III: Best Paper Awards*

Semeval-2022 Task 1: CODWOE – Comparing Dictionaries and Word Embeddings

Timothee Mickus[*]
Helsinki University
timothee.mickus@helsinki.fi

Kees van Deemter
Utrecht University
c.j.vandeemter@uu.nl

Mathieu Constant
Université de Lorraine
CNRS, ATILF
mconstant@atilf.fr

Denis Paperno
Utrecht University
d.paperno@uu.nl

Abstract

Word embeddings have advanced the state of the art in NLP across numerous tasks. Understanding the contents of dense neural representations is of utmost interest to the computational semantics community. We propose to focus on relating these opaque word vectors with human-readable definitions, as found in dictionaries. This problem naturally divides into two subtasks: converting definitions into embeddings, and converting embeddings into definitions. This task was conducted in a multilingual setting, using comparable sets of embeddings trained homogeneously.

1 Introduction

Word embeddings are a success story in NLP. They have been equated to distributional semantics models (Lenci, 2018; Boleda, 2020), a theory of semantics which relates the meaning of words to their distribution in context (Harris, 1954). Recently introduced contextualized word embeddings (e.g. Devlin et al., 2019) have set a new state of the art on a wide variety of tasks. For this reason, they have attracted much research interest. Do they depict consistent semantic spaces and are they theoretically valid (Mickus et al., 2020b; Yenicelik et al., 2020)? What limitations are to be expected in these models (Bender and Koller, 2020)? Can they scale up in performance (Brown et al., 2020)?

Word embeddings are dense vector representations of meaning which are not easily intelligible to a human observer. Many techniques have been employed to make embedding spaces more interpretable. A promising approach consists in *converting these opaque vectors into human readable definitions, as one could find in a dictionary*: accurately translating a dense, opaque vector representation into an equivalent human-readable piece of text would allow us to peer into the black box

[*]Work conducted while at ATILF

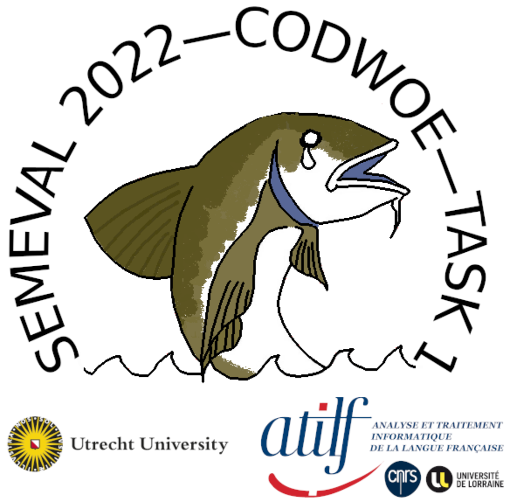

SEMEVAL 2022—CODWOE—TASK 1
Utrecht University
atilf
ANALYSE ET TRAITEMENT INFORMATIQUE DE LA LANGUE FRANÇAISE
cnrs
UNIVERSITÉ DE LORRAINE

Figure 1: Logo for CODWOE shared task

of modern neural network architectures. This avenue of research, known as definition modeling, was pioneered by Noraset et al. (2017). One may however question whether the task is at all feasible: there is no guarantee that the information content of a dictionary definition is similar to that which is described by real-valued vectors inferred from word distributions.

The SemEval Shared Task on Comparing Dictionaries and Word Embeddings (CODWOE) sets out to study whether embeddings and dictionaries encode similar information. We present the task and relevant state of the art in Section 2. We describe the data collected and presented to participants in Section 3. In Section 4, we discuss the metrics used to rank participant submissions. Our baseline model is presented in Section 5. We list results from participants' submissions in Section 6 and provide a more in-depth discussion in Section 7.

2 What we are fishing for

What is in a word embedding? Are word embeddings semantic descriptions, in the same sense that dictionary definitions are? If so, embeddings and definitions must be translatable into one another. The CODWOE shared task was set up to test this. The shared task participants investigated whether a word vector—e.g. $\vec{\texttt{cod}}$—contains the same information as the corresponding dictionary definition—viz. "*any of various bottom-dwelling fishes (family Gadidae, the cod family) that usually occur in cold marine waters and often have barbels and three dorsal fins*."[1]

We decompose this research problem into two tracks: the first corresponds to the vector-to-sequence task of Definition Modeling, the second to the sequence-to-vector Reverse Dictionary task. The task of definition modeling consists in using the vector representation of $\vec{\texttt{cod}}$ to produce the associated gloss, "*any of various bottom-dwelling fishes (family Gadidae, the cod family) that usually occur in cold marine waters and often have barbels and three dorsal fins*". The reverse dictionary task is the mathematical inverse: reconstruct an embedding $\vec{\texttt{cod}}$ from the corresponding gloss.

These two tracks display a number of interesting characteristics. These tasks are obviously useful for explainable AI, since they involve converting human-readable data into machine-readable data and back. They also have a theoretical significance: both glosses and word embeddings are representations of meaning, and therefore involve the conversion of distinct non-formal semantic representations. From a practical point of view, the ability to infer word-embeddings from dictionary resources, or dictionaries from large un-annotated corpora, would prove a boon for many under-resourced languages.

2.1 Track 1: Definition Modeling

The first track consists in an application of Definition Modeling. As training material, participants have access to a set of data points, each of which consists of a source word embedding and a corresponding target word definition (see Figure 2). Participants are tasked with generating new definitions for an unseen test set of embeddings.

Definition Modeling is a recent addition in NLG tasks (Noraset et al., 2017) which seeks to do just that. It has since then gained traction (Gadetsky et al., 2018; Mickus et al., 2019; Li et al., 2020; Zhang et al., 2020a, a.o.). Other languages than English have also been studied, including Chinese (Yang et al., 2019), French (Mickus et al., 2020a), Wolastoqey (Bear and Cook, 2021), and more (Kabiri and Cook, 2020). At its very inception, Definition Modeling was suggested as a means of evaluating the content of distributional semantic models (Noraset et al., 2017). In practice however, different researchers rarely use comparable sets of embeddings (Mickus et al., 2020a), effectively making proper comparisons across systems impossible as they use distinct inputs. To fill this gap, we created a dataset of comparable embeddings from different languages and neural architectures, trained as homogeneously as possible on comparable data; see 3.2 below.

2.2 Track 2: Reverse Dictionary

Reverse dictionaries (a.k.a. retrograde dictionaries) are lexical resources that flip the usual structure of dictionaries, allowing users to query words based on the definitions they would expect them to have. One of the major challenges of such resources consists in providing definition glosses that match with users' expectations. As a consequence, a trend of research in NLP has focused on producing dynamic reverse dictionaries, that would interpret input definitions and map them back to the corresponding word. We refer the reader to the comprehensive review of Siddique and Sufyan Beg (2019), and provide here mainly highlights.

An early strand of research focused on augmenting definitions using synonyms or other semantically related words, such as hypernyms or hyponyms. This approach has been applied to multiple languages, from Turkish to English and to Japanese (Shaw et al., 2013; Bila et al., 2004; El Khalout and Oflazer, 2004). Building on this query-augmentation approach, we find works focused on integrating richer lexical resources, such as WordNet, the Oxford dictionary, The Integral Dictionary, or LDA vector spaces (Dutoit and Nugues, 2002; Thorat and Choudhari, 2016; Méndez et al., 2013; Calvo et al., 2016).

A related trend of research is that of Zanzotto et al. (2010) and Hill et al. (2016), who use dictionaries as benchmarks for compositional semantics. Zanzotto et al. (2010) used a shallow neural network to implement a compositional distributional semantics model and use dictionaries as their training data. Hill et al. (2016) instead employ a LSTM to parse the full definition gloss and use the hidden state at the last time-step to predict the word be-

[1]From Merriam-Webster.

ing defined. In both cases, replacing the definition gloss with a user's query would lead to a reverse dictionary system. Since then, a number of works have attempted to implement reverse dictionaries using neural language models. The WantWords system (Zhang et al., 2020b; Qi et al., 2020) is based on a BiLSTM architecture, and incorporates auxiliary tasks such as part-of-speech prediction to boost the performance. Yan et al. (2020) seeks to replace the learned neural language models in Hill et al. (2016) or WantWords with a pre-trained model such as BERT (Devlin et al., 2019) and its multilingual variants, which allows them to use their system in a cross-lingual setting—querying in a language to obtain an answer in another. Most recently, Malekzadeh et al. (2021) used a neural language model based approach to implement a Persian reverse dictionary.

With respect to the CODWOE shared task, our interest lies in reconstructing the word embedding of the word being defined, rather than finding the corresponding word—an approach more closely related to that of Zanzotto et al. (2010) and Hill et al. (2016). Under this slight reformulation, the sequence-to-vector Reverse Dictionary task is strictly the inverse of the vector-to-sequence task of Definition Modeling. Hence we define the Reverse Dictionary task as *computing the components of a target word vector using as input a human-readable definition*. To solve this task, participants have access to a set of data points, each of which consists of a source word definition and a corresponding target word embedding, as training materials.

3 What's in the nets: Data used

The definition modeling and reverse dictionary tasks both require a parallel dataset, where dictionary definitions are aligned with corresponding word embeddings. The task is held in a multilingual setting. We provide data in English, French, Russian, Italian and Spanish. We selected these languages to facilitate the collection of comparable data: all these languages possess comparable large scale resources, including online dictionaries as well as corpora that can be used to train comparable embeddings. Our datasets are made available online at https://codwoe.atilf.fr/.

The aim of both tracks of CODWOE is to compare the semantic contents of definitions and embeddings. As a consequence, we ask participants to refrain from using external data such as pretrained models and lexical resources: including such external data would introduce another source of semantic information, and obfuscate the results from this shared task.

	with examples	without
en	0	806297
es	0	132583
fr	431793	573313
it	16127	86959
ru	122282	485208

Table 1: DBnary: number of items per language

	N. Sents.	**N. Tokens**	**N. Bytes**
it	78761031	955474050	5001829910
es	78973969	975762257	5001999992
fr	82082118	1004767254	5001999368
en	97622760	1035154295	5001999755
ru	79526583	1035661601	10036395727

Table 2: Embeddings: corpus statistics

3.1 Dictionary data

As a source of dictionary definitions, we primarily use the DBnary dataset (Sérasset, 2012),[2] an RDF-formatted version of some of the existing Wiktionary projects.[3] DBnary includes data for all of our selected languages. One sub-dataset per language is constructed. Definitions are selected according to corpus frequency and part-of-speech of the word being defined. We solely select nouns, adjectives, verbs and adverbs.

Table 1 presents the number of usable items in DBnary. Not all languages contain examples of usage. A brief regular expression lookup suggests that around 20K examples of usage can be found in the Spanish version of Wiktionary, while English yields at least 200K. We therefore discard the English version of DBnary and replace it by a manual parse, from which we also retrieve examples of usage.

3.2 Embeddings data

We have collected similar amounts of data for each language (Table 2) to use as training corpora. The sources we use to constitute these corpora are selected to be generally comparable: each cor-

[2]http://kaiko.getalp.org/about-dbnary/
[3]See https://www.wiktionary.org/

pus contains 2.5G data parsed and cleaned from Wikipedia,[4] 2.2G from the OpenSubtitles OPUS corpus (Lison and Tiedemann, 2016),[5] as well as 0.3G in books from various genres, drawn from LiberLiber[6] for Italian, Wikisource for Spanish and Russian, and Gutenberg[7] for English and French.

We focus on three embedding architectures: word2vec models (Mikolov et al., 2013) trained with gensim (Řehůřek and Sojka, 2010), the ELECTRA model of Clark et al. (2020), and character-based embeddings. The word2vec and ELECTRA models were selected so as to provide some comparison between static and contextual embeddings; both are trained with default hyperparameters aside from output vector size, which we set to 256. As for the ELECTRA models, given that we need contexts to derive token representations, we train the models only in English, French and Russian. The Spanish and Italian Wiktionary projects contain too few examples of usage. For French and Russian, we derive contextualized embeddings of a word to be defined from usage examples in DBnary datasets. Since the English DBnary dataset does not contain examples of usage, we extracted them from the original Wiktionary dumps.

The character-based embeddings are included to provide baseline expectations for non-semantic representations—as we can expect spelling to be more or less arbitrary with respect to word meaning (Saussure, 1916).[8] In practice, these embeddings are computed through a simple LSTM-based auto-encoder: the word is passed into an LSTM encoder as a sequence of characters, we sum all output hidden states, and use these summed hidden states to initialize an LSTM decoder, whose objective is to reconstruct the input word. As a character-based representation, we can therefore use the summed output hidden states, as they are tailored to contain all the information necessary to reconstruct the spelling of the corresponding word.[9] The datasets used to trained the models correspond to the set of all word types attested in our base corpora described in Table 2. All models achieve a 99% reconstruction accuracy.

[4]See here: https://dumps.wikimedia.org/

[5]See https://opus.nlpl.eu/

[6]Cf. https://www.liberliber.it/online/

[7]See here: https://www.gutenberg.org/

[8]Nonetheless, see Gutiérrez et al. (2016), Kutuzov (2017), Dautriche et al. (2017) or Pimentel et al. (2019), all of which question this assumption.

[9]Given that we implement this module ourselves, we use a Bayesian Optimization algorithm (Snoek et al., 2012) to select hyperparameters for our five character auto-encoder. We use this process to decide learning rate, weight decay, dropout, β_1 and β_2 parameters of the AdamW optimizer, batch size, number of epochs over the full dataset, as well as whether to share a single weight matrix for encoder and decoder character embeddings.

word	**POS**	**gloss**
sminuire	V	far figurare qualcosa o qualcuno come meno importante o rilevante

(a) Example definition in Italian

```
{
    "id": "it.42",
    "word": "sminuire"
    "gloss": "far figurare...",
    "pos": "v",
    "electra": [0.4, 0.2, ...],
    "sgns": [0.2, 0.4, ...],
    "char": [0.3, 1.4, ...],
}
```

(b) Corresponding JSON snippet

Figure 2: Toy example data point in the Italian dataset

3.3 Datasets

We construct one dataset per language. Each language-specific dataset is split in five: a trial split (200 datapoints per language), a training split (43 608 datapoints), a validation split (6375 datapoints), a definition modeling testing split (6221 datapoints) and a reverse-dictionary testing split (6208 datapoints). Splits are constructed such that there are no overlap in the embeddings. Dataset splits are formatted as JSON files.

Each file consists of a list of JSON dictionary notations. JSON items contain a unique identifier for the data point, the word being defined, definition, part of speech, and all word vectors. A depiction of the sort of items included in our datasets is shown in Figure 2. Sub-figure 2a summarizes the data presented as a JSON item in Sub-figure 2b.

Participants had access to the trial, train and validation splits of all languages. Test splits were made available at the beginning of the evaluation period.

4 The scales we use

We now turn to the metrics of our shared task.

4.1 Reverse Dictionary Metrics

The Reverse Dictionary task, as we have re-framed it here, consists in reconstructing embeddings. To that end, we consider three measures of vector similarity. First is MSE (mean squared error), which measures the difference between the components of the reconstructed and target embeddings. Mean-squared error is however not very easy to interpret on its own. Second is cosine: the reconstructed and target embeddings should have a cosine of 1. It is hard to place specific expectations for what a random output would produce, as this essentially differs from architecture to architecture: for instance, Transformer outputs are known to be anisotropic, so we shouldn't expect two random ELECTRA embeddings to be orthogonal (Ethayarajh, 2019; Timkey and van Schijndel, 2021, a.o.).

As neither MSE nor cosine provides us with a clear diagnosis tool comparable across all targets, we also include a ranking based measure: we compare the cosine of the reconstructed embedding $\vec{p}_i$ and the target embedding $\vec{t}_i$ to the cosine of the reconstruction $\vec{p}_i$ and all other targets $\vec{t}_j$ in the test set, and evaluate the proportion of such targets that would yield a closer association—viz., the number of cosine values greater than $\cos(\vec{p}_i, \vec{t}_i)$. More formally, we can describe this ranking metric as:

$$\text{Ranking}(\vec{p}_i) = \frac{\sum\limits_{\vec{t}_j \in \text{Test set}} \mathbb{1}_{\cos(\vec{p}_i, \vec{t}_j) > \cos(\vec{p}_i, \vec{t}_i)}}{\#\text{Test set}} \tag{1}$$

4.2 Definition Modeling Metrics

A common trope in NLG is to stress the dearth of adequate automatic metrics. Most of the metrics currently existing focus on token overlap, rather than semantic equivalence. The very popular BLEU and ROUGE metrics (Papineni et al., 2002; Lin, 2004) measure the overlap rate in n-grams of various lengths (usually 1-grams to 4-grams).

To alleviate this, researchers have suggested using external resources, such as lists of synonyms and stemmers (Banerjee and Lavie, 2005) or pre-trained language models (Zhao et al., 2019). The reliance of these augmented metrics on external resources is problematic. Different languages will use different resources with varying degrees of quality—and this will necessarily impact scores, introducing a confounding factor for any analysis down the line. In the extreme case, if these resources are not available for a particular language, then the metric will have to be discarded. Even assuming the availability of the required external resources, none of these improved metrics is entirely satisfactory. In the case of synonymy-aware metrics such as METEOR (Banerjee and Lavie, 2005), we can stress that syntactically different sentences can express the same meaning, but would not be captured by such metrics. Embeddings-based metrics such as MoverScore (Zhao et al., 2019) are very recent, and therefore less well understood; moreover concerns can be raised about whether using a method derived from neural networks trained on text will prove of any help in studying the meaning of texts generated by other neural networks.

One alternative frequently used by the NLG community is perplexity, which weighs the probability that the model would generate the target. This last alternative is however not suited to a shared task setup, as it requires us to have access to the actual neural networks trained by participants so we can investigate the probability distributions they model—unlike the other metrics we mentioned thus far, which only require model outputs.

In short, none of the currently available NLG metrics are fully satisfactory. Some are not applicable given the shared task format, some depend on external resources of varying quality, and some merely measure formal similarity, rather than semantic equivalence. Our approach is therefore twofold: on the one hand, we select multiple metrics with the expectation that each might shed light on one specific factor; on the other hand, we encourage participants to go beyond automatic scoring for the evaluation of their model.

As for which metrics we select, we narrow our choice to three. First is a basic BLEU score (Papineni et al., 2002) between a production p_i and the associated target t_i; our reasoning here is that as it is one of the most basic metrics, it is a consistent default choice. Second is the maximum BLEU score between a production p_i and any of the targets $t_i, t_j \dots t_n$ for which the definiendum is the same as that of p_i. This second metric is designed to not penalize models that rely solely on SGNS or char embeddings: as the input would always be the same, deterministic models would always produce the same definition $p_i = p_j = \dots = p_n$.[10] To distinguish between our two BLEU variants, we refer to the former as S-BLEU (or Sense-BLEU),

[10]One way of bypassing this problem would be to include a source of noise, as is done in GAN architectures (Goodfellow et al., 2014). This would still leave open the question of how to optimally align the outputs to the possible targets.

and the latter as L-BLEU (or Lemma-BLEU).

Given that some definitions in our dataset can be very short, we also apply a smoothing to both BLEU-based metrics. In practice, BLEU computes an overlap of n-grams of size m and under; by default, $m = 4$. This overlap is a geometric mean across all n-gram sizes $1 \ldots m$. If a definition d contains less than m tokens, then any associated production for which d is used as a target will contain 0 overlapping n-grams of size m. The use of a geometric mean then entails that the BLEU score for any production associated to d will be 0. To circumvent this limitation of BLEU, it is common to use some form of smoothing. Here, for any n-gram size $\hat{m}$ that would yield an overlap of 0 (i.e., $\hat{m}$ such that $\#d < \hat{m} \leq m$), we replace the overlap count with a pseudocount of $1/\log \#d$.

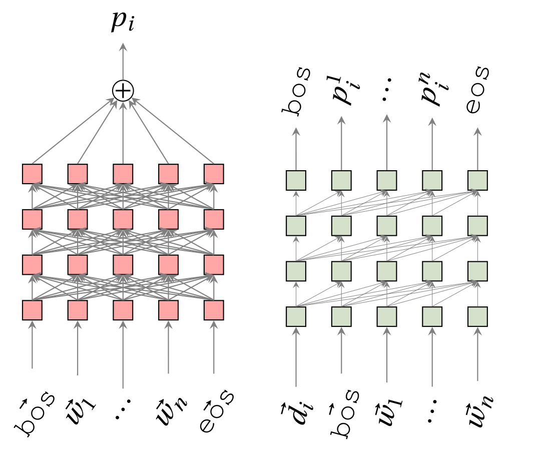
Lastly, we include MoverScore (Zhao et al., 2019), using a multilingual DistilBERT model as the external resource. The fact that this model is multilingual means that we can use it for all five languages of interest. Embedding-based methods have the potential to overcome some of the limitations of purely token-based metrics, which is why we deem them worth including in our setup.

The second part of our approach for evaluating submissions consists in encouraging participants to not rely solely on the automatic scoring system of their outputs. Concretely, we provide participants with a richly annotated trial dataset, which contains frequency and hand-annotated semantic information, and strongly suggest participants to use it for a manual evaluation of their system. We include the presence of a manual evaluation as a criterion to evaluate the quality of a system description paper, and plan to formally recognize the most enlightening evaluations conducted by participants.

Neither our selection of metrics nor our insistence on manual evaluation solves the evaluation issues of NLG systems. We duly note the importance of this question, and plan to conduct a follow-up evaluation campaign on the CoDWoE submissions.

5 Testing the waters: baseline architectures

We implement simple neural network architecture baselines to lower the barrier to entry to this shared task. They are based on the Transformer architecture of Vaswani et al. (2017) and designed to be as simple as possible. Our code is publicly available at `https://github.com/TimotheeMickus/codwoe`.



(a) Reverse dictionary (b) Definition modeling

Figure 3: Baseline architectures for the CoDWoE shared task

We illustrate our Reverse Dictionary baseline architecture in Figure 3a. It consists in feeding the input gloss $\langle \vec{bos}, \vec{w}_1, \ldots, \vec{w}_n, \vec{eos} \rangle$ into a simple Transformer encoder, and then summing all the hidden representations to produce the prediction p_i. In practice, the summed hidden states are passed into a small non-linear feed-forward module to derive the prediction:

$$p_i = W_p \left(\text{ReLU} \left(\sum_t \vec{h}_t \right) \right) \tag{2}$$

Our Definition Modeling baseline is presented in Figure 3b. It consists in a simple Transformer encoder, where earlier time-step representations are prevented from attending to later time-step representations. To provide information about the definiendum to the model, we use the definiendum embedding $\vec{d}_i$ as the input for the first time-step instead of a start-of-sequence token. We train the models with teacher-forcing: i.e., during training we ignore the definientia $p_i^1, \ldots, p_i^n$ that the model produces; instead we feed it the target $w_1, \ldots, w_m$ attested in the training set at each time-step. During inference, we feed the model with its own prediction. This creates a train-test mismatch, which we alleviate by using a beam-search. We stop generation when all beams have produced an end-of-sequence token.

For both tracks, we train one model for each distinct pair of language and embedding architecture. We start by re-tokenizing the datasets using sentence piece with a vocabulary size of 15000. This is done in order to mitigate the effects of different

Team	en			es			fr			it			ru		
	Mv	SB	LB	Mv	SB	LB	Mv	SB	LB	Mv	SB	LB	Mv	SB	LB
Bl. SGNS	0.084	0.030	0.040	0.065	0.035	0.052	0.046	0.030	0.041	0.107	0.053	0.076	0.112	0.039	0.054
Bl. char	0.047	0.026	0.033	0.059	0.031	0.043	0.022	0.028	0.037	0.046	0.029	0.038	0.072	0.025	0.037
Bl. Electra	0.065	0.031	0.039				0.043	0.031	0.039				0.101	0.032	0.041
Locchi	0.049	0.022	0.027	0.038	0.020	0.026				0.071	0.008	0.012			
LingJing	−0.045	0.004	0.005	0.023	0.013	0.020	−0.113	0.003	0.005	−0.012	0.018	0.029	−0.010	0.011	0.014
BLCU-ICALL	**0.135**	0.031	0.040	**0.128**	0.039	0.056	0.042	0.027	0.037	**0.117**	**0.066**	**0.099**	**0.148**	0.048	0.065
IRB-NLP	0.094	**0.033**	0.042	0.093	**0.045**	**0.064**	0.056	0.028	0.033	0.077	0.010	0.015	0.080	0.027	0.036
RIGA	0.093	0.026	0.032	0.107	0.031	0.045	**0.075**	0.024	0.030	0.093	0.012	0.018	0.094	0.031	0.043
lukechan1231	0.071	0.022	0.027	0.068	0.025	0.036	0.054	0.021	0.026	0.101	0.037	0.054	0.109	0.029	0.040
Edinburgh	0.104	0.031	0.038	0.101	0.035	0.053	0.026	**0.029**	**0.038**	0.107	0.060	0.092	0.109	**0.049**	**0.072**
talent404	0.128	0.033	**0.043**												

Table 3: Participants' best scores on the Definition Modeling track. Highest participant scores per metric are displayed in bold font.

vocabulary sizes when training our Transformer baselines, and make the models overall easier to compare across different languages.

We set hyperparameters using a Bayesian Optimization procedure, with 100 hyperparameter configurations tested and 10 initial random samples. For the Reverse dictionary models, we tune the following hyper-parameters: learning rate, weight decay penalty, the β_1 and β_2 hyperparameters of the Adam optimizing algorithm, dropout rate, length of warmup, batch size,[11] number of heads in the multi-head attention layers, and number of stack layers. For the Definition Modeling systems, we also include a label smoothing parameter to tune. Models are trained over up to 100 epochs; training is stopped early if no improvement of at least 0.1% is observed during 5 epochs. In all cases, we decay the learning rate after the warmup following a half cosine wave, such that the learning rate reaches 0 at the end of the 100 epochs.

6 How whale did it go? Shared task results.

Scores attained by participants are shown in Tables 3 and 4. In Table 3, "Mv", "SB" and "LB" refer to Moverscore, Sense-BLEU and Lemma-BLEU respectively; in Table 4, each sub-table corresponds to a different architecure, and "rnk" refers to the cosine ranking metric (cf. Section 4).

In total, we received 159 valid submissions from 15 different users; out of which 11 teams produced a submission paper. 9 of these teams tackled the Definition Modeling, and 10 addressed the reverse dictionary track. Competition rankings are established by ranking each submission received, selecting for each participant the best performance on all metrics, and finally taking the average best rank. Some participants' submissions were faulty and could not be processed by the evaluation website scoring program.

Among the system descriptions we received, two focused solely on definition modeling. Kong et al. (2022, BLCU-ICALL) use a multitasking framework for definition modeling, based on a generation and a reconstruction objectives. Mukans et al. (2022, RIGA) focus on what are the effects of model size and duration of training on GRUs and LSTMs for definition modeling, and whether MoverScore corroborates human judgment.

Five submissions specifically focus on the reverse dictionary task. Bendahman et al. (2022, BL.research) compare the performances of MLP-based to LSTM-based networks for reverse dictionary. Li et al. (2022, LingJing) study pretraining objectives for the reverse dictionary track. Ardoiz et al. (2022, MMG) pay specific attention to how the not-so-satisfactory quality of the Spanish dataset impacts results on Spanish reverse dictionary. Cerniavski and Stymne (2022, Uppsala) study whether foreign language entries can improve the performance of the English reverse dictionary baseline model. Wang et al. (2022, 1cademy) introduce multiple technical tweaks for reverse dictionary, such as a dynamic weight averaging loss, language-specific tags and residual cutting.

[11] In practice, we first manually find the largest batch size that fits on our GPU, and then let the model select the number of batches it should accumulate gradient on.

Team	en			es			fr			it			ru		
	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk
Baseline	0.911	0.151	0.490	0.930	0.204	0.499	1.141	0.198	0.491	1.125	0.204	0.477	0.577	0.253	0.490
Locchi	0.875	0.204	0.394							1.087	0.274	0.386			
BL.research	0.895	0.166	0.312	0.910	0.252	0.253	1.107	0.212	0.314	1.111	0.246	0.247	0.566	0.298	0.290
LingJing	0.862	0.243	0.329	**0.858**	0.353	0.251	1.030	0.328	0.282	1.039	0.360	0.230	**0.528**	**0.424**	0.187
MMG				0.911	**0.403**	**0.167**									
chlrbgus321	**0.854**	0.248	0.319												
IRB-NLP	0.964	**0.260**	**0.231**	0.883	0.367	0.197	1.068	**0.342**	**0.193**	1.076	**0.380**	**0.165**	0.568	0.421	**0.150**
Edinburgh	0.864	0.241	0.326	0.860	0.347	0.271	**1.026**	0.312	0.302	**1.031**	0.374	0.197	0.538	0.383	0.247
theOne	0.900	0.185	0.500												
JSI	0.909	0.156	0.499	0.913	0.223	0.495	1.122	0.216	0.498	1.196	−0.004	0.499	0.615	0.006	0.499
1cadamy	0.915	0.194	0.374	0.906	0.262	0.375	1.100	0.228	0.439	1.097	0.260	0.384	0.578	0.335	0.291

(a) SGNS Reverse Dictionary track results

Team	en			es			fr			it			ru		
	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk
Baseline	0.148	0.790	0.502	0.570	0.806	0.498	0.395	0.759	0.499	0.363	0.727	0.497	0.135	0.826	0.495
Locchi	**0.141**	**0.798**	0.483							0.355	0.734	0.478			
BL.research	0.143	0.795	0.450	0.510	0.824	0.412	0.366	0.770	0.428	0.359	0.728	0.417	0.132	0.830	0.410
LingJing	0.176	0.782	0.486	0.583	0.824	0.500	0.411	0.752	0.502	0.438	0.681	0.496	0.184	0.791	0.472
IRB-NLP	0.162	0.770	**0.419**	0.526	0.819	**0.403**	0.390	0.756	0.421	0.366	0.724	**0.383**	0.140	0.824	**0.357**
Edinburgh	0.143	0.795	0.500	**0.467**	**0.839**	0.424	**0.335**	**0.789**	0.428	**0.334**	**0.747**	0.428	**0.116**	**0.852**	0.389
theOne	0.143	0.796	0.500												
1cadamy	0.168	0.792	0.478	0.557	0.820	0.410	0.391	0.769	**0.416**	0.364	0.739	0.438	0.156	0.836	0.377

(b) Char Reverse Dictionary track results

Team	en			fr			ru		
	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk
Baseline	1.413	0.843	0.498	1.153	0.856	0.498	0.874	0.721	0.491
Locchi	**1.301**	0.843	0.478						
BL.research	1.326	0.844	0.434	1.112	0.858	0.442	0.864	0.721	0.399
LingJing	1.509	0.846	0.478	1.271	0.859	0.478	**0.828**	0.734	0.420
IRB-NLP	1.685	0.828	**0.432**	1.339	0.847	**0.429**	0.911	0.724	**0.345**
Edinburgh	1.310	**0.847**	0.490	**1.066**	**0.862**	0.476	0.828	**0.735**	0.417
theOne	1.340	0.846	0.500						

(c) ELECTRA Reverse Dictionary track results

Table 4: Participants' best scores on the Reverse Dictionary track. Highest participant scores per metric are displayed in bold font.

The last four submissions addressed both tracks. Chen and Zhao (2022, Edinburgh) propose to project embeddings and definitions on a shared representational space. Korenčić and Grubišić (2022, IRB-NLP) take inspiration from Noraset et al. (2017) to address definition modeling, and experiment with pooling strategies over Transformer embeddings for the reverse dictionary track. Tran et al. (2022, JSI) focus on comparing the effects of adding LSTM and BiLSTM layers on top of a Transformer model, as well as zero-shot cross-lingual generalization. Srivastava and Harsha Vardhan (2022, TLDR) propose two Transformer-based architectures for the two tracks, leveraging contrastive learning and unsupervised pretraining.

Looking at Tables 3 and 4, we see that the metrics we chose in section 4 are not always aligned. On the Definition Modeling track (Table 3), while the multitask framework of Kong et al. (2022, BLCU-ICALL) yields generally the most consistent performance, it is often outmatched in specific setups. For instance, BLEU-based metrics favor the shared projection technique of Chen and Zhao (2022, Edinburgh) in Russian and French, while the pooling strategies of Korenčić and Grubišić (2022, IRB-NLP) appear especially effective on the Spanish dataset. As for the Reverse Dictionary track (Table 4), the strongest contender is generally the Edinburgh team, although the IRB-NLP team almost systematically produces the highest cosine ranking score. Interestingly, BLCU-ICALL, IRB-NLP and Edinburgh all rely on multi-task learning. Note however that the SGNS targets seem to depict a rather different picture, where the pretraining objectives of Li et al. (2022, LingJing) bring about some of the best results.

7 A deeper dive into our results

When looking at the competition results, two trends emerge. First, the baseline architectures from Section 5 remain quite competitive with solutions proposed by participants. Second, scores are generally unsatisfactory, especially in the definition modeling track: we do not see a clear divide between char embeddings and distributional semantic representations. The NLG metrics are, in absolute terms, low compared to modern NLP standards and results reported elsewhere on other definition modeling benchmarks. As for the reverse dictionary track, we see that across all submissions, at least a third of the test set is closer (in terms of cosine distance) to the production than the intended target.

Participants have suggested multiple reasons for these hardships. In particular, Ardoiz et al. (2022, MMG) highlight that the automated data compilation in DBnary (Sérasset, 2012) is of an unsatisfactory quality. Similar remarks can be made with respect to the embeddings, which are trained on rather small corpora. Other submissions such as Mukans et al. (2022, RIGA), Chen and Zhao (2022, Edinburgh), Korenčić and Grubišić (2022, IRB-NLP) highlight the limited applicability of mainstream NLG metrics, as we ourselves have discussed in Section 4.[12] One last remark is the limited size of our dataset, discussed by the Edinburgh and RIGA teams. All these remarks suggest avenues for future research: in particular, the release of the full dataset should alleviate some of the concerns with respect to dataset size. The MMG team also suggest some concrete preprocessing steps to handle some of the issues they identify in the proposed definitions.

[12]See also Mickus et al. (2021) for a discussion.

In terms of solutions explored, we can stress that teams have adopted a variety of strategies and architectures: systems used Transformer, RNN and CNN components, often leveraging or exploring multilingualism (Tran et al. 2022, JSI; Cerniavski and Stymne 2022, Uppsala; Wang et al. 2022, 1cademy; Bendahman et al. 2022, BL.research), multitasking, or multiple training objectives (Kong et al. 2022, BLCU-ICALL; 1cadamy; Korenčić and Grubišić 2022, IRB-NLP; Srivastava and Harsha Vardhan 2022, TLDR; Chen and Zhao 2022, Edinburgh). Multi-task training tends to yield varied yet competitive results for our data. No preponderant architecture emerges from the system descriptions; we note that multiple submissions based their work on other contextualized embedding architectures, trained from scratch on the CODWOE dataset (Wang et al. 2022, 1cademy; Li et al. 2022, LingJing). The comprehensive review of architectures by team 1cadamy suggests nonetheless that Transformers might be less suited to this shared task than recurrent models.

7.1 Manual analyses

As for manual evaluations, Kong et al. (2022, BLCU-ICALL) provide a thorough review of the errors produced by their model. Mukans et al. (2022, RIGA) provide some example outputs of their models, while Srivastava and Harsha Vardhan (2022, TLDR) and Wang et al. (2022, 1cademy) include ablation studies. The most thorough analysis, however, is that of Chen and Zhao (2022, Edinburgh), who provide both quantitative and qualitative (PCA-based) analyses across embedding architectures, languages, and trial dataset features. Korenčić and Grubišić (2022, IRB-NLP) provide an extremely well documented review of their systems performances, along multiple analyses of the embeddings proposed for the shared tasks, ranging from 2D down-projection visualizations to descriptive statistics of components. We refer the reader to the respective system papers for a more thorough review and focus here on a few promising approaches to summarize trends that emerge from these manual analyses.

Current metrics are not satisfactory. The IRB-NLP team highlight that the BLEU scores reported on the shared task are dramatically lower than what is generally expected in the literature; the Edinburgh team even shows that the S-BLEU scores obtained by non-sensical glosses such as "`, or .`"

can end up among the highest scores for some languages. The Reverse Dictionary metrics can also be sensitive to different aspects of the embeddings, as shown by the IRB-NLP team: this can lead to very different rankings of model productions, especially when comparing the cosine-based ranking metric to the cosine and MSE metrics. BLEU-based scores are also often sensitive to the length of the production, the target, or both, as shown by both the Edinburgh and the Riga teams.

Erroneous productions abound. Related to the previous remark, many Definition Modeling systems produce irrelevant or under-specified glosses, for which the proposed metrics are not satisfactory. For instance, the BLCU-ICALL report 52% irrelevant glosses and 23.5% under-specified glosses, from a manual evaluation of 200 productions. Other participating teams, such as RIGA or IRB-NLP, also display generated glosses with varying degress of semantic accuracy.

Embeddings contain more than semantics. The Edinburgh team highlights how different linguistic features retrieved from the trial dataset can significantly impact the scores they observe. They also highlight that char embeddings are separable by length, and that the Electra embeddings are clustered according to their frequency.

Not all setups are created equal. The Uppsala team report that Russian seems to be the most effective data source in their multilingual transfer experiments. The IRB-NLP team stresses that vector component distributions across languages and architectures as well as gloss length across languages can take very different values, and they also include 2D visualization suggesting the Electra embeddings tend to form neat cluster not observed for SGNS embeddings. Scores also vary quite a lot across setups (cf. Tables 3 and 4).

8 Conclusions and future perspectives

The CODWOE shared task was constructed so that participants' submissions would be likely to have linguistic significance. Yet, it is not trivial to tease apart the various factors that lead to the overall low results we observed. While the inadequacy of mainstream NLG metrics and the limitations of the dataset certainly play a role, they do not resolve the fundamental issue that we wished to investigate with CODWOE. Whether word embeddings and dictionaries contain the same information is not a solved research problem.

This has two immediate consequences: firstly, one can question the use of definition modeling as an evaluation tool for embeddings, as suggested by the seminal work of Noraset et al. (2017). The CODWOE shared task results indicate that the metrics currently used in the field are rife with caveats; in the controlled setup we have proposed here, participants rarely, if ever, found that character-based embeddings starkly contrasted with distributional semantic representations.

Second, one can question whether definition modeling and reverse dictionary are fit for building lexical resources for under-tooled languages: the crosslingual route proposed by Bear and Cook (2021) seems more practical than training models from scratch, even with relatively large datasets. Our embeddings were trained on corpora comparable in size to the 1B Words benchmark (Chelba et al., 2013): while modern text corpora are now several orders of magnitude larger, this dataset remained a landmark for several years. Our definitions were selected from DBnary (Sérasset, 2012), which focuses the largest Wiktionary projects.

Overall, the CODWOE shared task has been a success: we were able to show that the task at hand was far from trivial and we drew significant interest towards the issues addressed in the Definition Modeling and Reverse Dictionary literature. In future work, we plan to investigate better ways to perform NLG evaluation for the Definition Modeling task (in particular relying on human annotations) and we plan to focus on existing embeddings trained from very large corpora.

Acknowledgements

We thank the annotators who have helped us in putting an annotated dataset in a very short amount of time: Elena del Olmo Suárez, Hermès Martinez, Maria Copot, Nikolay Chepurnykh and Toma Gotkova, as well as Cyril Pestel for his help with data hosting.

This work was also supported by a public grant overseen by the French National Research Agency (ANR) as part of the "Investissements d'Avenir" program: Idex *Lorraine Université d'Excellence* (reference: ANR-15-IDEX-0004).

References

Alfonso Ardoiz, Miguel Ortega-Martín, Óscar García-Sierra, Jorge Álvarez, Ignacio Arranz, and Adrián Alonso. 2022. MMG at SemEval-2022 Task 1: A reverse dictionary approach based on a review of the dataset from a lexicographic perspective.

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

Diego Bear and Paul Cook. 2021. Cross-lingual wolastoqey-English definition modelling. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 138–146, Held Online. INCOMA Ltd.

Nihed Bendahman, Julien Breton, Lina Nicolaieff, Mokhtar Boumedyen Billami, Christophe Bortolaso, and Youssef Miloudi. 2022. BL.Research at SemEval-2022 Task 1: Deep networks for reverse dictionary using embeddings and lstm autoencoders.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Slaven Bila, Wataru Watanabe, Taiichi Hashimoto, Takenobu Tokunaga, and Hozumi Tanaka. 2004. Dictionary search based on the target word description. In *Proceedings of the 10th Annual Meeting of the Association for Natural Language Processing (ANLP 2004)*.

Gemma Boleda. 2020. Distributional semantics and linguistic theory. *Annual Review of Linguistics*, 6(1):213–234.

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

Hiram Calvo, Oscar Méndez, and Marco A. Moreno-Armendáriz. 2016. Integrated concept blending with vector space models. *Comput. Speech Lang.*, 40(C):79–96.

Rafal Cerniavski and Sara Stymne. 2022. Uppsala university at SemEval-2022 Task 1: Multilingualism in reverse dictionaries: Can foreign entries enhance an english reverse dictionary?

Ciprian Chelba, Tomás Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005.

Pinzhen Chen and Zheng Zhao. 2022. Edinburgh at SemEval-2022 Task 1: Jointly fishing for word embeddings and definitions.

Kevin Clark, Minh-Thang Luong, Quoc Le, and Christopher D. Manning. 2020. Pre-training transformers as energy-based cloze models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 285–294, Online. Association for Computational Linguistics.

Isabelle Dautriche, Kyle Mahowald, Edward Gibson, and Steven T. Piantadosi. 2017. Wordform similarity increases with semantic similarity: An analysis of 100 languages. *Cognitive Science*, 41(8):2149–2169.

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

Dominique Dutoit and Pierre Nugues. 2002. A lexical database and an algorithm to find words from definitions. In *Proceedings of the 15th European Conference on Artificial Intelligence*, ECAI'02, page 450–454, NLD. IOS Press.

Ilknur Durgar El Khalout and Kemal Oflazer. 2004. Use of wordnet for retrieving words from their meanings. In *Proceedings of the Second Global Wordnet Conference (GWC 2004)*, pages 118–123.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 266–271, Melbourne, Australia. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information*

Processing Systems, volume 27. Curran Associates, Inc.

E. Dario Gutiérrez, Roger Levy, and Benjamin Bergen. 2016. Finding non-arbitrary form-meaning systematicity using string-metric learning for kernel regression. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2379–2388, Berlin, Germany. Association for Computational Linguistics.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Arman Kabiri and Paul Cook. 2020. Evaluating a multi-sense definition generation model for multiple languages. In *Text, Speech, and Dialogue*, pages 153–161, Cham. Springer International Publishing.

Cunliang Kong, Yujie Wang, Ruining Chong, Liner Yang, Hengyuan Zhang, Erhong Yang, and Yaping Huang. 2022. BLCU-ICALL at SemEval-2022 Task 1: Cross-attention multitasking framework for definition modeling.

Damir Korenčić and Ivan Grubišić. 2022. IRB-NLP at SemEval-2022 Task 1: Exploring the relationship between words and their semantic representations.

Andrei Kutuzov. 2017. Arbitrariness of linguistic sign questioned: Correlation between word form and meaning in russian.

Alessandro Lenci. 2018. Distributional models of word meaning. *Annual Review of Linguistics*, 4(1):151–171.

Bin Li, Yixuan Weng, Fei Xia, Shizhu He, Bin Sun, and Shutao Li. 2022. Lingjing at SemEval-2022 Task 1: Multi-task self-supervised pre-training for multilingual reverse dictionary.

Yinqiao Li, Chi Hu, Yuhao Zhang, Nuo Xu, Yufan Jiang, Tong Xiao, Jingbo Zhu, Tongran Liu, and Changliang Li. 2020. Learning architectures from an extended search space for language modeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6629–6639, Online. Association for Computational Linguistics.

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

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Arman Malekzadeh, Amin Gheibi, and Ali Mohades. 2021. PREDICT: persian reverse dictionary. *CoRR*, abs/2105.00309.

Oscar Méndez, Hiram Calvo, and Marco A. Moreno-Armendáriz. 2013. A reverse dictionary based on semantic analysis using wordnet. In *Advances in Artificial Intelligence and Its Applications*, pages 275–285, Berlin, Heidelberg. Springer Berlin Heidelberg.

Timothee Mickus, Mathieu Constant, and Denis Paperno. 2020a. Génération automatique de définitions pour le français (definition modeling in French). In *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Volume 2 : Traitement Automatique des Langues Naturelles*, pages 66–80, Nancy, France. ATALA et AFCP.

Timothee Mickus, Mathieu Constant, and Denis Paperno. 2021. About neural networks and writing definitions. *Dictionaries: Journal of the Dictionary Society of North America*, 42(2).

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2020b. What do you mean, BERT? In *Proceedings of the Society for Computation in Linguistics 2020*, pages 279–290, New York, New York. Association for Computational Linguistics.

Timothee Mickus, Denis Paperno, and Matthieu Constant. 2019. Mark my word: A sequence-to-sequence approach to definition modeling. In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 1–11, Turku, Finland. Linköping University Electronic Press.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Eduards Mukans, Gus Strazds, and Guntis Barzdins. 2022. RIGA at SemEval-2022 Task 1: Scaling recurrent neural networks for CODWOE dictionary modeling.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3259–3266. AAAI Press.

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

Tiago Pimentel, Arya D. McCarthy, Damian Blasi, Brian Roark, and Ryan Cotterell. 2019. Meaning to form: Measuring systematicity as information. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1751–1764, Florence, Italy. Association for Computational Linguistics.

Fanchao Qi, Lei Zhang, Yanhui Yang, Zhiyuan Liu, and Maosong Sun. 2020. Wantwords: An open-source online reverse dictionary system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–181.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Ferdinand de Saussure. 1916. *Cours de linguistique générale*, 1995 edition. Payot & Rivage, Paris.

Gilles Sérasset. 2012. Dbnary: Wiktionary as a LMF based multilingual RDF network. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2466–2472, Istanbul, Turkey. European Language Resources Association (ELRA).

Ryan Shaw, Anindya Datta, Debra VanderMeer, and Kaushik Dutta. 2013. Building a scalable database-driven reverse dictionary. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):528–540.

Bushra Siddique and Mirza Mohd Sufyan Beg. 2019. A review of reverse dictionary: Finding words from concept description. In *Next Generation Computing Technologies on Computational Intelligence*, pages 128–139, Singapore. Springer Singapore.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Aditya Srivastava and Vemulapati Harsha Vardhan. 2022. TLDR at SemEval-2022 Task 1: Using transformers to learn dictionaries and representations.

Sushrut Thorat and Varad Choudhari. 2016. Implementing a reverse dictionary, based on word definitions, using a node-graph architecture. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2797–2806, Osaka, Japan. The COLING 2016 Organizing Committee.

William Timkey and Marten van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4527–4546, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thi Hong Hanh Tran, Matej Martinc, Matthew Purver, and Senja Pollak. 2022. JSI at SemEval-2022 Task 1: CODWOE - reverse dictionary: Monolingual and cross-lingual approaches.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Zhiyong Wang, Ge Zhang, and Nineli Lashkarashvili. 2022. 1cademy at SemEval-2022 Task 1: Investigating the effectiveness of multilingual, multitask, and language-agnostic tricks for the reverse dictionary task.

Hang Yan, Xiaonan Li, Xipeng Qiu, and Bocao Deng. 2020. BERT for monolingual and cross-lingual reverse dictionary. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4329–4338, Online. Association for Computational Linguistics.

Liner Yang, Cunliang Kong, Yun Chen, Yang Liu, Qinan Fan, and Erhong Yang. 2019. Incorporating sememes into chinese definition modeling. ArXiv preprint : 1905.06512.

David Yenicelik, Florian Schmidt, and Yannic Kilcher. 2020. How does BERT capture semantics? a closer look at polysemous words. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 156–162, Online. Association for Computational Linguistics.

Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1263–1271, Beijing, China. Coling 2010 Organizing Committee.

Haitong Zhang, Yongping Du, Jiaxin Sun, and Qingxiao Li. 2020a. Improving interpretability of word embeddings by generating definition and usage. *Expert Systems with Applications*, 160:113633.

Lei Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2020b. Multi-channel reverse dictionary model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 312–319.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

1Cademy at Semeval-2022 Task 1: Investigating the Effectiveness of Multilingual, Multitask, and Language-Agnostic Tricks for the Reverse Dictionary Task

Zhiyong Wang[1 4*], Ge Zhang[2 4*], Nineli Lashkarashvili[3 4]
[1]University of Colorado Boulder, USA
[2]University of Michigan Ann Arbor, USA
[3]San Diego State University, USA
[4]1Cademy Community, USA
Zhiyong.Wang@colorado.edu

Abstract

This paper describes our system for the SemEval2022 task of matching dictionary glosses to word embeddings. We focus on the Reverse Dictionary Track of the competition, which maps multilingual glosses to reconstructed vector representations. More specifically, models convert the input of sentences to three types of embeddings: SGNS, Char, and Electra. We propose several experiments for applying neural network cells, general multilingual and multitask structures, and language-agnostic tricks to the task. We also provide comparisons over different types of word embeddings and ablation studies to suggest helpful strategies. Our initial transformer-based model achieves relatively low performance. However, trials on different retokenization methodologies indicate improved performance. Our proposed Elmo-based monolingual model achieves the highest outcome, and its multitask, and multilingual varieties show competitive results as well.

1 Introduction

Reverse dictionary Task is defined as word generation based on user descriptions (Hill et al., 2016). Following competition rules, pre-trained models and external information should be avoided, and large-scale language models are unsuitable for the task. Our paper is devoted to the performance comparison of different neural network structures, multilingual and multitask tricks, and elaborating on language-agnostic or bidirectional structure helpfulness. The competition (Mickus et al., 2022) has significant potential in contributing pretraining process acceleration, low-resource language model development, and commonsense using. Furthermore, the task is of high importance for explainable AI and natural language processing since it models direct mapping from human-readable data to machine-readable data.

Known word representation methods using dictionaries, knowledge databases, or glosses have been a common approach for years. Related models can be divided into two major groups. In the former, category methods highly rely on large-scale model construction. Levine et al. (2019) develop SenseBert, introducing super-senses from Wordnet (Miller, 1995) into general Bert model. Ernie (Sun et al., 2019) combines node embeddings from knowledge graph and matched entities to enhance word representations. KnowBert (Peters et al., 2019) subsumes the entity connection and Bert models, which are trained together. There are similar research works relevant to the topic (Wang et al., 2021, 2020; Yin et al., 2020). Still, their models' performances are dependent on the basic large-scale language model trained by sentence samples. In the latter group, traditional dependency-based language models learn directly from word dependency and glosses. They have two major disadvantages: incompatibility with modern language models and relatively low performance (Tissier et al., 2017; Levy and Goldberg, 2014; Wieting et al., 2015). There is ambiguity about whether recent embeddings and dictionary glosses are mappable from each other.

The paper specifically focuses on progressing utilization of the glosses, different word representations, and languages. **First**, we discuss ablation studies for language-agnostic trick, bidirectional, multilingual, and multitask models and explain the experimental results. **Second**, we apply and analyze different re-tokenization methods. **Finally**, we give instructive conclusions about encoder structures, distinctive word representation relations, and cross-lingual dictionary performance based on our experiment results. We find that **(1)** transformer-based model performance is inferior to other models for its high complexity, **(2)** bidirectional models with similar parameter size outperform the unidirec-

[*] The two authors contributed equally to this work.

tional model because of their better understanding of context-environments even in the low-resource condition, and **(3)** different word embeddings have a potential relations and can be collaboratively learnt from glosses using a multitask learning structure. We make our codes and results publicly available[1].

2 Task Description

The competition, comparing dictionaries and word embeddings, proposes definition modeling (Noraset et al., 2017) and reverse dictionary sub-tracks (Hill et al., 2016). These sub-tracks are designed to test the equivalency of dictionary glosses and word embedding representations. This paper focuses on the reverse dictionary direction. The task refers to word recalling using gloss input and provides word representations that are separately generated by word2vec (SGNS) (Mikolov et al., 2013), character (Wieting et al., 2016), and Electra (Clark et al., 2020) embeddings as training data. External data and large-scale language models are strictly restricted from this competition since the models might learn the word embeddings majorly from the sentence samples instead of the dictionary glosses. The words matched with the dictionary glosses are hidden in the datasets, implying that dependency-based word representation algorithms cannot be applied directly.

3 Methodology

To clarify, we affirm that we only refer to the model structures instead of the trained models when we mention Elmo and MBert in the section and use no external data.

3.1 Language Model Structure

Baseline monolingual models with five distinctive structures were trained: RNN, LSTM, Bi-RNN, Elmo, and Transformer.

We experiment how bidirectional and different feature generator cell structures help.

RNN is the classical deep learning model dealing with ordinal or sequential data (Zaremba et al., 2014). Its major disadvantage is the vanishing and exploding gradient issue. Nevertheless, the model is fast to converge and works well on smaller sentences. Our experiments show that RNN, having similar results to the LSTM-based model, performs slightly better than the transformer-based one.

[1] https://github.com/ravenouse/Revdict_1Cademy

LSTM is another classical and widely-used feature generator structure in natural language processing. The comparison of LSTM-based and RNN-based models can suggest whether vector representation of glosses suffers from the long-term dependencies problem. Earlier works (Jozefowicz et al., 2015) demonstrate that variants of LSTM achieve similar performances in the majority of natural language processing tasks. We select the classical LSTM structure for the experiments.

Transformer (Vaswani et al., 2017) is a milestone feature extractor allowing deeper neural network design for natural language processing tasks. However, given the much smaller size of the competition data, it performs relatively worse compared to the expectation.

3.2 Multitask Structure

Although character embedding generation has a similar algorithm to general word embedding methods, it focuses on character representation and is mightier to better tackle the Out Of Vocabulary (**OOV**) problem. We applied Mean Squared Loss (**MSE Loss**) and Dynamic Weight Averaging (**DWA**) (Liu et al., 2019) as a basic multitask structure for predicting word2vec, Char, and Electra embedding together. It achieves competitive performance in both tasks.

DWA (Liu et al., 2019) is designed for keeping different tasks converging at the same pace. N denotes the number of tasks, T adjusts the weight-changing sensitivity according to loss difference of the tasks, $L_n(t-1)$ and $r_n(t-1)$ represent the loss and the training speed of task n at $(t-1)$th step. $w_i(t)$ is the loss weight of task i at tth step. The key update equations can be expressed as follows:

$$w_i(t) = \frac{N\, exp(r_i(t-1)/T)}{\sum_n exp(r_i(t-1)/T)} \tag{1}$$

$$r_n(t-1) = \frac{L_n(t-1)}{L_n(t-2)} \tag{2}$$

3.3 Retokenize Algorithm

We tried 3 widely-used retokenization algorithms for vocabulary generation including Byte Pair Encoding (**BPE**) (Sennrich et al., 2015), **WordPiece** Model (Schuster and Nakajima, 2012), and Unigram Language Model (**ULM**) (Kudo, 2018). BPE is a greedy algorithm that can not model word relation probability successfully. WordPiece considers

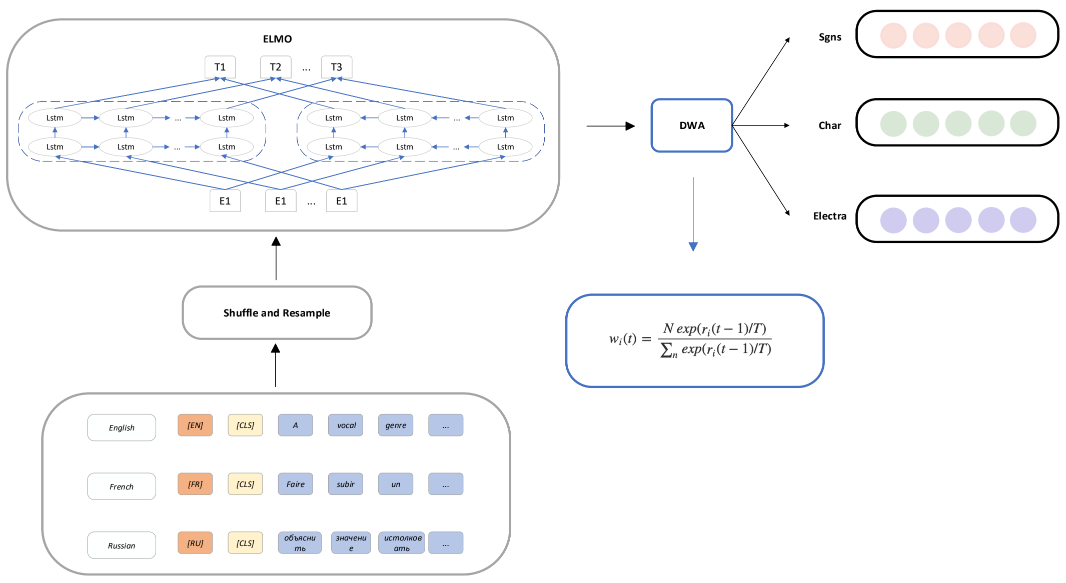


Figure 1: Sketch Map of the multilingual and multitask Elmo-based model structure.

word co-occurrence probability and is influenced by the source data. ULM assumes that all subwords are independent and the probability of a subword sequence is the multiple of its element subwords' probability.

3.4 Multilingual Structure

We applied two basic multilingual structures for the task: mBert (Pires et al., 2019) and adding the language tag. **MBert** has a shared vocabulary for all source languages. The results show that mBert can successfully model similar grammar structure, and sentences with similar meanings have akin representations using mBert. By applying mBert structure, we can estimate how these important conclusions would work for the reverse vocabularies task. We **add the language tag** as the first token to improve models' ability to separate different languages' representations.

We speculate that language-agnostic representations might aid multilingual models in achieving better performance. Residual connection cutting trick proposed by (Liu et al., 2020) was tried, to test how the research findings would work for our specific task.

3.5 Selected Model Design

Following experiment results and ablation studies, our best model is the monolingual Elmo with WordPiece tokenizer. The Multitask and multilingual tricks have proved to achieve competitive results with the Elmo language model. Adding language tokens achieves a better performance than the plain mBert structure while the Residual Cutting trick does not. It implies that the language-specific information is beneficial for the multilingual word representations of the reverse dictionaries task. Adding language tokens has demonstrated to help the Elmo-based multilingual model as well. The most promising multilingual and multitask Elmo-based model structure is shown in Figure 1.

4 Results and Discussion

4.1 Implementation Details

We apply **Bidirectional RNN** and **Elmo** (Peters et al., 2018) models with the same parameter size to find whether bidirectional structure helps. We selected AdamW (Loshchilov and Hutter, 2017) as optimizer. All monolingual models share the same hyper-parameters: the number of layers - 4, the hidden/input size - 256, and the dropout rate - 0.3. WordPiece tokenization was used as the best model design. We follow Devlin et al. (2019) to set the [CLS] token as the first token for monolingual models. We keep the [CLS] token when adding language tokens but set the language token as the first token instead.

Word Representations	SGNS			Char			Electra		
Monolingual Models	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK
RNN+WordPiece	1.000	0.249	0.310	0.158	0.778	0.442	**1.454**	**0.832**	0.433
LSTM+WordPiece	0.990	0.228	0.375	0.148	0.791	0.458	1.491	0.831	0.449
Transformer+WordPiece	1.042	0.214	0.367	0.194	0.780	0.453	1.796	0.827	0.486
BiRNN+WordPiece	0.989	0.221	0.395	0.150	0.791	0.454	1.483	0.832	0.449
Elmo+WordPiece	1.041	0.252	0.282	0.161	0.772	0.430	1.512	0.829	0.434
Elmo+BPE	1.037	0.250	**0.250**	0.162	0.774	0.443	1.537	0.822	0.436
Elmo+ULM	1.022	**0.265**	0.259	0.157	0.781	**0.430**	1.525	0.829	0.432
Elmo+WordPiece+DWA	**0.985**	0.246	0.298	**0.142**	**0.799**	0.447	1.514	0.827	**0.428**

Table 1: Experiment results on English resource test data using the monolingual models. Check section 2 for word algorithm representations' abbreviation. Check section 3 for details of monolingual models.

4.2 Main Results

Reverse dictionary results are evaluated using three metrics: mean squared error (**MSE**) between the reconstructed and reference embeddings, cosine similarity (**COS**) between the reconstructed embedding and the reference embedding, and the cosine-based ranking (**RANK**) between the reconstructed and reference embeddings, measuring the number of other test items having higher cosine with the reconstructed embedding than with the reference embedding(Mickus et al., 2022).

4.2.1 Monolingual Model Performance

We show monolingual models' results in Table 1. As depicted, our proposed model demonstrates competitive if not the best results across the metrics. English, for having the most detailed dictionary data, is selected to present monolingual models' performance[2].

We notice that the transformer-based model has inferior performance on the task. The competition provides a low-resource data set that can explain poorer outcomes for models with high complexity. We tried unidirectional and bidirectional models with similar feature extractors and parameter sizes. The results confirm that bidirectional models perform better and benefit from grasping the context-environment more accurately.

4.2.2 Multilingual Model Performance

We show two ablation experiment results to explain the influence of adding language tags and residual connection removal. **First**, experiment results of the Transformer-based multilingual model on SGNS embedding can suggest the benefits of language tags and curbing residual connection separately or jointly. **Second**, we propose experimental results of the original and adjusted Elmo-based multilingual models. The latter subsumes added language tokens. Such a comparison would clarify whether adding language tokens lead to a general improvement across different languages and word representations.

Electra word representations of Spanish and Italian are not available, implying no related experimental results. The outcomes demonstrate that multilingual models benefit from language-specific information but not from language-agnostic structure. Adding language tags has proved a positive influence on various language models.

4.3 Ablation Study

4.3.1 Tokenizer

We tried three widely-used tokenizers for our proposed model: BPE, ULM, and WordPiece. Both ULM and WordPiece show competitive performance in transformer- and Elmo-based structures. BPE has relatively low performance since the data resource is insufficient and has higher resource requests.

4.3.2 Multitask Model

According to the performance comparison in Table 1, DWA helps the Elmo model achieve better performance and reconstructs three-word representations simultaneously. It demonstrates that differently learned word representations have an internal relation and can be learned together using a shared bottom structure.

[2]check Table 5

Languages	EN			ES			FR			IT			RU		
Multilingual Models	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK
Transformer	1.023	0.201	0.400	0.977	0.300	**0.310**	1.051	0.278	**0.338**	1.143	**0.280**	**0.340**	0.564	0.318	0.363
Transformer+RC	1.029	0.199	0.417	1.005	0.298	0.329	1.069	0.253	0.374	1.189	0.267	0.364	0.601	0.279	0.409
Transformer+ALT	1.043	**0.215**	**0.397**	1.014	**0.308**	0.310	1.103	**0.280**	0.350	1.158	0.276	0.341	0.603	**0.326**	**0.337**
Transformer+RC+ALT	**1.011**	0.159	0.500	**0.955**	0.266	0.422	**1.044**	0.271	0.360	**1.129**	0.264	0.376	**0.561**	0.308	0.371

Table 2: Experiment results on **SGNS** word representation using the multilingual Transformer-based models. Check section 3 for details of multilingual models. **RC** represents the Residual Cutting trick. **ALT** represents the Adding Language Token trick.

Word Representations	SGNS			Char			Electra		
Multilingual Models	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK
Elmo_EN	1.023	0.238	0.317	0.177	0.759	**0.447**	1.555	0.818	**0.440**
Elmo+ALT_EN	**1.014**	**0.246**	**0.300**	**0.164**	**0.762**	0.449	**1.540**	**0.825**	0.441
Elmo_ES	**0.953**	0.342	**0.234**	0.532	0.810	0.405	NA	NA	NA
Elmo+ALT_ES	0.960	**0.351**	0.235	**0.511**	**0.822**	**0.393**	NA	NA	NA
Elmo_IT	**1.094**	0.343	**0.218**	0.355	0.720	0.403	NA	NA	NA
Elmo+ALT_IT	1.106	**0.343**	0.214	**0.354**	**0.735**	**0.387**	NA	NA	NA
Elmo_FR	**1.001**	0.313	0.255	0.388	0.752	**0.411**	1.298	0.845	0.445
Elmo+ALT_FR	1.004	**0.321**	**0.246**	**0.387**	**0.757**	0.411	**1.228**	**0.859**	**0.439**
Elmo_RU	**0.547**	0.357	0.247	0.145	0.816	**0.398**	0.891	**0.729**	0.386
ELmo+ALT_RU	0.563	**0.368**	**0.232**	**0.137**	**0.828**	0.400	**0.887**	0.728	**0.384**

Table 3: Experiment results of the multilingual ELmo-based models. **ALT** represents the Adding Language Token trick.

4.3.3 Difficulty of Reconstructing Different Word Representations

Compared with the Char and Electra, we find that the SGNS is harder to learn from the gloss corpus, suggesting that the contextualized information of words in sentences might be missing from the pure dictionary glosses. Additionally, the result along with (Kaneko and Bollegala, 2021) indicates dictionary corpus can be a promising way to remove the unfair biases rooted in large corpus learned word embeddings.

4.3.4 Difficulty of Learning Different Languages

Languages	Gloss Num	Dict.Size	Avg.Gloss Len	Elmo SGNS COS
English	43608	29042	11.7	0.252
French	43608	40028	14.3	0.333
Italian	43608	40126	13.6	0.352
Spanish	43608	46761	14.8	0.362
Russia	43608	57137	11.3	0.387

Table 4: Language Vocabulary Size Ablation Study. **Dict. Size** means the number of non-repeating tokens shown in the glosses. **Avg. Gloss Len** means the average token numbers contained in a gloss.

Our results of experiments show a strong positive correlation between language's tokens dictionary size and the models' achievable performance Table 4.

There are several possible reasons for the observation. First, as the language model dictionary size decreases, the models' and glosses' ability to explain the slight differences between words, especially the polysemies and synonyms, decreases. Second, a smaller dictionary size indicates that the covered tokens in the language model are a relatively incomplete part of words of the language.

Noted that the second explanation above does not consider the intrinsic differences between languages. The morphologically rich languages, like Russian, tend to have larger vocabulary sizes and bring many unknown words that influence performance negatively (Jurafsky and Martin, 2020).

5 Conclusion

The paper proposes a model showing competitive results in most cases of the reverse dictionaries task. Several conclusions are provided about the reverse dictionaries task by the paper based on the ablation studies. **First**, the transformer-based model, for its high complexity, performs worse compared to RNN- or LSTM-based models. Multilingual transformer-based model benefits from specifying languages and including language-related grammar positional information. **Second**, bidirectional models with similar parameter sizes outperform the unidirectional one since they better grasp the context in low-resource conditions. **Third**, different word representations are potential connections and can be collaboratively learned from glosses using a multitask learning structure. SGNS embedding is much harder to model compared to Character embedding and Electra embedding.

6 Acknowledgements

We express our gratitude to Prof. Shi Wang and Prof. Alexis Palmer for providing computing resources and guidance. We are grateful to the organizers for providing such a fascinating and inspiring competition and promptly resolving all our questions. Special thanks to Rebecca Lee, Xingran Chen, and Natalia Wojarnik for idea sharing and a deep discussion in the initial stage.

References

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

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

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *International conference on machine learning*, pages 2342–2350. PMLR.

Dan Jurafsky and James H Martin. 2020. *Speech and language processing*, 3rd edition draft edition.

Masahiro Kaneko and Danushka Bollegala. 2021. Dictionary-based debiasing of pre-trained word embeddings. *arXiv preprint arXiv:2101.09525*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.

Yoav Levine, Barak Lenz, Or Dagan, Ori Ram, Dan Padnos, Or Sharir, Shai Shalev-Shwartz, Amnon Shashua, and Yoav Shoham. 2019. Sensebert: Driving some sense into bert. *arXiv preprint arXiv:1908.05646*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

Danni Liu, Jan Niehues, James Cross, Francisco Guzmán, and Xian Li. 2020. Improving zero-shot translation by disentangling positional information. *arXiv preprint arXiv:2012.15127*.

Shikun Liu, Edward Johns, and Andrew J Davison. 2019. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

ME Peters, M Neumann, M Iyyer, M Gardner, C Clark, K Lee, and L Zettlemoyer. 2018. Deep contextualized word representations. arxiv 2018. *arXiv preprint arXiv:1802.05365*, 12.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Julien Tissier, Christophe Gravier, and Amaury Habrard. 2017. Dict2vec: Learning word embeddings using lexical dictionaries. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

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

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1515, Austin, Texas. Association for Computational Linguistics.

Da Yin, Tao Meng, and Kai-Wei Chang. 2020. SentiBERT: A transferable transformer-based architecture for compositional sentiment semantics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3695–3706, Online. Association for Computational Linguistics.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

A Appendix: A

Check Table 5 for experiment results of the monolingual models.

B Appendix: B

Check Table 6 for selected multilingual models' performance.

Word Representations	SGNS			Char			Electra		
Monolingual Models	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK
				Language English					
RNN+WordPiece	1.000	0.249	0.310	0.158	0.778	0.442	**1.454**	**0.832**	**0.433**
LSTM+WordPiece	**0.990**	0.228	0.375	**0.148**	**0.791**	0.458	1.491	0.831	0.449
Transformer+WordPiece	1.042	0.214	0.367	0.194	0.780	0.453	1.796	0.827	0.486
BiRNN+WordPiece	0.989	0.221	0.395	0.150	0.791	0.454	1.483	0.832	0.449
Elmo+WordPiece	1.041	**0.252**	**0.282**	0.161	0.772	**0.430**	1.512	0.829	0.434
				Language Spanish					
RNN+WordPiece	0.936	0.358	0.225	0.512	0.822	0.402	NA	NA	NA
LSTM+WordPiece	**0.928**	0.334	0.287	**0.497**	**0.829**	0.418	NA	NA	NA
Transformer+WordPiece	1.011	0.307	0.313	0.577	0.828	0.432	NA	NA	NA
BiRNN+WordPiece	0.939	0.315	0.329	0.511	0.826	0.423	NA	NA	NA
Elmo+WordPiece	0.968	**0.362**	**0.207**	0.520	0.820	**0.396**	NA	NA	NA
				Language French					
RNN+WordPiece	0.975	0.329	0.254	0.379	0.761	0.408	1.272	0.856	0.444
LSTM+WordPiece	**0.971**	0.303	0.329	**0.361**	**0.772**	0.420	**0.191**	0.862	0.457
Transformer+WordPiece	1.057	0.273	0.366	0.461	0.771	0.430	1.523	0.856	0.488
BiRNN+WordPiece	0.984	0.290	0.361	0.366	0.770	0.424	1.202	**0.863**	0.454
Elmo+WordPiece	1.007	**0.333**	**0.239**	0.373	0.763	**0.402**	1.341	0.850	**0.437**
				Language Italian					
RNN+WordPiece	1.078	**0.353**	0.218	0.345	0.741	0.391	NA	NA	NA
LSTM+WordPiece	**1.077**	0.324	0.276	0.340	0.744	0.413	NA	NA	NA
Transformer+WordPiece	1.160	0.256	0.373	0.377	0.731	0.419	NA	NA	NA
BiRNN+WordPiece	1.086	0.309	0.303	**0.338**	**0.747**	0.415	NA	NA	NA
Elmo+WordPiece	1.106	0.352	**0.200**	0.354	0.736	**0.384**	NA	NA	NA
				Language Russian					
RNN+WordPiece	**0.537**	**0.388**	0.226	0.132	0.832	0.391	0.899	0.727	0.372
LSTM+WordPiece	0.547	0.338	0.346	**0.131**	**0.834**	0.401	**0.885**	**0.728**	0.400
Transformer+WordPiece	0.565	0.315	0.377	0.156	0.827	0.411	1.071	0.707	0.473
BiRNN+WordPiece	0.551	0.321	0.397	0.135	0.831	0.403	0.919	0.727	0.410
Elmo+WordPiece	0.557	0.387	**0.217**	0.134	0.831	**0.390**	0.904	0.723	**0.362**

Table 5: **Appendix A**. Experiment results of the monolingual models. Check section 2 for word algorithm representations' abbreviation. Check section 3 for details of monolingual models.

Word Representations	SGNS			Char			Electra		
Monolingual Models	MSE	COS	RANK	MSE	COS	RANK	MSE	COS	RANK
				Language English					
Elmo+WordPiece	1.041	**0.252**	**0.282**	0.161	0.772	**0.430**	**1.512**	**0.829**	0.434
Elmo + WordPiece + DWA	**0.985**	0.246	0.298	**0.142**	**0.799**	0.447	1.514	0.827	**0.428**
				Language French					
Elmo+WordPiece	1.007	**0.333**	**0.239**	0.373	0.763	**0.402**	1.341	0.850	0.437
Elmo + WordPiece + DWA	**0.937**	0.327	0.243	**0.364**	**0.770**	0.406	**1.315**	**0.854**	**0.428**
				Language Russian					
Elmo+WordPiece	0.557	0.387	0.217	0.134	0.831	**0.390**	0.904	0.7226	**0.362**
Elmo + WordPiece + DWA	**0.534**	**0.388**	**0.189**	**0.127**	**0.838**	0.376	**0.908**	**0.7235**	0.364

Table 6: **Appendix B**. The table shows the selected multilingual models' performance. Check section 2 for word algorithm representations' abbreviation. Check section 3 for details of monolingual models.

BLCU-ICALL at SemEval-2022 Task 1: Cross-Attention Multitasking Framework for Definition Modeling

Cunliang Kong[1], **Yujie Wang**[2], **Ruining Chong**[1], **Liner Yang**[1]*,
Hengyuan Zhang[1], **Erhong Yang**[1], **Yaping Huang**[2]
[1]School of Information Science, Beijing Language and Culture University
[2]School of Computer and Information Technology, Beijing Jiaotong University
cunliang.kong@outlook.com

Abstract

This paper describes the BLCU-ICALL system used in the SemEval-2022 Task 1 Comparing Dictionaries and Word Embeddings, the Definition Modeling subtrack, achieving 1st on Italian, 2nd on Spanish and Russian, and 3rd on English and French. We propose a transformer-based multitasking framework to explore the task. The framework integrates multiple embedding architectures through the cross-attention mechanism, and captures the structure of glosses through a masking language model objective. Additionally, we also investigate a simple but effective model ensembling strategy to further improve the robustness. The evaluation results show the effectiveness of our solution. We release our code at: https://github.com/ blcuicall/SemEval2022-Task1-DM.

1 Introduction

Word embeddings (Mikolov et al., 2013a; Pennington et al., 2014; Yogatama et al., 2015) are dense and low dimensional vectors used in many NLP tasks because they are found to be useful representations of words and often lead to better performance in various tasks. In recent years, large pretrained language models (PLMs), such as BERT (Devlin et al., 2019) and GPT (Petroni et al., 2019) families of models, have taken the NLP field by storm, achieving state-of-the-art performance on many tasks (Min et al., 2021). The contextual embeddings generated by PLMs are proven to capture syntax and semantic features of words (Jawahar et al., 2019; Turton et al., 2020). But for human beings, word embeddings containing these information is still a *black box* and unexplainable.

There have been many efforts devoted to evaluating the word embeddings' lexical information, such as the word similarity (Landauer and Dumais, 1997; Downey et al., 2007) and analogical relation (Mikolov et al., 2013c) tasks. However, these tasks can only serve as indirect evaluation methods. In light of this, Noraset et al. (2017) proposed the task of definition modeling to evaluate whether a word embedding can be employed to generate a dictionary gloss. Since the gloss is a direct and explicit statement of word meaning, this task provides a more transparent view.

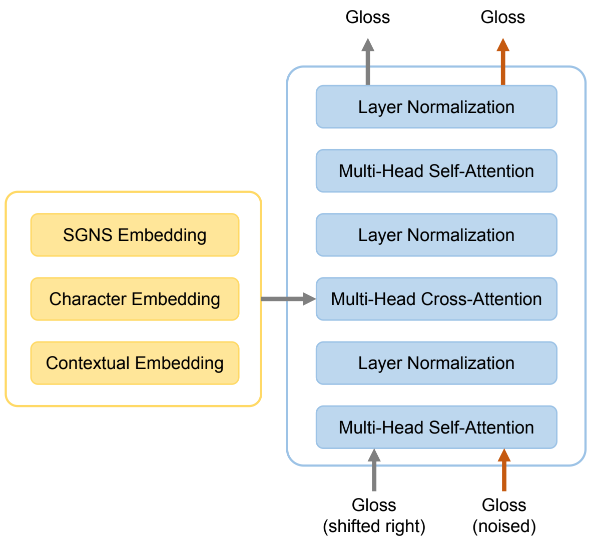


Figure 1: Architecture of the Cross-Attention Multitasking Framework.

The SemEval-2022 Task 1 Comparing Dictionaries and Word Embeddings (Mickus et al., 2022) aims at comparing the two types of semantic descriptions: dictionary glosses and word embeddings. The subtrack 1 is a definition modeling task, which requires models to generate glosses from word embeddings. The task provides data from 5 languages (English, Spanish, French, Italian, Russian) as well as static, character, and contextual embeddings.

Our team propose a transformer-based (Vaswani et al., 2017) Cross-Attention Multitasking Framework to explore the task and apply the framework to all 5 languages. We integrate the multiple embed-

*Corresponding author: Liner Yang.

	Train	Dev.	Test	SGNS Emb.	Character Emb.	Electra Emb.	Gloss Len.
English	43,608	6,375	6,221	✓	✓	✓	11.73
Spanish	43,608	6,375	6,221	✓	✓	✗	14.84
French	43,608	6,375	6,221	✓	✓	✓	14.31
Italian	43,608	6,375	6,221	✓	✓	✗	13.58
Russian	43,608	6,375	6,221	✓	✓	✓	11.32

Table 1: Detailed statistics of the dataset. The last column lists the average length of glosses in the training set.

ding architectures through a cross-attention mechanism, which allows the model to query all the embeddings at each time step during generation. To better capture the structure of glosses, we employ an additional masking language model (MLM) (Devlin et al., 2019) into the framework. We also investigate the ensemble strategies to further enhance the robustness.

Therefore, the contributions of our system lie in:

- We propose the Cross-Attention Multitasking Framework as a novel solution to the definition modeling task.
- The evaluation results show the effectiveness of our solution. Our system achieves 1st on Italian, 2nd on Spanish and Russian, and 3rd on English and French.

2 Background

The definition modeling subtrack provides participants with a multilingual dataset in the form of $\{E, \boldsymbol{g}\}$, where E is a set including SGNS (Mikolov et al., 2013b), character (Kim et al., 2016), and Electra (Clark et al., 2020) embeddings, and $\boldsymbol{g}$ is a dictionary gloss. This task takes E as the input, and requires models to generate $\boldsymbol{g}$. Note that all the embeddings have 256 dimensions, and the Electra embeddings are only available for 3 of the 5 languages. More detailed statistics of the dataset are listed in Table 1.

Many previous work used additional data to improve the performance of generation, such as example sentences (Gadetsky et al., 2018; Chang et al., 2018; Ishiwatari et al., 2019; Kong et al., 2020) and semantic features (Yang et al., 2020). Some studies also investigated how to employ PLMs for this task (Reid et al., 2020; Bevilacqua et al., 2020; Huang et al., 2021; Kong et al., 2022).

Differently, to keep the results linguistically significant and easily comparable, the SemEval-2022 Task 1 prohibits the usage of external data and PLMs. Therefore, our system focuses on effectively integrating all given embeddings and modeling the glosses.

3 System Overview

Figure 1 illustrates the entire architecture of our system, which is a Cross-Attention Multitasking Framework based on transformer. The framework consists of two objectives, namely the generation and reconstruction objectives. This section introduces the system in detail.

3.1 The Generation Objective

The generation objective serves as a standard transformer decoder, which generates the gloss as the following language model:

$$P(\boldsymbol{g}|E;\boldsymbol{\theta}) = \prod_t P\left(\boldsymbol{g}_t|\boldsymbol{g}_{<t}, E;\boldsymbol{\theta}\right), \quad (1)$$

where $\boldsymbol{g}_t$ is the t-th token in the gloss, and $\boldsymbol{\theta}$ is the set of parameters. The model is then optimized using the following loss function:

$$\mathcal{L}_{gen}(\boldsymbol{\theta}) = -\sum_{\boldsymbol{g}\in D} \log P(\boldsymbol{g}|E;\boldsymbol{\theta}), \quad (2)$$

where D is the training dataset.

In the above operations, a crucial challenge is to integrate multiple embeddings corresponding to one word. We assume that the SGNS, character, and Electra embeddings contain different lexical features, and better results can be obtained by comprehensively considering all the information. To achieve that, we feed the set E, including all these embeddings, into the cross-attention mechanism:

$$\text{Cross-Attn}(H, E, E) = \text{softmax}(\frac{HE^T}{\sqrt{d_h}})E \quad (3)$$

where H is the hidden-states obtained from by self-attention, and d_h is the dimension of the hidden-states. This operation ensures the given embeddings are adaptively integrated at each time-step.

3.2 The Reconstruction Objective

Our system is a language model specially designed for dictionary glosses. We further enhance this model by incorporating a reconstruction objective.

We corrupt each gloss $\boldsymbol{g}$ by randomly substituting or blanking some words. And then we obtain a corrupted version $\tilde{\boldsymbol{g}}$. We input $\tilde{g}$ into our system and obtain $\boldsymbol{g}$ by solving a self-supervised task of:

$$P(\boldsymbol{g}|\tilde{\boldsymbol{g}};\boldsymbol{\theta}) = \prod_t P(\boldsymbol{g}_t|\boldsymbol{g}_{<t},\tilde{\boldsymbol{g}};\boldsymbol{\theta}). \tag{4}$$

Note that we share exactly the same parameters $\boldsymbol{\theta}$ as in the generation objective. The model is optimized by the following loss function:

$$\mathcal{L}_{rec}(\boldsymbol{\theta}) = -\sum_{\boldsymbol{g}\in D} \log P(\boldsymbol{g}|\tilde{\boldsymbol{g}};\boldsymbol{\theta}), \tag{5}$$

The goal of the reconstruction objective is to better model the glosses. Therefore, we don't use the given embeddings in this operation. In practice, we feed a zero vector into the cross-attention mechanism to mask it out as $\text{Cross-Attn}(H,\mathbf{0},\mathbf{0})$.

3.3 Training and Ensembling

We train the entire multitasking framework by jointly minimizing the weighted sum of both loss functions:

$$\mathcal{L} = \mathcal{L}_{gen} + \lambda\mathcal{L}_{rec}, \tag{6}$$

where λ is a hyper-parameter.

Model ensembling is proven to be effective to improve the robustness (Allen-Zhu and Li, 2020). In our work, we adopt a simple but effective model ensembling strategy. We train a series of models initialized by different random seeds, and then vote with the trained models during inference.

4 Experimental Setup

4.1 Implementation Details

Many neural network-based generation systems struggle with the OOV (out-of-vocabulary) problem. To alleviate the problem, we apply the SentencePiece algorithm (Kudo and Richardson, 2018) to glosses to reduce the vocabulary size. We use the tokenizers[1] toolkit for implementation and set the size to 10k for all 5 languages.

Our system is a 3-layer, 8-head transformer-based model implemented by the Pytorch library (Paszke et al., 2019). We use the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. We adopt the Noam Optimizer proposed by (Vaswani et al., 2017) with an initial learning rate of $1e-7$, a maximum learning rate of $1e-3$, and a minimum learning rate of $1e-9$. We set the warmup steps to 4000 and batch size to 128. The maximum epochs is set to 500. And we set an early stop strategy in the patience of 5 epochs. To avoid gradient exploding, we clipped the gradient norm within 0.1. We also employ label smoothing technique (Pereyra et al., 2017) with a smoothing value of 0.1 during training. For the gloss corruption in the reconstruction objective, we follow Devlin et al. (2019) to randomly delete and blank words with a uniform probability of 0.2. And the λ (in Equation 6) is set to 1. For model ensembling, we train 5 models with different seeds. Due to the time constraints, our official submission has a result of ensembling three models on English, and results of single models on the reset of 4 languages. We submitted the results of ensembling 5 models in the post-evaluation phase.

For each language, we use the development set released by organizers for model selection. We select the best epoch using the summary of BLEU (Papineni et al., 2002) and MoverScore (Zhao et al., 2019) on the development set.

4.2 Evaluation Metrics

The definition modeling subtrack uses three metrics, which are MoverScore (Zhao et al., 2019), BLEU (Papineni et al., 2002) , and lemma-level BLEU respectively. Readers can refer to the task paper (Mickus et al., 2022) for more details.

5 Results and Analysis

In this section, we present the evaluation results and discuss our analysis of the generated definitions.

5.1 Main Results

Table 2 presents the evaluation scores on all 5 languages. Results show that our system significantly outperforms the baseline models in terms of the sentence BLEU and lemma-level BLEU. This indicates the effectiveness of our proposed cross-attention multitasking framework. However, the SGNS and Char are strong baselines in terms of the MoverScore, and our system only outperforms the baselines on English. We speculate that our results have more coincide words with references, but are not fluent enough, which leads to a low score from

[1]tokenizers: https://github.com/huggingface/tokenizers.

	Models	S-BLEU	L-BLEU	MvSc.
EN	SGNS	0.00125	0.00250	0.10339
	Char	0.00011	0.00022	0.08852
	Electra	0.00165	0.00215	0.08798
	CAMF	**0.03127**	**0.03957**	**0.13475**
	Ensemble	0.03106	0.03906	0.13273
ES	SGNS	0.01536	0.02667	**0.20130**
	Char	0.01505	0.02471	0.19933
	CAMF	0.03914	0.05606	0.12778
	Ensemble	**0.03925**	**0.05624**	0.13121
FR	SGNS	0.00351	0.00604	0.18478
	Char	0.00280	0.00706	**0.18579**
	Electra	0.00219	0.00301	0.17391
	CAMF	0.02679	0.03691	0.04193
	Ensemble	**0.02700**	**0.03738**	0.04455
IT	SGNS	0.02591	0.04081	**0.20527**
	Char	0.00640	0.00919	0.15920
	CAMF	0.06646	0.09926	0.11717
	Ensemble	**0.06812**	**0.10147**	0.12233
RU	SGNS	0.01520	0.02112	**0.34716**
	Char	0.01313	0.01847	0.32307
	Electra	0.01189	0.01457	0.33577
	CAMF	0.04843	0.06548	0.14820
	Ensemble	**0.05192**	**0.07074**	0.15702

Table 2: Evaluation results of different models in 5 languages. The SGNS, Char, Electra are baseline models provided by the organizers. The CAMF (Cross-Attention Multilingual Framework) is the model of official submission. And the Ensemble is an ensemble of 5 models submitted in the post-evaluation. Bold and underline mark the best and second scores, respectively.

(1) Redundancy and overusing common phrases: 42.00%	
word	explosion
reference	A sudden outburst.
hypothesis	A sudden, sudden, or destruction.
(2) Self-reference: 2.00%	
word	discover
reference	To reveal (information); to divulge, make known.
hypothesis	To make a conclusion of; to discover.
(3) Wrong Part-Of-Speech: 5.50%	
word	genius
reference	ingenious, brilliant, very clever, or original.
hypothesis	A person or thing that is extraordinary.
(4) Under-specified: 23.50%	
word	mayor
reference	The leader of a city.
hypothesis	A person who is a member of authority.
(5) Opposite: 2.00%	
word	solid
reference	Excellent , of high quality , or reliable.
hypothesis	Having no size or value.
(6) Close Semantics: 17.00%	
word	bed
reference	The time for going to sleep or resting in bed.
hypothesis	The state or quality of being a room.
(7) Incorrect: 52.00%	
word	smooth
reference	Lacking projections or indentations; not serrated.
hypothesis	Having the shape of a tree.

Table 3: Error types and examples.

the pretrained model used by MoverScore.

We also observe that model ensembling has brought the improvement of performance. It can be seen from the table that the Ensemble model outperforms the CAMF on 4 of the 5 languages, except for a slight decline on English. This may be due to the randomness of the parameter initialization. We also argue that better performance can be obtained by applying hyper-parameter searching algorithms and ensembling more models.

5.2 Error Analysis

In order to qualitatively analyze the definitions generated by our system, we randomly select several items from the English test set and manually annotate the error types following Noraset et al. (2017). In total, we extract 200 items, of which 197 contain some degree of error. We illustrate the error types and examples in Table 3. Note that each item may contain multiple errors, so the sum of the percentages in the table is greater than 100%.

From the table, we observe that the quality of English definitions generated by our system still need to be improved. Error types (1) to (3) are problems from the system, and types (4) to (6) are shortcomings in the embeddings. As we can see, the former accounts for a much larger proportion than the latter. The 52% incorrectness indicated by type (7) shows that many glosses generated by our system are irrelevant to the word. And the dataset released in this task will support significant future work on the definition modeling task.

6 Conclusion

In this paper, we present the implementation of the BLCU-ICALL system submitted to the SemEval-2022 Task 1, Definition Modeling subtrack. We propose a Cross-Attention Multitasking Framework that leverages multiple embedding architectures and jointly trains two objectives. We also investigate a simple but effective ensembling strategy to enhance the robustness. In future efforts, we plan to further improve our system to better handle the problems of redundancy and incorrect glosses.

Acknowledgements

This work was supported by the funds of Beijing Advanced Innovation Center for Language Resources (No. TYZ19005), Research Project of the National Language Commission (No. ZDI135-105, No. ZDI135-131), and BLCU supported project for young researchers program (supported by the Fundamental Research Funds for the Central Universities)(No. 20YCX142). We would like to thank all anonymous reviewers for their valuable comments and suggestions on this work.

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221, Online. Association for Computational Linguistics.

Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. 2018. xsense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *CoRR*, abs/1809.03348.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

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

Doug Downey, Stefan Schoenmackers, and Oren Etzioni. 2007. Sparse information extraction: Unsupervised language models to the rescue. In *ACL*.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 266–271.

Han Huang, Tomoyuki Kajiwara, and Yuki Arase. 2021. Definition modelling for appropriate specificity. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2499–2509. Association for Computational Linguistics.

Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa. 2019. Learning to describe unknown phrases with local and global contexts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3467–3476. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI conference on artificial intelligence*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Cunliang Kong, Yun Chen, Hengyuan Zhang, Liner Yang, and Erhong Yang. 2022. Multitasking framework for unsupervised simple definition generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Cunliang Kong, Liner Yang, Tianzuo Zhang, Qinan Fan, Zhenghao Liu, Yun Chen, and Erhong Yang. 2020. Toward cross-lingual definition generation for language learners. *arXiv preprint arXiv:2010.05533*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Thomas K Landauer and Susan T Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. 2021. Recent advances in natural language processing via large pre-trained language models: A survey. *arXiv preprint arXiv:2111.01243*.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Thirty-First AAAI Conference on Artificial Intelligence*.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *CoRR*, abs/1701.06548.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Machel Reid, Edison Marrese-Taylor, and Yutaka Matsuo. 2020. VCDM: Leveraging Variational Bi-encoding and Deep Contextualized Word Representations for Improved Definition Modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6331–6344.

Jacob Turton, David Vinson, and Robert Elliott Smith. 2020. Deriving contextualised semantic features from bert (and other transformer model) embeddings. *arXiv preprint arXiv:2012.15353*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Liner Yang, Cunliang Kong, Yun Chen, Yang Liu, Qinan Fan, and Erhong Yang. 2020. Incorporating sememes into chinese definition modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1669–1677.

Dani Yogatama, Manaal Faruqui, Chris Dyer, and Noah Smith. 2015. Learning word representations with hierarchical sparse coding. In *International Conference on Machine Learning*, pages 87–96. PMLR.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *CoRR*, abs/1909.02622.

LingJing at SemEval-2022 Task 1: Multi-task Self-supervised Pre-training for Multilingual Reverse Dictionary

Bin Li[1]*, Yixuan Weng[2]*, Fei Xia[2,3]*, Bin Sun[1], Shutao Li[1], Shizhu He[2,3]
[1] College of Electrical and Information Engineering, Hunan University
[2] National Laboratory of Pattern Recognition, Institute of Automation, CAS
[3] School of Artificial Intelligence, University of Chinese Academy of Sciences
{libincn, shutao_li, sunbin611}@hnu.edu.cn, wengsyx@gmail.com,
xiafei2020@ia.ac.cn, shizhu.he@nlpr.ia.ac.cn

Abstract

This paper introduces the result of Team LingJing's experiments in SemEval-2022 Task 1 Comparing Dictionaries and Word Embeddings (CODWOE)[1]. This task aims at comparing two types of semantic descriptions, including the definition modeling and reverse dictionary track. Our team focuses on the reverse dictionary track and adopts the multi-task self-supervised pre-training for multilingual reverse dictionaries. Specifically, the randomly initialized mDeBERTa-base model is used to perform multi-task pre-training on the multilingual training datasets. The pre-training step is divided into two stages, namely the MLM pre-training stage and the contrastive pre-training stage. As a result, all the experiments are performed on the pre-trained language model during fine-tuning. The experimental results show that the proposed method has achieved good performance in the reverse dictionary track, where we rank the 1-st in the Sgns targets of the EN and RU languages. All the experimental codes are open-sourced at `https://github.com/WENGSYX/Semeval`.

1 Introduction

The CODWOE shared task invites the participants to compare two types of semantic descriptions: dictionary glosses and word embedding representations. The intuitions come from the questions: "Are these two types of representation equivalent? Can we generate one from the other?". To study this question, the CODWOE proposes two sub-tracks: a definition modeling track (Noraset et al., 2017), where participants have to generate glosses from vectors, and a reverse dictionary track (Hill et al., 2016), where participants have to generate vectors from glosses. These two tracks are fairly challenging (Hill et al., 2016), where more efficient methods are required to be designed for implementation.

*These authors contribute equally to this work.
[1]https://codwoe.atilf.fr/

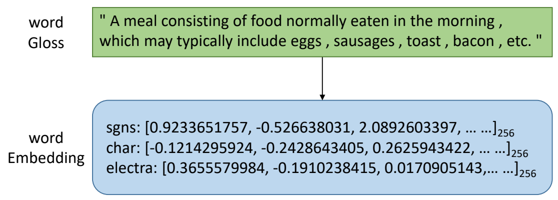

word
Gloss
" A meal consisting of food normally eaten in the morning , which may typically include eggs , sausages , toast , bacon , etc. "
word
Embedding
sgns: [0.9233651757, -0.526638031, 2.0892603397, … …]$_{256}$
char: [-0.1214295924, -0.2428643405, 0.2625943422, … …]$_{256}$
electra: [0.3655579984, -0.1910238415, 0.0170905143,… …]$_{256}$

Figure 1: An example of the reverse dictionary task. Given the word gloss, it is required to generate the vectors of their corresponding Sgns, Char, and Electra, respectively.

These tasks are also useful for explainable AI, since they involve converting human-readable data into machine-readable data and back (Li et al., 2021). In this paper, we focus on the reverse dictionary track. As shown in Figure 1, given the gloss "A meal consisting of food normally eaten in the morning, which may typically include eggs, sausages, toast, bacon, etc.", the reverse dictionary task requires us to generate corresponding three sets of 256-dimensional word vectors. The Sgns (Mikolov et al., 2013) , char (Vakulenko et al., 2017), and Electra (González et al., 2020) are skip-gram with negative sampling embeddings, character-based embeddings, and Transformer-based contextualized embeddings, respectively.

It is noted that this task comprises datasets in 5 languages: English, Spanish, Italian, French, and Russian. The reverse dictionary task is difficult due to the significant inborn differences between word vectors and glosses and the vast differences between languages (Bosc and Vincent, 2018).

To solve the above problems, we use a multi-task self-supervised pre-training approach with Masked language modeling (MLM) (Taylor, 1953; Devlin et al., 2019) and contrastive learning (Reimers and Gurevych, 2019; Su et al., 2021). On the one hand, MLM can better capture the semantic representation of the input text (Liu et al., 2019). On the other hand, contrast learning can further improve the performance of downstream regression tasks (Jaiswal et al., 2020). Specifically, we use a randomly initialized mDeBERTa-base (He et al., 2021) model to perform MLM pre-training on five text datasets in different languages. Contrastive pre-training (Gao et al., 2021) is then performed using vectors with and without dropout (Srivastava et al., 2014). Afterward, the model is fine-tuned using the Reverse Dictionary dataset. The experimental results show that the proposed method has achieved good performance in the reverse dictionary track. We achieve the top three results on the Sgns evaluation metrics in all languages. Specifically, we get first place in English and Russian, second place in Spanish and French, and third place in Italian.

2 Main method

In this section, we will elaborate on the main methods for the reverse dictionary track. As the pre-training method can enhance the performance of semantic representation (Qiu et al., 2020), we adopt masked language modeling (MLM) task (Devlin et al., 2019) and contrastive pre-training task (Jaiswal et al., 2020) for implementing this regression task.

2.1 Masked language modeling task

Masked language modeling (MLM) task consists of giving the model a random masked sentence and optimizing the weights inside the model to output the unmasked sentence on the other side. We implement the MLM pre-training method with the same original setting as BERT (Devlin et al., 2019). What's more, we adopt the standard implementations of the MLM from the website[2].

2.2 Contrastive pre-training task

Our method follows the SimCSE (Gao et al., 2021) method, where the self-supervised model is adopted for the contrastive pre-training task. For the self-supervised part, we use dropout to add noise to the text twice, thus constructing a pair of positive samples, and pairs of negative samples are sentences processed with the dropout in the batch. The above processes can be formulated as the equation (1)

$$\mathcal{L}_{\text{CL}} = -\log \frac{\exp\left(\text{sim}\left(\tilde{h}_i, h_i\right)/\tau\right)}{\sum_{j=1}^{n} \exp\left(\text{sim}\left(\tilde{h}_i, h_j\right)/\tau\right)}, \quad (1)$$

where the h_i represents the hidden feature of the positive sample, while the h_j is the hidden feature of the negative drop-out sample. The τ is a temperature hyper-parameter and sim(,) means the cosine similarity function.

[2]https://github.com/lucidrains/mlm-pytorch

2.3 Multi-task pre-training

Multi-task learning is known to fully enhance the performance of the single task with multiple related tasks to be designed and optimized (Sanh et al., 2021). We combine the above two pre-training task to the multi-task objectives, where the final loss function can be represented as follows

$$\mathcal{L} = \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{CL}}. \quad (2)$$

2.4 Downstream fine-tuning

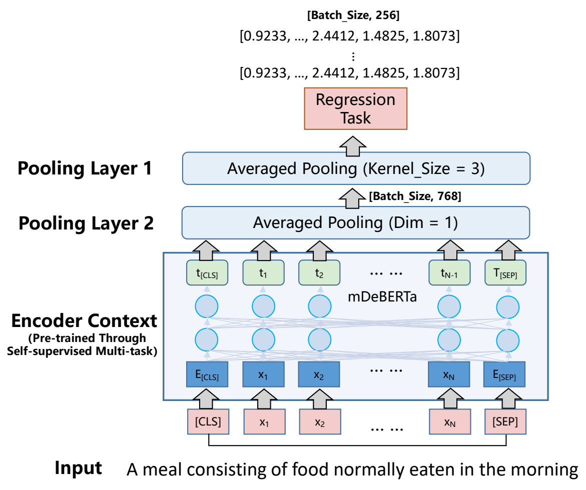


Figure 2: Main structure of the proposed method.

Concretely, given the input sentence, the semantic representation can be obtained through the context encoder with pre-training. As shown in the Figure 2, we use the pre-trained language model through self-supervised multi-task pre-training as the backbone for the regression task. Once obtaining the final representation of the pre-trained language model, two pooling layers (Lin et al., 2013) are designed to get the useful features with the probable size. The mean pooling layer is added on top of the pre-trained model for squeezing the features. Another pooling layer (with the kernel_size=3) is added before the final regression task.

3 Experimental setup

3.1 Data Description

The CODWOE shared task provides datasets in five different languages (EN, ES, FR, IT, RU). For these datasets of five languages, each dataset has 43,608 training sets, 6375 dev sets and 4208 test sets. Each language contains multiple embeddings containing "Char" and "Sgns", while English, French and Russian have the embedding "Electra". We will introduce these datasets as follows.

Char corresponds to character-based embeddings, computed with an auto-encoder on the spelling of a word. In addition, the "gloss" key in each dataset is the source in the reverse dictionary track. We need to use "gloss" to generate the associated embeddings.

Sgns corresponds to skip-gram with negative sampling embeddings (aka. word2vec (Mikolov et al., 2013)).

Electra corresponds to the Transformer-based (Vaswani et al., 2017) contextualized embeddings.

Moreover, the organizers want the shared task to be as linguistically relevant as possible and hope to provide a fair competition environment for all participants. The organizer forbids the use of external resources and pre-trained language models in CODWOE.

3.2 Evaluation metrics

In this task, the performance of the system is evaluated through three evaluation indicators (Mickus et al., 2022).

Mean squared error (MSE) between the submission's reconstructed embedding and the reference embedding.

Cosine similarity (Cossim) between the submission's reconstructed embedding and the reference embedding.

$$\text{MSE} = \frac{1}{n}\Sigma_{i=1}^{n}\left(\frac{A_i - B_i}{\sigma_i}\right)^2$$

$$\text{Cossim} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

where the A and B refer to two matrices that need to be calculated.

Cosine-based ranking[3] between the submission's reconstructed embedding and the reference embedding; i.e., how many other test items have a cosine similarity with the reconstructed embedding higher than that with the reference embedding.

[3]Specific implementations can refer to https://github.com/WENGSYX/Semeval.

3.3 Method introduction

The Baseline provided by the organizer[4] uses the encoder structure of the Transformer (Vaswani et al., 2017; Wolf et al., 2020) framework. After each token passes through the embedding layer, positional encoding will be added to indicate the location structure of the token. Then it will be input to the encoder based on the transformer and finally output to the linear layer to make the dimension of the matrix consistent with the label.

In addition, the organizer has made some improvements to the baseline.

1. The principled way of selecting hyper-parameters (using Bayesian Optimization (Snoek et al., 2012; Frazier, 2018)).

2. A sentence-piece re-tokenization, to ensure the vocabulary is of the same size for all languages.

3. The beam-search (Wiseman and Rush, 2016; Freitag and Al-Onaizan, 2017) decoding for the definition modeling pipeline.

Our method uses the randomly initialized mDeBERTa (He et al., 2021) model. The mDeBERTa improves the BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) models using disentangled attention and enhanced mask decoder. It shares the base model with 12 layers and 768 hidden size, which is pre-trained on the multilingual corpus. It has 86M backbone parameters with a vocabulary containing 250K tokens which introduce 190M parameters in the Embedding layer. It supports most languages around the world, since it is believed that there should be some shared semantic features between different languages[5].

3.4 Implementation details

We use the hugging-face[6] (Wolf et al., 2020) framework and train the model based on the Pytorch (Paszke et al., 2019). During training, we employ the AdamW optimizer (Loshchilov and Hutter, 2017). The default learning rate is set to 1e-5 with the warm-up (He et al., 2016). Four 3090 GPUs are used for all experiments.

[4]Specific implementation can refer to https://github.com/TimotheeMickus/codwoe/tree/main/baseline_archs

[5]Please refer: https://ai.glossika.com/blog/a-map-to-the-syntax-of-all-spoken-languages

[6]https://github.com/huggingface/transformers

Experimental Items	Baseline			Ours		
Language	MSE	Cosine	Ranking	MSE	Cosine	Ranking
English	0.91092	0.15132	0.49030	0.86239	0.24310	0.32907
Espana	0.92996	0.20406	0.49912	0.85770	0.35275	0.25101
French	1.14050	0.19774	0.49052	1.02968	0.32799	0.28213
Italian	1.12536	0.20430	0.47692	1.03945	0.35955	0.22995
Russian	0.57683	0.25316	0.49008	0.52827	0.42440	0.18711

Table 1: Results of the Sgns track.

Experimental Items	Baseline			Ours		
Language	MSE	Cosine	Ranking	MSE	Cosine	Ranking
English	0.14776	0.79006	0.50218	0.47103	0.00331	0.48599
Espana	0.56952	0.80634	0.49778	0.50121	0.85770	0.35275
French	0.39480	0.75852	0.49945	0.96678	0.00809	0.51862
Italian	0.36309	0.72732	0.49663	0.88129	-0.02992	0.49603
Russian	0.13498	0.82624	0.49451	0.47905	0.00479	0.47228

Table 2: Results of the Char track.

On the MLM pre-training task, we alternately carry out the pre-training tasks of long text and short text. After mixing the data sets of five different languages, we train them for 40 epochs. In detail, we classify all data sets with a text length of 30. In each epoch, firstly, samples with text length less than or equal to 30 are trained with a maximum length of 32 tokens (including <CLS> and <SEP>) and the batch size is set to 70. Then we change the maximum length to 160 tokens and set the batch size to 18 for training the remaining samples.

Referring to the settings of WWM (Cui et al., 2021; Joshi et al., 2020), we use the text mask rate with a probability of 20%, and adopt that the 1, 2, 3, 4 n-gram masking length with a probability of 85%, 5%, 5%, and 5%.

In contrastive pre-training, we repeatedly integrate a sample into the model twice. During this period, because our model has dropout, it will add noise to the input, so that the output of the two times is distinct. As a result, our method can be improved in the sentence representation ability through self-supervised.

Based on the pre-trained language model, we fine-tune with the maximum length of all samples to 100 tokens, the batch size to 50 (there will be 2 * 50 samples for each step to be calculated by the model at the same time). The number of training epochs is 40.

4 Results and discussions

In this section, we introduce the experimental results of the Sgns, the Char and the Electra tracks. The online results and further discussions are also presented.

4.1 Experimental results

The experimental results of the Sgns, Char and Electra can be found in the Table 1, 2 and 3. Specifically, for the Sgns track, we outperform the experiments of each baseline according to all the metrics. The reason may be that the pre-training method with MLM and contrastive learning can well provide well-formed vector space representations between samples. As for the Char and Electra track, the baseline is better than ours. It may be because the word and contextual character features are hard to be captured due to the smaller corpus. In the future, we will explore more efficient methods to perform well definition modeling in these tracks.

4.2 Official online results

As shown in Table 4, we achieve the top three results on the Sgns evaluation metrics in all languages. Specifically, we get first place in English and Russian, second place in Spanish and French, and third place in Italian. Our method is effective on the Electra evaluation metrics, but not the best. Our team ranks the second place, fourth and fourth place in Russian, English, and French, respectively. Our approach does not achieve good results on the char metric, which represents the character level. This result may be that it is difficult for the model to capture semantics while maintaining high precision letter-level fine-grained word vector learning.

Experimental Items	Baseline			Ours		
Language	MSE	Cosine	Ranking	MSE	Cosine	Ranking
English	1.41287	0.84283	0.49849	1.50876	0.84592	0.47773
French	1.15348	0.85629	0.49784	1.27066	0.85859	0.47762
Russian	0.87358	0.72086	0.49120	0.82773	0.73397	0.42020

Table 3: Results of the Electra track.

Online	Sgns					Char					Electra		
TEAM	EN	ES	FR	IT	RU	EN	ES	FR	IT	RU	EN	FR	RU
LingJing(ours)	1	2	2	3	1	7	5	5	6	5	4	4	2
pzchen	2	4	3	2	3	3	1	1	1	1	1	1	1
IRB-NLP	3	1	1	1	2	4	3	4	2	2	5	3	3
Locchi	4	/	/	4	/	1	/	/	4	/	3	/	/
Nihed_Bendahman_	5	5	4	6	4	2	2	2	3	4	2	2	4
zhwa3087	6	6	5	5	5	6	4	3	5	3	/	/	/
theOne	7	/	/	/	/	5	/	/	/	/	6	/	/
tthhanh	8	7	6	7	6	/	/	/	/	/	/	/	/

Table 4: Results of the online official Rank.

5 Conclusion

In this paper, it is mainly introduced that in order to solve the reverse dictionary track in Semeval-22 CODWOE, the LingJing team makes the model have the ability of semantic understanding through the MLM task with contrastive learning in the randomly initialized mDeBERTa model. After that, we report the performance of our model in CODWOE, and obtain the best performance in English and Russian tasks of Sgns dataset, which proves that our method is effective. In the future, we will further study how to make full use of the characteristics of different languages and make the model embed the text into a more accurate vector space.

Acknowledgement

This work is supported by the National Key R&D Program of China (2018YFB1305200), the National Natural Science Fund of China (62171183).

References

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1532, Brussels, Belgium. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert.

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

Peter I. Frazier. 2018. A tutorial on bayesian optimization.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. *CoRR*, abs/1702.01806.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

José Ángel González, Lluís-F Hurtado, and Ferran Pla. 2020. Transformer based contextualization of pre-trained word embeddings for irony detection in twitter. *Information Processing & Management*, 57(4):102262.

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

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2020. A survey on contrastive self-supervised learning. *CoRR*, abs/2011.00362.

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

Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized transformer for explainable recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4947–4957, Online. Association for Computational Linguistics.

M. Lin, Q. Chen, and S. Yan. 2013. Network in network. *Computer Science*.

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

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

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3259–3266. AAAI Press.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *CoRR*, abs/2003.08271.

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

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'12, page 2951–2959, Red Hook, NY, USA. Curran Associates Inc.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *CoRR*, abs/2103.15316.

Wilson L. Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Svitlana Vakulenko, Lyndon Nixon, and Mihai Lupu. 2017. Character-based neural embeddings for tweet clustering. *SocialNLP 2017*, page 36.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. *CoRR*, abs/1606.02960.

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

IRB-NLP at SemEval-2022 Task 1: Exploring the Relationship Between Words and Their Semantic Representations

Damir Korenčić[*]
Division of Electronics
Ruđer Bošković Institute
Zagreb, Croatia
damir.korencic@irb.hr

Ivan Grubišić[*]
Division of Electronics
Ruđer Bošković Institute
Zagreb, Croatia
ivan.grubisic@irb.hr

Abstract

What is the relation between a word and its description, or a word and its embedding? Both descriptions and embeddings are semantic representations of words. But, what information from the original word remains in these representations? Or more importantly, which information about a word do these two representations share? Definition Modeling and Reverse Dictionary are two opposite learning tasks that address these questions. The goal of the Definition Modeling task is to investigate the power of information laying inside a word embedding to express the meaning of the word in a humanly understandable way – as a dictionary definition. Conversely, the Reverse Dictionary task explores the ability to predict word embeddings directly from its definition. In this paper, by tackling these two tasks, we are exploring the relationship between words and their semantic representations. We present our findings based on the descriptive, exploratory, and predictive data analysis conducted on the CODWOE dataset. We give a detailed overview of the systems that we designed for Definition Modeling and Reverse Dictionary tasks, and that achieved top scores on SemEval-2022 CODWOE challenge in several subtasks. We hope that our experimental results concerning the predictive models and the data analyses we provide will prove useful in future explorations of word representations and their relationships.

1 Introduction

The **COmparing Dictionaries and WOrd Embeddings** (`CODWOE`) task (Mickus et al., 2022) is aimed at explaining two different types of semantic descriptions of words: dictionary glosses and word embeddings. A *dictionary gloss* is a brief textual explanation of a word and a *word embedding* is a vector representation that captures the word's semantic and syntactic properties (Smith, 2020).

In order to investigate the relationship between these two types of descriptions, two complementary subtracks were put together: 1. *Definition Modeling* (`DEFMOD`) track, where correct glosses need to be generated from word embedding vectors (Noraset et al., 2017); and 2. *Reverse Dictionary* (`REVDICT`) track, where correct embedding vectors should be generated from dictionary glosses (Hill et al., 2016). The datasets for both tracks cover five different languages: English (EN), Spanish (ES), French (FR), Italian (IT), and Russian (RU).

The key challenge of the `CODWOE` task is that it needs to be performed without external data, which precludes the use of pretrained models and vectors. Additionally, the training dataset is relatively small in comparison to the datasets on which models are typically trained.

Our strategy was to adapt an RNN-based decoder model (Noraset et al., 2017) for the `DEFMOD` track, and to use a transformer-based encoder (Devlin et al., 2019) for the `REVDICT` track. With the limited amount of available data in mind, we hypothesized that models should not be large. Therefore we aimed to limit the model complexity by reducing the number of parameters, for example by using a subword tokenizer (Kudo and Richardson, 2018), which yields a smaller dictionary of optimized subword fragments. All of the models we used were built for a single language, and their structure and parameters were optimized either iteratively or by way of Bayesian hyperparameter optimization (BHO) (Snoek et al., 2012).

We conducted data analyses of the `CODWOE` datasets and analyses of the developed machine learning models. We performed a statistical and visual analysis of the pretrained `CODWOE` embeddings, i.e., of their distributions and relationships. `DEFMOD` analyses include an analysis of model performance factors and a qualitative analysis of generated glosses. In the `REVDICT` predictive analysis,

[*]Equal contribution.

Table 1: Aggregated language-level ranks of our team for the DEFMOD (DM) and REVDICT (RD) tracks (and the number of teams competing in a subtask).

TASK	EN	ES	FR	IT	RU
DM-all	2 (9)	**1** (7)	**1** (6)	5 (7)	5 (6)
RD-sgns	3 (9)	**1** (7)	**1** (6)	**1** (7)	2 (6)
RD-char	4 (7)	3 (5)	4 (5)	2 (6)	2 (5)
RD-electra	5 (6)		3 (4)		3 (4)

we investigate the impact of many different settings on models' performance defined in terms of distance and similarity scores between predicted and target vectors.

We show that our adaptation of the DEFMOD architecture (Noraset et al., 2017) can perform competitively and that the use of multiple word embeddings can clearly improve the generation of word glosses. For REVDICT, we demonstrate that our approaches achieve top performance in terms of ranking, which makes them suitable for information retrieval applications. Our models perform competitively and our results on the CODWOE challenge can be found in Table 1. We make the code of our models and data analyses publicly available[1].

2 Background

2.1 Related Work

Definition Modeling The Definition Modeling (DEFMOD) task, first introduced in Noraset et al., 2017, is focused on the prediction of dictionary word glosses from word embeddings. Noraset et al. (2017) experimented on two English dictionaries and proposed a successful architecture based on RNN.

Subsequent work on Definition Modeling focused on variations of the problem of prediction of a word gloss from the word sense. These approaches consider gloss prediction based on sense-specific word embeddings (Gadetsky et al., 2018; Kabiri and Cook, 2020; Zhu et al., 2019), and on a word-based context indicating the word sense (Bevilacqua et al., 2020; Gadetsky et al., 2018; Mickus et al., 2019; Yang et al., 2020; Zhang et al., 2020). The proposed approaches are based either on RNNs (Gadetsky et al., 2018; Kabiri and Cook, 2020; Zhang et al., 2020; Zhu et al., 2019) or Transformers (Bevilacqua et al., 2020; Mickus et al., 2019). All of the previous approaches rely on word embeddings pre-trained on large corpora, most commonly word2vec (Mikolov et al., 2013). Sense-aware approaches that take embeddings as input make use of either sense-aware word embeddings (Gadetsky et al., 2018; Kabiri and Cook, 2020) or of decomposition of word embeddings into sense-specific vectors (Zhu et al., 2019).

The initially proposed architecture of Noraset et al. (2017) is often used as a baseline solution. The most commonly used measure of model performance is the BLEU (Papineni et al., 2002) metric. Although there is some overlap in used datasets, most experiments rely on a specific dataset. The reported model performances vary greatly. Noraset et al. (2017) report BLEU of 31 and 23, depending on the dictionary. Subsequent experiments report, for the same approach, BLEU scores that range from as little as 11 (Gadetsky et al., 2018) to as much as 60 (Kabiri and Cook, 2020). The variation can be great even for the same language and experimental setup (Kabiri and Cook, 2020). The original approach of Noraset et al. (2017) remains competitive in the sense-aware setting, with the sense-aware approaches achieving BLEU increases that range between $1-2$ (Gadetsky et al., 2018; Kabiri and Cook, 2020; Zhang et al., 2020) and $5-6$ (Kabiri and Cook, 2020; Yang et al., 2020; Zhang et al., 2020), depending on the setting.

While we view the Definition Modeling primarily as a theoretically interesting task, potential applications include explainability of word embeddings and automatic generation of dictionaries, which might be of interest in low-resource settings.

Reverse Dictionary The Reverse Dictionary (REVDICT) is a task of finding the right word when a word description is given (Bilac et al., 2004; Dutoit and Nugues, 2002; Zock and Bilac, 2004). It is the formulation of the tip-of-the-tongue problem (TOT) (Brown and McNeill, 1966) that occurs during text synthesis. It is a condition in which a person knows a lot about the word, such as its meaning and origin, but is unable to recall it. REVDICT is a complex task. There are countless variations of input definitions that should lead to the same one-word concept. This complexity comes in part from the representation of the one-word concepts in the human mind. People tend to relate concepts on the conceptual and lexical level and form a highly connected network of abstractions (Zock and Bilac, 2004).

Therefore, a natural approach to solving REVDICT is to form a semantic network with nodes (one-word concepts) and edges (associ-

[1] https://github.com/dkorenci/codwoe-irb-nlp/

ations) to search for the target word (Thorat and Choudhari, 2016; Zock and Bilac, 2004). REVDICT can be realized directly by comparing the input definitions with all the definitions in the dictionary and returning the most similar ones, without taking into account any semantic or grammatical information (El-Kahlout and Oflazer, 2004). However, REVDICT systems that include semantics give better results, such as in Méndez et al., 2013 and Calvo et al., 2016 where words are represented as vectors in a semantic space.

Recent REVDICT approaches utilize deep learning (DL) to map arbitrary-length definition phrases to the vector representation of the target word (Hill et al., 2016; Malekzadeh et al., 2021; Qi et al., 2020; Yan et al., 2020). The success of DL approaches indicates that REVDICT can be solved implicitly, i.e. by directly learning from given data, and doesn't require an explicit injection of domain knowledge. According to this observation, the DL approach is a good choice for solving the REVDICT task.

2.2 Dataset

The CODWOE datasets (Mickus et al., 2022) cover five languages (EN, ES, FR, IT, RU) and are derived from the Dbnary lexical data[2]. Each data point corresponds to a single word and contains word embedding vectors and the word gloss. Three types of embedding are used, labeled as sgns (pretrained word2vec), electra (contextual pretrained embeddings) and char (character-based embeddings). Pretrained embeddings are based on large corpora containing approximately 1B tokens.

Each dataset is divided into three sections: training, validation (development), and test. Datasets for training and validation have 43.608 and 6.375 samples, respectively. Each track also has a separate set of test data. The DEFMOD test dataset has 6.221 samples while the REVDICT has 6.208 samples.

More detailed statistics and analyses of the dataset can be found in the Appendices, including the gloss statistics (Table 5) and embedding vector statistics (Table 11). Descriptive analysis of the embedding vectors shows large variation in values that depend on a language and an embedding type (Figures 3 and 4). Additionally, an exploratory analysis showed that the embeddings for different languages are easily separable (Figures 6 and 5). Interestingly, patterns of vector-based word similarity seem to differ significantly across embedding types, and in this regard there are no visible relations between different embeddings (Figure 7).

[2]http://kaiko.getalp.org/about-dbnary/

3 System overview

Both the DEFMOD and the REVDICT models rely on unigram subword tokenizers (Kudo, 2018) trained on glosses from the train datasets.

3.1 Definition Modeling

Our approach to the challenging task of Definition Modeling on a limited dataset consists of preprocessing the input data, extracting the semantic information from the dataset, and controlling the model size and complexity.

The inspection of the learning data revealed that the gloss texts are often long since they consist of several alternative definitions. We opted to include only one definition per learning example. Our intuition is that this approach, also taken in (Noraset et al., 2017), alleviates the learning problem by inducing the model to learn shorter and atomic definitions. The approach should also reduce noise (since the number of alternative definitions in a gloss is arbitrary).

The inspection of glosses also revealed the presence of lexicographic labels that precede the gloss definitions. These labels, present for all languages except English, convey data about, for example, word semantics (ex. geography, history) or temporal category (ex. archaic). We chose to remove these labels since they introduce noise (the presence and the amount of labels appears arbitrary), increase the dictionary size, and thus make the learning problem harder.

To construct the dictionary we use the unigram subword tokenizer (Kudo, 2018) implemented as part of the SentencePiece tool (Kudo and Richardson, 2018). The reasons for using the subword tokenization were the expected improvement in performance for low-resource tasks (Kudo, 2018) and the reduction in the number of model parameters corresponding to token embeddings.

Since we opted for a deep learning model depending on token embeddings, we initialized the token embeddings with GloVe vectors (Pennington et al., 2014) trained on the dataset of normalized and cleaned atomic glosses. To demonstrate that the GloVe vectors capture a degree of word semantics, we aggregated the vectors on a gloss level using tf-idf weighting. Then we inspected, for each "target" gloss from a sample of English glosses, other dataset glosses ordered by cosine similarity to the target. This revealed that GloVe similarity corresponds to the similarity in gloss meaning. Additionally, we found that the models initialized with GloVe vectors achieve a lower final loss.

Machine learning model We decided to use an adaptation of the RNN-based model of (Noraset et al., 2017), that proved competitive in a number of experimental settings. In the context of the DEFMOD task, the model takes as input one or more word embeddings (sgns, electra or char) and produces a gloss (a sequence of tokens) that should correspond to the word's correct gloss.

From the input embeddings, we form two vectors, the *seed* vector s that is used to initialize the RNN, and the *context* vector c. For both the seed and the context vectors we consider using a single embedding, concatenation of embeddings, and a nonlinear transformation of the concatenation. At each position in the sequence the context vector is passed as input, together with the RNN's output, to the special GRU-like gated cell (Noraset et al., 2017). The output of the gated cell is then transformed (via linear transformation and softmax activation) to produce token-level probabilities. The gated cell can learn to effectively combine the semantic context with the RNN-level features in guiding the generation process (Noraset et al., 2017). The network architecture we use is labeled as S+G in Noraset et al. (2017).

The described model performs conditional generation of tokens in a sequence, which is a standard approach in RNN-based language modeling. The probability of a gloss g is factorized under the assumption that each token g_i depends on the previous tokens, the seed embbedding s, and the context c :

$$p(g|s,c) = \prod_{i=1}^{|g|} p(g_i|g_{0:i-1}, s, c)$$

In (Noraset et al., 2017), the context is equal to the seed, i.e., the input word embedding. In our case, both the seed and the context can either be a single embedding or a function of multiple embeddings. This approach enables us to leverage the information from several word embeddings in a flexible way. For example, sgns embeddings can be used as a seed while the context can be formed by passing all the embeddings through a multilayer perceptron. Another important difference is that we use the unigram subword tokenization (Kudo, 2018). Finally, we experiment with using both LSTM and GRU as the network's RNN components.

3.2 Reverse Dictionary

We approach REVDICT as a supervised vector regression task and employ an end-to-end deep learning solution. Our model is based on a transformer architecture (Vaswani et al., 2017) used as a definition sentence encoder, and a fully connected feed-forward network used as an output regression module.

The transformer is used to produce useful representations from given inputs, where the inputs are tokenized definition sentences. For each subword token in the input sequence, the transformer gives a representation in the form of a vector. Our REVDICT systems implement three different approaches for aggregating the output vectors produced by the transformer: 1. *sum*, where we sum the representations given for each token in the input sequence; 2. *average*, where we average the representations given for each token in the input sequence; and 3. *eos*, where we use only the representation of the last token in the input sequence, i.e. end-of-sequence (eos) token. The output module further transforms these representations into word embedding vectors.

Additionally, we utilize a multi-task learning (Caruana, 1997; Ruder, 2017) approach. To support multi-task learning, we implemented multiple output regression modules that simultaneously predict different types of embedding vectors from the same representations produced by a single encoder. Multi-task learning is used during the model training phase and only output from one output module makes final predictions. The motivation for using a multi-task learning approach is to benefit from inductive transfer between tasks that could improve the results of predicting a single task (Caruana, 1997).

4 Model Selection and Experimental Setup

In this section we describe the technical details of data preprocessing and model selection that comprise our methods of constructing the DEFMOD and REVDICT models. The conceptual description of the methods is given in Section 3.

4.1 Definition Modeling

Our choices regarding the technical details of data preprocessing and model construction were guided by what we will call *development experiments*. These experiments consisted of training the model on the train set, and observing both the final development set loss and the quality of the produced glosses.

Output gloss quality was assessed using a separate “trial” dataset - a small dataset of 200 items provided by the organizers, containing gloss information consisting of the embedding vector, the original word, and the gloss text. The assessment was performed for English glosses only and aimed to assess the quality of the generated text, and the similarity of the output and the original glosses. A choice was deemed an improvement if it led to the improvement of development loss and either improved the generated glosses or caused no degradation in gloss quality. The development of the final algorithm was performed iteratively and heuristically. However, the overall improvement over the iterations is confirmed by the results of the test set evaluations.

Dataset transformation The transformation of the original dataset is performed by creating unambiguous training examples and removing the uninformative data that makes the problem harder.

In the original dataset a gloss definition often consists of several equivalent but differently phrased definitions. We divided the dictionary glosses into atomic definitions by splitting the text strings around the “;” character. This heuristic was motivated by gloss sample analysis and the inspection of a sample of atomic glosses revealed that it works in the majority of cases. Each atomic gloss in the new dataset was paired with all the embedding vectors of the original gloss.

In order to remove lexicographical labels from the beginning of the glosses' text, simple language-specific regular expressions and removal rules were formed based on gloss sample analysis. This approach proved to be effective for a large majority of glosses.

To perform further normalization we additionally lowercased all the glosses and removed the punctuation from the end of texts. The code used to preprocess the original dataset, the new dataset, and the transformation log can be found in the code repository. We note that both the SentencePiece dictionary and the GloVe vectors used for `DEFMOD` are derived from the transformed dataset. The statistics of the transformed glosses are presented in Table 6

Dictionary We used the unigram subword tokenizer (Kudo, 2018) available as part of the SentencePiece tool (Kudo and Richardson, 2018). and trained it using the default parameters. Experiments in Gowda and May (2020) suggest that a vocabulary of 8000 subwords is a good default choice for several languages in the case of machine translation. Additionally, our *development experiments* showed that English models using a vocabulary of 8000 subwords are superior to 10000 subword models. Therefore we decided to set the number of unigram tokens to 8000 in case of English, and to 8500 in case of other, highly inflected languages expected to have a higher number of distinct suffixes.

Pretrained token embeddings GloVe embeddings (Pennington et al., 2014) of the subword tokens, introduced to initialize the tokens with corpus-level semantic information, were constructed as follows. The model was trained on the set of transformed glosses, and the embedding size was fixed to 256 (the size of the gloss embeddings). The number of training iterations was set to 50, the “cutoff” parameter x_{max} was set to 10, while all the other parameters retained their default values. No frequency-based vocabulary pruning was performed.

Machine learning model We fixed the maximum sequence length of the RNN models to 64 subword tokens. Our intuition is that this alleviates the learning problem and could lead to models focused on generating shorter but more correct glosses.

The models were optimized using the AdamW algorithm (Loshchilov and Hutter, 2017) and the standard categorical cross-entropy loss. The training process was stopped after a fixed number of epochs, or if the best solution did not improve by more than 0.1% over 10 epochs. During inference, the optimal solution was constructed using the beam search algorithm implementation provided by the competition organizers[3].

We iteratively improved the models using the described *development experiments*, i.e., relying on the development set loss and analysis of model

[3]https://github.com/TimotheeMickus/codwoe/

Table 2: Characteristics of our REVDICT (RD) approaches (BS = batch size; ME = max epochs; HP = hyperparameter optimization points; S = scheduler; L = loss; MT = multi-task learning).

RD	BS	ME	HP	S	L	MT
1	1024	20	30	CS	MSE	no
2	2048	20	30	CS	MSE	no
3	4096	20	30	CS	MSE	no
4	8192	20	30	CS	MSE	no
5	2048	150	10	PS	MSE	no
6	2048	150	10	PS	MSE	yes

glosses produced for the trial dataset. We experimented with several architectural elements and hyperparameters: the formulation of the seed (RNN init. value) and context (gate input) of the network, RNN cell type, dropout, learning rate (LR) and LR scheduler, and the number of training epochs.

The most successful variant is constructed by using the concatenation of all the gloss embeddings as the context and the sgns embedding as the seed. This variant uses input dropout of 0.1 and network dropout of 0.3. The input dropout is applied to the seed and context vectors, as well as to the word embeddings. The network dropout is applied to the output of the RNN (final layer) and to the output of the gate cell. The chosen learning rate is 0.001, and the "plateau" LR scheduler is used – LR is multiplied by 0.1 if there is no improvement over 5 epochs.

For the context vector, we tried single embeddings and the combined embeddings merged via a multilayer perceptron. Both variants proved inferior to the concatenation of all vectors. The merged seed vector proved no different from the single embedding seed, so we opted for the simpler solution. Both the development experiments and the results showed no difference between the LSTM and the GRU cell.

Analysis of errors revealed that models sometimes produce a deformed output (very short or non-alphabetic string), and that this almost never occurs simultaneously for two distinct models. Therefore a way of heuristic model improvement is to combine it with another *fallback* model to be used in case of deformed outputs. We combined a model with a concatenated context and a model with a single-embedding context, or two models with distinct RNN cell types. A more detailed analysis of the model variants can be found in Appendix A.2.

4.2 Reverse Dictionary

We conducted various development experiments before deciding on the final configuration of our REVDICT solutions. In all of the experiments, we used the entire set of train data to train the model, and the entire set of validation (development) data for scoring. We used *Mean Squared Error* (MSE) as a loss function during training. We tested the effect of cosine loss if added to MSE with different coefficients, but we obtained the best results without cosine loss. We also used MSE for scoring models during Bayesian hyperparameter optimization (BHO).

To determine the optimal model size, we searched the space of two transformer hyperparameters: the number of heads and the number of layers. We used a grid search approach with these values $v \in \{1, 2, 4, 8\}$ for both hyperparameters. Additionally, we used BHO (Snoek et al., 2012) to find the optimal model for each grid point. However, the increase in model size did not increase the performance of the model. These results were in line with the expectations we had due to the small size of the datasets. Accordingly, we decided to use a transformer with two heads and two layers. Additionally, we experimented with the maximum length of the input sequence and achieved better validation performance with 256 tokens than 512 with tokens.

We compared performance with and without token embeddings initialization with GloVe vectors. Contrary to our expectations, there was no significant difference in validation performance between these two options, so we skipped the GloVe initialization in the REVDICT system settings. Another development experiment we conducted was to find the optimal method for aggregating the output vectors produced by the transformer, described in Section 3.2. We found that the *average* method gives the best results in all cases. Furthermore, we examined the influence of the number of layers in the output module on the final prediction. According to the results, there is no benefit in increasing the number of layers in the output module, so we chose a single-layer fully connected network. We also chose Rectified Linear Unit (ReLU) activation function for the output regression module, because it yielded better performance than hyperbolic tangent (Tanh) activation.

Finally, we made six different solutions for REVDICT task. All of these solutions used a two-

Table 3: Results for the IRB-NLP team systems on the DEFMOD task. MoverScore, BLEU, and lemma-BLEU results are given for each of the five languages. Best result across all teams and models is given, followed by the results of our two best systems. Overall best results of our team are bolded and the rankings can be found in Table 1.

	EN			ES			FR			IT			RU		
	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU
BEST	0.135	0.033	0.043	0.128	0.045	0.064	0.075	0.029	0.038	0.117	0.066	0.099	0.148	0.049	0.072
IRBv3	0.089	0.032	0.040	0.093	**0.045**	**0.064**	0.055	0.026	0.032	0.074	0.009	0.014	0.080	0.027	0.035
IRBv4	0.094	**0.033**	0.042	0.092	0.044	0.062	0.056	0.028	0.033	0.077	0.010	0.015	0.078	0.027	0.036

head transformer architecture, where each head consists of two layers. We used a vocabulary size of 8000 tokens and a maximum sequence length of 256 tokens. The unigram SentencePiece tokenizers used were trained on lowercased but otherwise unmodified glosses contained in a train set. We used the *average* method for combining an encoder's output representations and fed them to the output module, which is a single fully-connected layer with RELU activation functions. We varied five hyperparameters between solutions (Table 2): batch size, max. epochs, number of BHO points, scheduler type, and learning approach. We used the *Cosine Annealing with Linear Warmup* scheduler (CS) for the first four solutions, and the *Plato* scheduler (PS) for the final two solutions. We utilized BHO (Snoek et al., 2012) to automatically search for optimal hyperparameters and submitted for testing only models with the best MSE validation scores.

5 Results

Definition Modeling On the DEFMOD task, the models were evaluated using three metrics: BLEU score (Papineni et al., 2002), lemma-level BLEU score, and MoverScore (Zhao et al., 2019). While the BLEU score is based on matching token n-grams between the reference and the model-produced text, MoverScore calculates a measure of distance between texts embedded in a semantic space, i.e., between two sets of contextual word embeddings computed using a transformer model.

Table 3 contains scores for two of our best model configurations, "version 3" and "version 4". Both model configurations are described in detail at the end of Section 4.1. While version 3 models are based on GRU RNN and trained using 300 training epochs, version 4 models are built with either GRU or LSTM and 450 epochs. The fallback strategy, which yields slight performance gains, is also used. These results are presented and analyzed in a more detailed manner in Appendix A.2.

Results in Table 3 show that our models are competitive with other teams' models on English, Spanish and French, especially in terms of the BLEU scores. MoverScore results are weaker than those produced by the top models, but rank among the upper half of the systems except for Italian and Russian, languages for which our models' performance is below average. Rankings aggregated across all the scores, displayed in Table 1, reflect the above observations and show that the models we produced can perform quite competitively.

Our approach shows inter-language variation, both in relative (ranks) and absolute (score values) terms. The full results provided by the organizers[4] show that this is also true for other teams – for example, few of the high-performing models perform markedly better for Italian and Russian than for other languages. However, some approaches yield more stable results across all languages.

All of the models yielded by the CODWOE shared task perform weakly in terms of BLEU. Namely, the BLEU scores of the existing DEFMOD approaches commonly achieve BLEU scores in the range of 20 to 30 (Kabiri and Cook, 2020; Noraset et al., 2017), with some settings yielding BLEU as high as 60 (Kabiri and Cook, 2020). The experiments with the weakest reported BLEU scores (Gadetsky et al., 2018; Kabiri and Cook, 2020) reports BLEU scores of approx. 12, while the best CODWOE scores are below BLEU 10.

CODWOE DEFMOD models perform better in terms of MoverScore, a metric designed for machine summarization (Zhao et al., 2019). An analysis of a number of summarization systems showed that MoverScore values range between 15 and 24, with an absolute minimum of 10 and an average slightly below 20 (Fabbri et al., 2021). In comparison, top CODWOE systems reach scores between 12 and 15, except in the case of French, which puts them on the lower end of the summarization scale.

We hypothesize that the main reason for the described weak performance is comparatively small amount of CODWOE training data (for each indi-

[4]https://github.com/TimotheeMickus/codwoe/

Table 4: Results for the IRB-NLP team systems on the REVDICT task. The best result over all teams and models is given (BEST), followed by the best results of our team (IRB-all) and results of our two specific approaches, IRB-v1 and IRB-v6. Finally, the ranks of our team are given (and the number of teams competing in a subtask).

		EN			ES			FR			IT			RU		
		MSE	COS	RNK	MSE	COS	RNK	MSE	COS	RNK	MSE	COS	RNK	MSE	COS	RNK
sgns	BEST	0.854	0.260	0.231	0.858	0.403	0.167	1.026	0.342	0.193	1.031	0.380	0.165	0.528	0.424	0.150
	IRB-all	0.964	**0.260**	**0.231**	0.883	0.367	0.197	1.068	**0.342**	**0.193**	1.076	**0.380**	**0.165**	0.568	0.421	**0.150**
	IRB-v1	1.024	0.250	0.247	0.941	0.362	0.197	1.068	0.342	0.214	1.076	0.380	0.165	0.568	0.412	0.161
	IRB-v6	1.119	0.214	0.262	1.020	0.354	0.201	1.319	0.255	0.262	1.318	0.339	0.187	0.653	0.381	0.150
	IRB-rnk	9 (9)	**1** (9)	**1** (9)	3 (7)	2 (7)	2 (7)	3 (6)	**1** (6)	**1** (6)	3 (7)	**1** (7)	**1** (7)	4 (6)	2 (6)	**1** (6)
char	BEST	0.141	0.798	0.419	0.467	0.839	0.403	0.335	0.789	0.416	0.334	0.747	0.383	0.116	0.852	0.357
	IRB-all	0.162	0.770	**0.419**	0.526	0.819	**0.403**	0.390	0.756	0.421	0.366	0.724	**0.383**	0.140	0.824	**0.357**
	IRB-v1	0.169	0.761	0.438	0.526	0.819	0.407	0.409	0.744	0.425	0.366	0.724	0.397	0.145	0.818	0.361
	IRB-v6	0.172	0.765	0.444	0.635	0.784	0.420	0.434	0.734	0.421	0.399	0.711	**0.383**	0.144	0.821	**0.357**
	IRB-rnk	5 (7)	7 (7)	**1** (7)	3 (5)	5 (5)	**1** (5)	3 (5)	4 (5)	2 (5)	5 (6)	5 (6)	**1** (6)	3 (5)	4 (5)	**1** (5)
electra	BEST	1.301	0.847	0.432				1.066	0.862	0.429				0.828	0.735	**0.345**
	IRB-all	1.685	0.828	**0.432**				1.339	0.847	**0.429**				0.911	0.724	**0.345**
	IRB-v1	1.723	0.821	0.438				1.339	0.847	0.447				0.911	0.724	0.350
	IRB-v6	1.988	0.792	**0.432**				1.566	0.825	**0.429**				1.049	0.702	**0.345**
	IRB-rnk	6 (6)	6 (6)	**1** (6)				4 (4)	4 (4)	**1** (4)				4 (4)	3 (4)	**1** (4)

vidual language), as well as the lack of word embeddings pre-trained on a large outside corpus. Namely, most of the other DEFMOD approaches use at least 2–3 times more training data, both in terms of the number of (embedding, text) examples, and the overall number of tokens (Bevilacqua et al., 2020; Gadetsky et al., 2018; Mickus et al., 2019; Noraset et al., 2017; Yang et al., 2020; Zhang et al., 2020; Zhu et al., 2019). Additionally, these approaches make use of the pre-trained word embeddings that carry the semantic information extracted from a huge corpus.

As for the representativeness of the test data, the visual analysis performed in A.1 shows that the distribution of test gloss embeddings matches the train distribution well. Another factor that potentially influences performance is word rarity. We observed that the English test examples contain a significant amount of rare words (such as "pelta", "akimbo", "gothy", or "dungarees"), while some DEFMOD experiments explicitly focus on the most frequent words (Noraset et al., 2017).

The greatest performance gains for the models we used come from using all three vector embeddings to form a context vector. This suggests that future approaches can benefit from leveraging several distinct embeddings types as input for gloss generation.

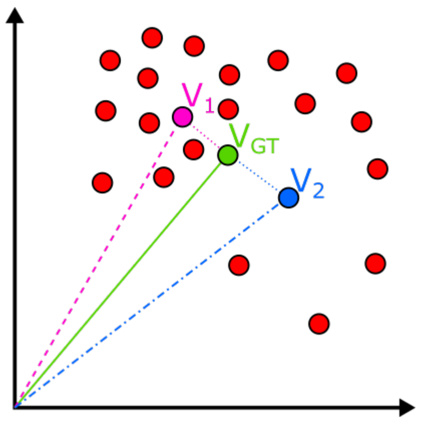
We believe that the question of the influence of various factors on the performance of DEFMOD systems is important and under-explored. These factors include model structure and parameters, performance metric, dataset size (both for training and pre-training), and the semantic relation between training and test data. Closely related is the question of the nature of semantic generalization that DEFMOD systems are capable of – what kind of examples (and relations contained within them) can inform a successful inference of glosses for unseen embeddings.

Further performance-related analyses can be found in Appendix A.2. Appendix A.3 contains a qualitative analysis of glosses that shows that generated glosses can capture varying levels of semantic properties of the correct glosses. We hypothesize that these variations in similarity are hard to capture with metrics such as MoverScore and BLEU.

Reverse Dictionary We used the following metrics for internal validation of our REVDICT solutions (described in Section 4.2): *Mean Squared Error* (MSE), *Cosine Similarity* (COS), and *Central Kernel Alignment* (CKA) (Cortes et al., 2012; Kornblith et al., 2019). COS measure has noted drawbacks (Heidarian and Dinneen, 2016). Therefore, we use the linear CKA similarity measure to gain another perspective on model performance. Validation scores can be found in Appendix B.2, Table 12. It is evident that each subsequent approach gives better validation results than the previous ones.

Test predictions were scored by the following metrics: MSE, COS, and *Cosine-Based Ranking* (RNK). The RNK measure is defined as the proportion of test samples with cosine similarity to the model output embedding higher than the ground

Figure 1: Example of two different predictions for ground truth vector V_{GT}, where predicted vector V_1 has better MSE and COS scores than V_2, and V_2 has better RNK score than V_1. The rest of the points represent vectors of other test samples.



truth embedding. The final results of our solutions can be found in Table 13 (see Appendix B.2). Here, each subsequent approach has lower scores than the previous ones, which is the complete opposite of the validation results. This suggests potential overfitting to the dev dataset that could be the result of BHO. However, this is contrary to expectations as the last two solutions have three times fewer BHO points and should not overfit to the dev dataset. The reason for this phenomenon is unclear and needs further investigation. Finally, the best `REVDICT` results for each team can be found in Appendix B.2 (Table 14 for MSE, Table 15 for COS, and Table 16 for RNK). The test results and overall rankings of our solutions are summarized in Table 4.

Compared to other solutions, our systems have average or below-average performance in terms of MSE and COS test scores. However, they perform significantly better than the other approaches in terms of RNK test scores, from which we conclude that our solutions are better suited for the retrieval task. This is an interesting situation which we elaborate with the following example, shown in Figure 1. It depicts two different predictions, V_1 and V_2, the first with better MSE and COS scores, and the second with a better RNK score. The second solution prefers a vector subspace with a lower density of test samples even if the absolute distance from the correct vector is greater. With a smaller set of possible surrounding solutions, retrieving the vector V_{GT} from the vector V_2 is more precise than retrieving it from the vector V_1.

6 Conclusion

Definition Modeling and Reverse Dictionary are two opposite learning tasks for exploring the relationship between different semantic representations of words. `CODWOE` SemEval task (Mickus et al., 2022) is designed to investigate these tasks on five different languages using three different types of word embeddings.

We propose an adaptation of an existing `DEFMOD` model and analyze its performance and the glosses generated by the model. We believe that `DEFMOD` is a theoretically interesting problem and that further investigations should focus on discovering which types of semantic generalization the models are able to perform, and how this generalization ability is influenced by both the data and the models' structure. The existing `DEFMOD` experiments are largely incomparable since they are based on different data and setups. We believe that a contribution of the `CODWOE` task is the creation of a multilingual evaluation setting, as well as the use of the flexible MoverScore as an evaluation metric.

Our `REVDICT` systems are based on deep regression models based on transformer architecture that achieved top scores for the difficult-to-predict `sgns` (word2vec) embeddings. In most cases our `REVDICT` solutions perform significantly better then the other systems in terms of the RNK score. These results imply that our solutions could be the appropriate approach for retrieving the right word from its description, a problem crucial for solving the TOT problem (Brown and McNeill, 1966) in machine-assisted text synthesis.

In summary, the models that we produced for the `CODWOE` task perform competitively when compared to other participants' models, and can therefore serve as a reasonable starting point for future tackling of `DEFMOD` and `REVDICT` problems. We believe that the promising directions for future optimizations include the construction of multilingual and multi-task models, as well as investigations of the influence of the external data, primarily in the form of huge pre-training corpora.

Acknowledgments

We would like to thank the anonymous reviewers for the useful comments, as well as Timothee Mickus for the help with challenge-related questions. We would also like to thank Miha Keber and Tomislav Lipić for helpful discussions and advice.

References

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221, Online. Association for Computational Linguistics.

Slaven Bilac, Wataru Watanabe, Taiichi Hashimoto, Takenobu Tokunaga, and Hozumi Tanaka. 2004. Dictionary search based on the target word description. In *Proc. of the Tenth Annual Meeting of The Association for Natural Language Processing (NLP2004)*, pages 556–559.

Roger Brown and David McNeill. 1966. The "tip of the tongue" phenomenon. *Journal of verbal learning and verbal behavior*, 5(4):325–337.

Hiram Calvo, Oscar Méndez, and Marco A Moreno-Armendáriz. 2016. Integrated concept blending with vector space models. *Computer Speech & Language*, 40:79–96.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. 2012. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13:795–828.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Dominique Dutoit and Pierre Nugues. 2002. A lexical database and an algorithm to find words from definitions. In *ECAI*, volume 45, pages 0–454. Citeseer.

Ilknur Durgar El-Kahlout and Kemal Oflazer. 2004. Use of wordnet for retrieving words from their meanings. In *Proceedings of the global Wordnet conference (GWC2004)*, pages 118–123.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 266–271, Melbourne, Australia. Association for Computational Linguistics.

Thamme Gowda and Jonathan May. 2020. Finding the Optimal Vocabulary Size for Neural Machine Translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3955–3964, Online. Association for Computational Linguistics.

Arash Heidarian and Michael J Dinneen. 2016. A hybrid geometric approach for measuring similarity level among documents and document clustering. In *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)*, pages 142–151. IEEE.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Arman Kabiri and Paul Cook. 2020. Evaluating a multi-sense definition generation model for multiple languages. In *International Conference on Text, Speech, and Dialogue*, pages 153–161. Springer.

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

Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Arman Malekzadeh, Amin Gheibi, and Ali Mohades. 2021. Predict: Persian reverse dictionary. *arXiv preprint arXiv:2105.00309*.

Oscar Méndez, Hiram Calvo, and Marco A Moreno-Armendáriz. 2013. A reverse dictionary based on semantic analysis using wordnet. In *Mexican International Conference on Artificial Intelligence*, pages 275–285. Springer.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Timothee Mickus, Denis Paperno, and Matthieu Constant. 2019. Mark my Word: A Sequence-to-Sequence Approach to Definition Modeling. In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 1–11, Turku, Finland. Linköping University Electronic Press.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3259–3266. AAAI Press.

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

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fanchao Qi, Lei Zhang, Yanhui Yang, Zhiyuan Liu, and Maosong Sun. 2020. Wantwords: An open-source online reverse dictionary system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–181.

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

Noah A. Smith. 2020. Contextual word representations: Putting words into computers. *Commun. ACM*, 63(6):66–74.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Sushrut Thorat and Varad Choudhari. 2016. Implementing a reverse dictionary, based on word definitions, using a node-graph architecture. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2797–2806, Osaka, Japan. The COLING 2016 Organizing Committee.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. 2020. Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *arXiv preprint arXiv:2012.04456*.

Hang Yan, Xiaonan Li, Xipeng Qiu, and Bocao Deng. 2020. BERT for monolingual and cross-lingual reverse dictionary. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4329–4338, Online. Association for Computational Linguistics.

Liner Yang, Cunliang Kong, Yun Chen, Yang Liu, Qinan Fan, and Erhong Yang. 2020. Incorporating sememes into chinese definition modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1669–1677.

Haitong Zhang, Yongping Du, Jiaxin Sun, and Qingxiao Li. 2020. Improving interpretability of word embeddings by generating definition and usage. *Expert Systems with Applications*, 160:113633.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Ruimin Zhu, Thanapon Noraset, Alisa Liu, Wenxin Jiang, and Doug Downey. 2019. Multi-sense Definition Modeling using Word Sense Decompositions.

Michael Zock and Slaven Bilac. 2004. Word lookup on the basis of associations : from an idea to a roadmap. In *Proceedings of the Workshop on Enhancing and Using Electronic Dictionaries*, pages 29–35, Geneva, Switzerland. COLING.

A Appendix - Analysis of DEFMOD Data and Models

A.1 Train and Test Data

Motivated by the weak performance of DEFMOD models (see Section 5), we examined whether the distributions of train and test data are comparable. To this end we created 2D projections of sgns and electra embedding for all five languages using the t-SNE method (Van der Maaten and Hinton, 2008).

The projections, depicted in Figure 2, show that the train and test distributions of the embeddings match well. It is therefore reasonable to expect that the distributions of the gloss texts are similar as well, as the gloss semantics expectedly matches the semantics of the corresponding words. However, this conjecture should be confirmed experimentally, for example by per-gloss aggregation of pretrained word embeddings extracted from huge corpora.

Figure also shows that the electra vectors are more separable than the sgns vectors. The separability of the embedding vectors varies across languages, probably influenced by the corpora used for pre-training of the embeddings. We note that the observations about the train and test embedding distributions are also applicable to the REVDICT problem aimed at the prediction of the embeddings from gloss texts.

Basic gloss statistics can be found in Table 5. There exists a large variation in gloss size between languages, e.g., the longest gloss from the ES dataset is almost twice the size of the longest EN gloss. In addition, the longest glosses in the validation (development) datasets are significantly smaller then those in the train datasets, on average 42.55% smaller. The 'dictionary size' column in the table is the number of distinct tokens in each dataset. Dictionary sizes vary, for example, EN dictionary is approximately half the size of the RU dictionary. Differences between the gloss and dictionary sizes suggest that it is reasonable to use a separate model for each language.

Basic statistics of the transformed dataset can be found in Table 6. As expected, the transformed glosses are significantly smaller then the glosses in the original dataset. For example, the median transformed gloss size is on average 29.25% smaller.

Lang.	Split	Dict. size	#Tokens	#Glosses	Gloss size						
					mean	st.dev	min	q25	median	q75	max
EN	train	29.046	511.531	43.608	11.73	7.98	1	6.0	10.0	15.0	129
EN	dev	9.478	76.073	6.375	11.93	7.98	1	6.0	10.0	15.0	70
ES	train	46.765	647.093	43.608	14.84	13.07	1	7.0	11.0	18.0	257
ES	dev	15.464	91.943	6.375	14.42	12.22	1	7.0	11.0	17.0	159
FR	train	40.032	623.978	43.608	14.31	9.74	1	8.0	12.0	18.0	159
FR	dev	12.760	91.475	6.375	14.35	9.91	1	8.0	12.0	18.0	113
IT	train	40.130	592.409	43.608	13.58	11.01	1	6.0	11.0	18.0	202
IT	dev	14.069	87.531	6.375	13.73	11.61	1	6.0	11.0	18.0	130
RU	train	57.141	492.978	43.608	11.30	7.78	1	6.0	9.0	14.0	169
RU	dev	15.498	70.392	6.375	11.04	7.22	1	6.0	9.0	14.0	74

Table 5: Statistics of the gloss and dictionary sizes for the original train and validation (development) datasets. Sizes are calculated by counting the number of whitespace-delimited tokens.

A.2 DEFMOD Models' Performance

Here we append Section 5 with a more fine-grained analysis of the DEFMOD models. Table 8 contains the models' performances. As can be seen, the largest gains are achieved by using all of the embedding vectors as input for gloss generation (context=allvec). There exists a negligible difference between the LSTM and GRU RNNs, with GRU performing slightly better. Using a fallback model always slightly improves the MoverScore of a model. In Table 8 the architecture of the fallback model is the architecture of the main model with the corresponding parameter replaced with the value in the 'fallback' column. Interestingly, using contextual electra vectors does not help, i.e., the sgns (word2vec) vectors which are not context-aware perform comparably. This is true even when only a single embedding is used, i.e.,

Lang.	Split	Dict. size	#Tokens	#Glosses	Gloss size						
					mean	st.dev	min	q25	median	q75	max
EN	train	25.921	456.673	58.792	7.77	6.97	1	3.0	6.0	10.0	128
EN	dev	8.892	68.145	8.403	8.11	7.06	1	3.0	6.0	11.0	69
ES	train	40.024	595.879	44.543	13.38	12.01	1	6.0	10.0	16.0	168
ES	dev	13.723	84.303	6.493	12.98	11.32	1	6.0	10.0	16.0	158
FR	train	33.963	487.013	46.537	10.47	9.18	1	4.0	8.0	14.0	155
FR	dev	11.216	71.021	6.786	10.47	9.22	1	4.0	8.0	14.0	101
IT	train	39.124	452.028	45.080	10.03	9.03	1	4.0	7.0	13.0	195
IT	dev	13.805	67.211	6.621	10.15	9.40	1	4.0	7.0	13.0	109
RU	train	56.467	428.787	50.843	8.43	6.99	1	4.0	7.0	11.0	142
RU	dev	15.241	61.074	7.509	8.13	6.44	1	4.0	6.0	11.0	72

Table 6: Statistics of the gloss and dictionary sizes for the transformed train and validation (development) datasets. Sizes are calculated by counting the number of whitespace-delimited tokens.

when context equals `electra`. The equality of `sgns` and `electra` is unexpected since both the train and test datasets contain polysemous `electra` vectors and words with multiple senses.

It is also interesting to consider the influence of the training data on the model's performance. We hypothesize that a `DEFMOD` model's score on a single test example is positively correlated with the semantic closeness of the example to the examples in the train set. To test this hypothesis we calculate Spearman correlation between test MoverScore and BLEU on one, and the cosine similarity of the test embedding and most similar train embeddings. This is done for the best-performing submitted model from Table 8. We also calculate the average scores on two sets of 10% test examples that are least similar and most similar to the train examples. Since the embeddings (`sgns` and `electra`) were built on large outside corpora, it is reasonable to believe that they capture semantic similarity of the associated words and glosses. Surprisingly, the results show a lack of consistent and strong correlation and the correlations range from weakly negative to weakly positive, depending on both the language and the embedding type. This lack of correlation could be caused by many factors, including the nature of the model, the nature of the pretrained embeddings, and the semantics of the cosine similarity measure.

The future extensions and improvements of the proposed analysis could reveal the nature of the train data necessary for the `DEFMOD` models to successfully generalize, and perhaps point to a similarity measure that reveals more fine-grained properties of such a generalization.

Table 7: Correlation between the best `DEFMOD` model's scores on one, and the closeness of the test examples to the train set on the other side. The unit of correlation is an example from the test set, and its similarity to the train set is calculated as the average cosine similarity with the 10 most similar train embeddings. The last two columns contain average model scores on 10% of the least and most train-similar test examples.

	Correlation of Score and Similarity				Avg. Score for Similarity Percentile			
	MVR		BLEU		MVR		BLEU	
LANG-EMB	spearman ρ	p-value	spearman ρ	p-value	bottom 10%	top 10%	bottom 10%	top 10%
EN-SGNS	0.0458	0.0003	0.0019	0.8831	0.0852	0.1004	0.0328	0.0297
EN-ELKT	0.0096	0.4508	0.0186	0.1414	0.0889	0.1182	0.0293	0.0503
FR-SGNS	-0.0625	0.0000	-0.1270	0.0000	0.0801	0.0427	0.0363	0.0222
FR-ELKT	0.0433	0.0006	0.0838	0.0000	0.0387	0.0760	0.0214	0.0322
RU-SGNS	0.0758	0.0000	0.0353	0.0054	0.0754	0.0947	0.0310	0.0279
RU-ELKT	0.0000	0.9979	-0.0063	0.6217	0.0748	0.0677	0.0279	0.0244
ES-SGNS	0.0458	0.0003	0.0019	0.8831	0.1084	0.1052	0.0523	0.0552
IT-SGNS	-0.0528	0.0000	0.0047	0.7084	0.1082	0.0783	0.0111	0.0128

Figure 2: t-SNE projections of the `sgns` and `electra` vectors from the train (green) and test (red) datasets. Color intensity is proportional to data density.

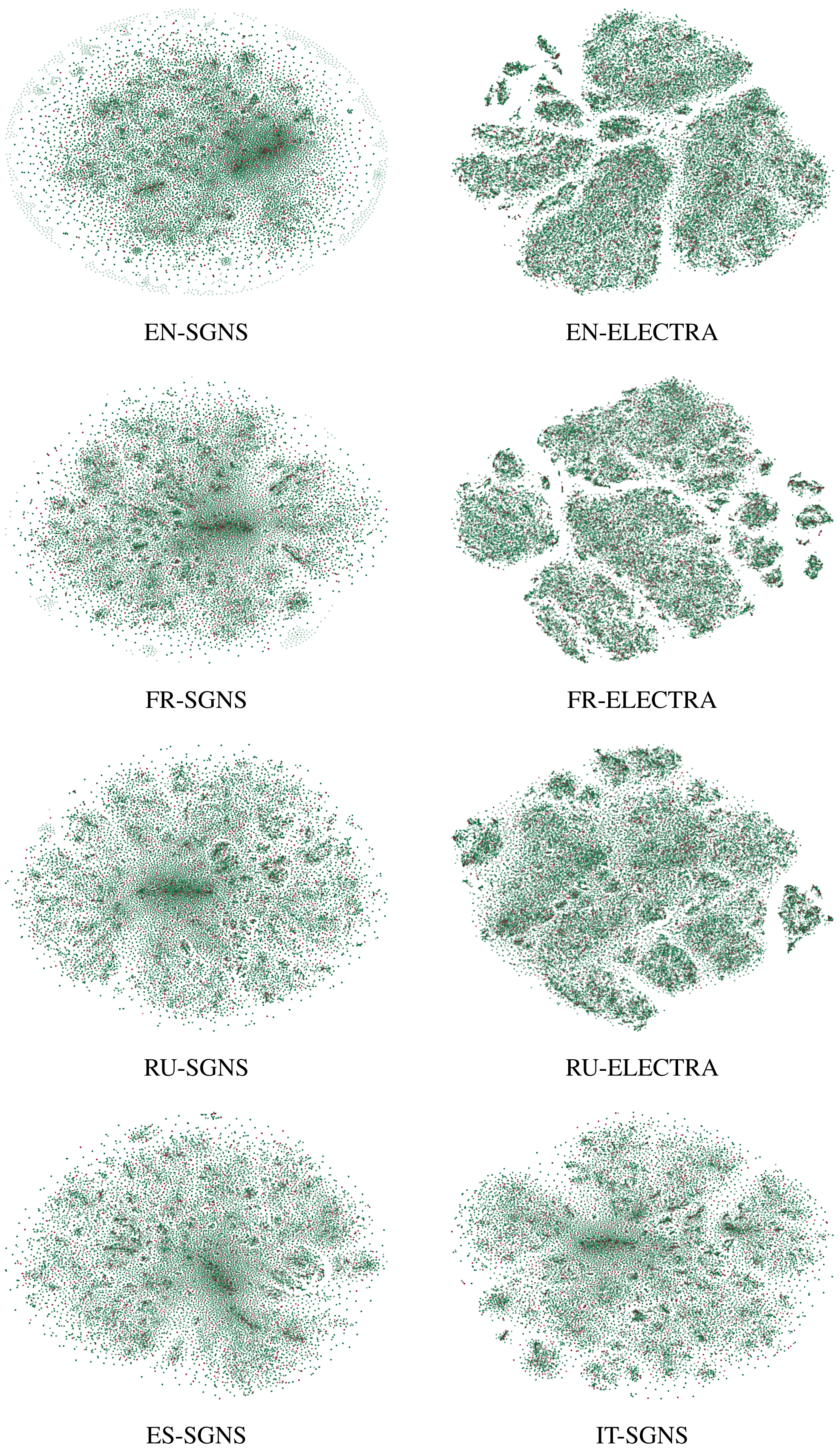


Table 8: Performance of best DEFMOD models. Overall best models from all the participants are included for comparison. Submitted models are colored gray and the best submitted results are bolded. The best overall results are colored red. Other models are included to illustrate the influence of design choices.

MODEL PARAMS					EN			ES			FR			IT			RU		
context	seed	rnn	fallback	#epochs	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU	MVR	BLEU	lBLEU
		BEST			0.135	0.033	0.043	0.128	0.045	0.064	0.075	0.029	0.038	0.117	0.066	0.099	0.148	0.049	0.072
`sgns`	`sgns`	`gru`		300	0.070	0.027	0.034	0.083	0.039	0.058	0.039	0.024	0.028	0.073	0.009	0.014	0.073	0.023	0.031
`allvec`	`sgns`	`gru`		300	0.085	0.031	0.040	0.092	0.045	0.064	0.048	0.026	0.032	0.072	0.009	0.014	0.077	0.027	0.035
`allvec`	`sgns`	`gru`	`sgns`	300	0.089	0.032	0.040	**0.093**	**0.045**	**0.064**	0.055	0.026	0.031	0.074	0.009	0.014	**0.080**	0.027	0.035
`sgns`	`sgns`	`gru`		450	0.077	0.029	0.038	0.080	0.037	0.057	0.048	0.026	0.031	0.071	0.009	0.014	0.073	0.022	0.029
`sgns`	`sgns`	`lstm`		450	0.075	0.028	0.035	0.082	0.038	0.056	0.052	0.025	0.030	0.070	0.015	0.009	0.073	0.022	0.030
`allvec`	`sgns`	`gru`		450	0.093	0.033	0.042	0.089	0.044	0.062	0.049	**0.028**	**0.033**	0.076	**0.010**	**0.015**	0.075	**0.027**	**0.036**
`allvec`	`sgns`	`lstm`		450	0.091	0.033	0.041	0.092	0.044	0.061	0.051	0.027	0.032	0.076	0.010	0.015	0.076	0.026	0.033
`allvec`	`sgns`	`gru`	`sgns`	450	**0.096**	**0.033**	**0.042**	**0.096**	**0.044**	0.063	**0.061**	**0.028**	**0.033**	**0.077**	**0.010**	**0.015**	**0.082**	**0.028**	**0.037**
`allvec`	`sgns`	`lstm`	`sgns`	450	0.095	0.033	0.042	0.093	0.044	0.061	0.057	0.027	0.032	0.077	0.010	0.015	0.079	0.026	0.034
`allvec`	`sgns`	`lstm`	`gru`	450	**0.094**	**0.033**	**0.042**	0.092	0.044	0.061	**0.056**	0.027	0.032	**0.077**	0.010	0.015	0.078	0.026	0.033
`electra`	`electra`	`gru`		450	0.079	0.028	0.034				0.050	0.029	0.024				0.072	0.024	0.031
`allvec`	`electra`	`gru`		450	0.091	0.032	0.041				0.047	0.027	0.032				0.073	0.027	0.035
`electra`	`electra`	`gru`	`sgns`	450	0.094	0.033	0.041				0.058	0.026	0.031				0.082	0.028	0.036

A.3 Qualitative Analysis of Generated Glosses

The `DEFMOD` models achieve weak results in comparison to the previous state-of-art approaches, which is probably due to the comparably small amount of training and pretraining data. Here we demonstrate that the generated glosses can nevertheless capture a degree of the semantics of the correct glosses.

Table 9 shows four categories of semantic similarity between the correct and model-generated glosses, in descending order (highest similarity first). These categories include hits or near hits (correct glosses), "near misses" (glosses that capture a significant amount of the original meaning), somewhat similar glosses, and complete misses. Several examples demonstrate that the subword-based models can produce syntactically incorrect glosses.

Table 10 contains generated glosses for different senses of the word "consider", which demonstrate that the model was able to approximate, to a degree, the semantics of the senses.

A principled analysis of the generated and correct glosses, based on a well defined semantic annotation scheme, might prove revealing but it would be time-consuming and impractical. Therefore it would be of interest to automatize such efforts. It would be interesting to explore if this can be done using large pretrained transformers able to measure fine-grained semantic similarity.

Table 9: Glosses generated by the top submitted `DEFMOD` model, alongside the correct glosses. The examples are ordered by descending semantic similarity between the correct and the generated gloss.

Word	True Gloss / `Generated Gloss`
lamebrain	A fool
	`A fool , idiot`
sentiment	A general thought , feeling , or sense
	`A feeling or feeling of thinking`
available	Capable of being used for the accomplishment of a purpose
	`Able to be used`
model	A representation of a physical object , usually in miniature
	`An act of designing`
supernumerary	Of an organ or structure : additional to what is normally present
	`Having four wings`
navy	Belonging to the navy ; typical of the navy
	`To be armed`
fuzzy	Vague or imprecise
	`lacking`
co-opt	To absorb or assimilate into an established group
	`To conceal`
misinformation	Information that is incorrect
	`prejudice`
cutthroat	Ruthlessly competitive , dog-eat-dog
	`Very large`
discretional	discretionary
	`Of or pertaining to`
abundantly	In an abundant manner ; in a sufficient degree ; in large measure
	`In a very manner`

Glosses generated by the top submitted `DEFMOD` model, alongside the correct glosses, for the multiple senses of the word “consider”.

Table 10

Word	True Gloss (describing the sense) / `Generated Gloss`
consider	To assign some quality to
	`To hold the opinion`
consider	To look at attentively
	`To make something certain`
consider	To have regard to ; to take into view or account ; to pay due attention to ; to respect
	`To hold into`
consider	To think of doing
	`To permit`
consider	To debate (or dispose of) a motion
	`To make something certain`

B Appendix - Analysis of `REVDICT` Data and Models

B.1 Data Analysis

Here we analyze the properties of the pretrained embedding vectors assigned to the words defined by the glosses. We start by analyzing the numeric values contained in the vectors. Basic statistics of vector elements can be found in Table 11. It is noticeable that there are large variations in value depending on the language and the embedding type. For example, there is a significant difference between maximum values, especially between `electra` and `sgns`. To further investigate the vector elements, we visualize the shapes of their distributions for train datasets (Figure 3 and 4). Distribution shapes look similar for dev datasets.

Next, we explore the vector data by reducing dimensionality to the 2D space using the Pairwise Controlled Manifold Approximation Projection (PaCMAP) algorithm (Wang et al., 2020). Figure 5 shows the distributions of all three types of embeddings in the train and validation (development) datasets for English, French, and Russian. We also visualize distributions of `sgns` (word2vec) and `char` embeddings for all languages, in Figure 6. As can be seen, the vector distributions vary greatly between the embedding types. Additionally, for all the embedding types, the vectors of different languages occupy a distinct area and are easily separable.

We further investigate the relationships between different embeddings in the following way. We first cluster the values of the `electra` vectors with k-means algorithm. We set the number of clusters to five and assign a different color to each cluster. We retain the `electra` cluster-based color of the samples (glosses) while visualizing the vectors of other embedding types, as shown in Figure 7. It can be clearly seen that the `electra`-based clusters are not preserved for other embedding types.

B.2 Model Performance

Here we present validation and test scores for our six `REVDICT` solutions described in Section 4.2. We use the following metrics for internal validation of our `REVDICT` solutions: *Mean Squared Error* (MSE), *Cosine Similarity* (COS) and *Central Kernel Alignment* (CKA) (Cortes et al., 2012; Kornblith et al., 2019). Validation scores for each `REVDICT` approach can be found in Table 12. The last three rows contain the total scores for each metric and each of our `REVDICT` solutions. A total score is the sum of the values of all datasets and we use it for a simple comparison of solutions. It is evident that each subsequent approach gives better validation results than the previous ones.

Test predictions are scored by these metrics: MSE, COS, and *Cosine-Based Ranking* (RNK). The RNK measure is defined as the proportion of test samples with cosine similarity to the model output embedding higher than the ground truth embedding. The final results for all our solutions can be found in Table 13. Here, each subsequent approach has lower scores than the previous ones, which is the complete opposite

lang	split	vector	min	mean	max	abs-min	abs-mean	abs-max
en	train	sgns	-8.66	0.012	8.33	2.40-08	0.641	8.66
en	train	char	-5.48	0.081	31.10	6.60-09	0.341	31.10
en	train	electra	-126.26	0.033	85.62	1.00-10	0.598	126.26
en	dev	sgns	-7.02	0.013	7.30	9.51-08	0.657	7.30
en	dev	char	-5.48	0.083	7.31	8.75-08	0.341	7.31
en	dev	electra	-48.24	0.028	52.19	1.00-09	0.587	52.19
it	train	sgns	-9.41	-0.014	9.72	6.60-09	0.700	9.72
it	train	char	-13.37	0.013	20.02	1.62-07	0.553	20.02
it	dev	sgns	-8.22	-0.013	7.82	1.02-07	0.706	8.22
it	dev	char	-9.95	0.008	16.23	3.96-07	0.551	16.23
fr	train	sgns	-10.38	-0.013	9.39	1.59-08	0.682	10.38
fr	train	char	-23.42	0.306	11.07	1.12-08	0.574	23.42
fr	train	electra	-46.24	0.045	89.07	3.00-10	0.644	89.07
fr	dev	sgns	-7.57	-0.017	7.81	3.51-07	0.666	7.81
fr	dev	char	-14.60	0.307	7.80	2.62-07	0.574	14.60
fr	dev	electra	-42.73	0.045	51.29	9.00-10	0.655	51.29
es	train	sgns	-9.79	-0.018	9.72	2.15-08	0.653	9.79
es	train	char	-15.03	0.577	13.37	2.27-07	0.822	15.03
es	dev	sgns	-9.32	-0.021	7.22	8.86-08	0.658	9.32
es	dev	char	-13.19	0.577	11.40	2.28-06	0.820	13.19
ru	train	sgns	-7.82	0.002	8.08	1.17-07	0.446	8.08
ru	train	char	-16.87	0.139	8.04	8.00-10	0.311	16.87
ru	train	electra	-30.24	-0.017	22.56	1.75-08	0.788	30.24
ru	dev	sgns	-8.06	0.002	7.91	7.94-08	0.439	8.06
ru	dev	char	-11.86	0.140	8.01	3.05-07	0.310	11.86
ru	dev	electra	-22.53	-0.017	21.70	4.75-08	0.789	22.53

Table 11: Statistics of the elements of the embedding vectors from the train and validation (development) datasets.

of the validation results. This suggests potential overfitting to the dev dataset that could be the result of Bayesian hyperparameter optimization (BHO). However, this is contrary to expectations as the last two solutions have three times fewer BHO points and should not overfit to the dev dataset. The reason for this phenomenon is unclear and needs further investigation.

The best `REVDICT` results for each team can be found in Table 14 for MSE score, Table 15 for COS score, and Table 16 for RNK score. When compared to other solutions, our systems have low to average performance according to the MSE scores. For the COS scores, our systems have very good performance on `sgns` (word2vec) vectors, and low performance on other embedding types. In terms of the RNK (ranking) our systems almost always yield the top performance, and this result is consistent across languages and embedding types.

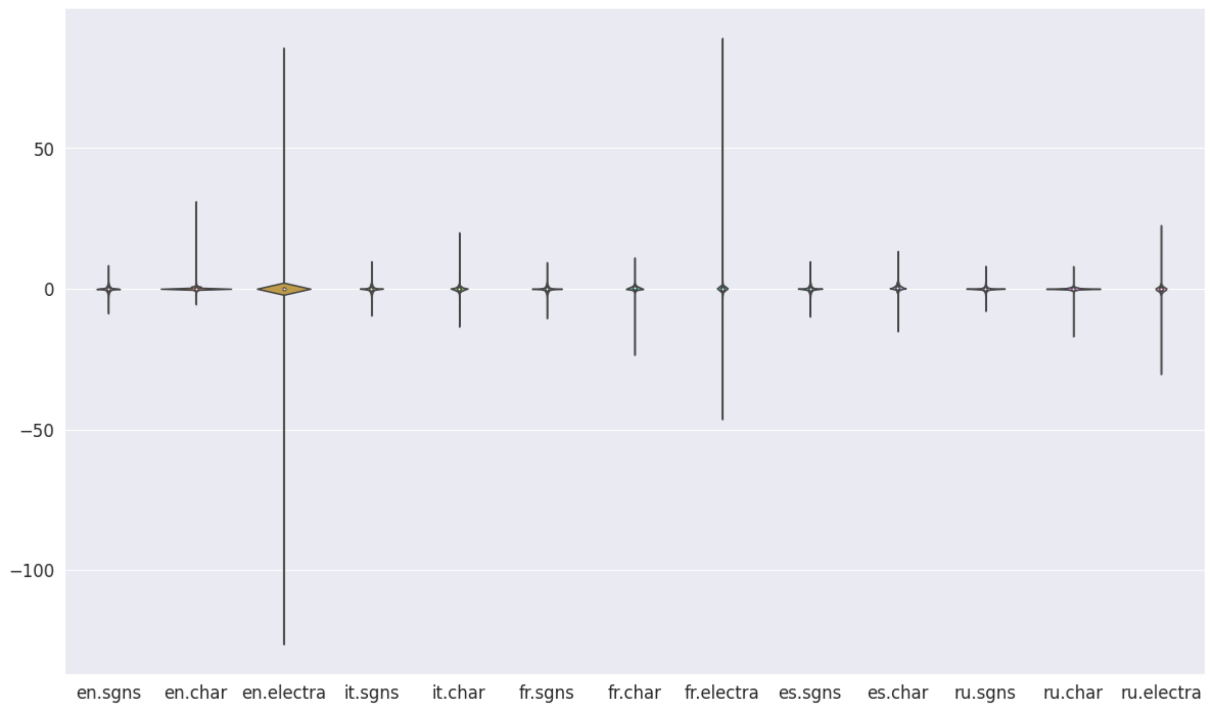


Figure 3: Distributions of vector elements in train datasets.

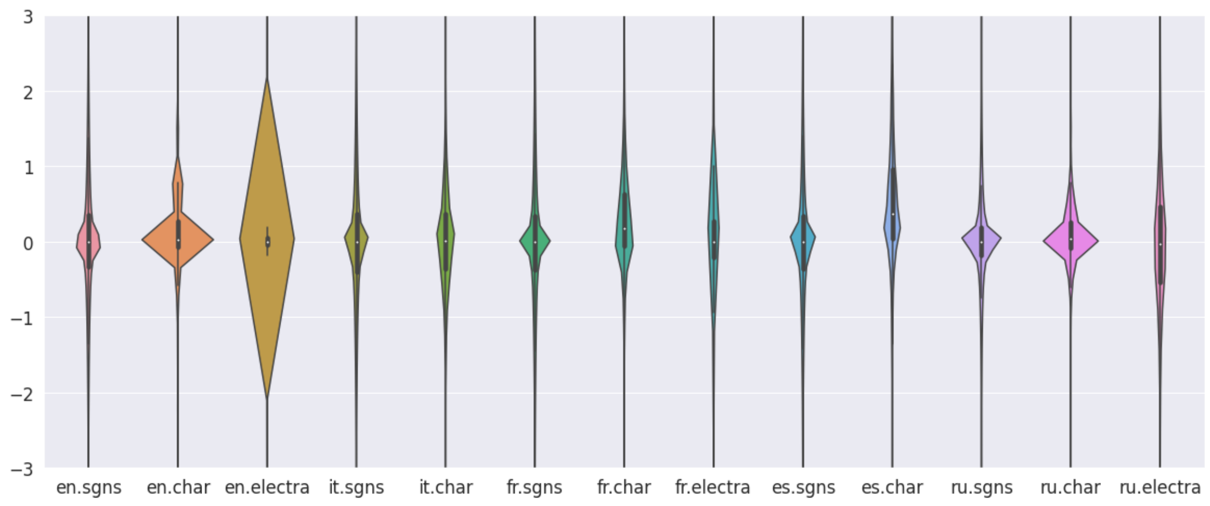


Figure 4: Distributions of vector elements in train datasets within the interval [-3,3].

METRICS	RD 1	RD 2	RD 3	RD 4	RD 5	RD 6	BEST
mse-en-sgns	0.521	0.632	0.521	0.428	0.348	0.343	**0.343**
mse-en-char	0.088	0.091	0.051	0.098	0.058	0.090	**0.051**
mse-en-electra	0.611	0.683	0.612	0.560	0.439	0.295	**0.295**
mse-it-sgns	0.846	0.700	0.783	0.650	0.670	0.485	**0.485**
mse-it-char	0.264	0.235	0.253	0.258	0.224	0.222	**0.222**
mse-fr-sgns	0.773	0.671	0.629	0.585	0.514	0.443	**0.443**
mse-fr-char	0.210	0.205	0.211	0.246	0.180	0.178	**0.178**
mse-fr-electra	0.634	0.616	0.518	0.651	0.400	0.360	**0.360**
mse-es-sgns	0.627	0.764	0.675	0.560	0.543	0.562	**0.543**
mse-es-char	0.341	0.338	0.323	0.353	0.291	0.265	**0.265**
mse-ru-sgns	0.363	0.236	0.268	0.207	0.162	0.109	**0.109**
mse-ru-char	0.053	0.060	0.062	0.060	0.038	0.051	**0.038**
mse-ru-electra	0.544	0.481	0.519	0.497	0.313	0.421	**0.313**
cos-en-sgns	0.483	0.453	0.492	0.537	0.571	0.549	**0.571**
cos-en-char	0.875	0.871	0.927	0.860	0.918	0.873	**0.927**
cos-en-electra	0.895	0.887	0.896	0.900	0.915	0.938	**0.938**
cos-it-sgns	0.470	0.510	0.482	0.527	0.521	0.580	**0.580**
cos-it-char	0.801	0.824	0.810	0.807	0.834	0.836	**0.836**
cos-fr-sgns	0.457	0.490	0.508	0.522	0.544	0.556	**0.556**
cos-fr-char	0.866	0.870	0.866	0.843	0.886	0.887	**0.887**
cos-fr-electra	0.894	0.894	0.904	0.892	0.921	0.929	**0.929**
cos-es-sgns	0.501	0.450	0.486	0.531	0.539	0.538	**0.539**
cos-es-char	0.881	0.883	0.887	0.878	0.899	0.908	**0.908**
cos-ru-sgns	0.525	0.599	0.582	0.618	0.662	0.674	**0.674**
cos-ru-char	0.933	0.925	0.922	0.925	0.952	0.935	**0.952**
cos-ru-electra	0.807	0.821	0.811	0.817	0.872	0.841	**0.872**
cka-en-sgns	0.755	0.776	0.879	0.939	0.889	0.895	**0.939**
cka-en-char	0.991	0.994	0.998	0.996	0.996	0.994	**0.998**
cka-en-electra	0.993	0.995	0.997	0.998	0.997	0.998	**0.998**
cka-it-sgns	0.608	0.773	0.811	0.905	0.780	0.857	**0.905**
cka-it-char	0.979	0.989	0.992	0.993	0.989	0.990	**0.993**
cka-fr-sgns	0.647	0.779	0.860	0.916	0.838	0.866	**0.916**
cka-fr-char	0.981	0.989	0.993	0.991	0.990	0.991	**0.993**
cka-fr-electra	0.991	0.995	0.997	0.996	0.997	0.997	**0.997**
cka-es-sgns	0.685	0.698	0.826	0.909	0.802	0.795	**0.909**
cka-es-char	0.982	0.989	0.993	0.992	0.991	0.992	**0.993**
cka-ru-sgns	0.611	0.821	0.850	0.927	0.880	0.927	**0.927**
cka-ru-char	0.995	0.997	0.998	0.998	0.997	0.997	**0.998**
cka-ru-electra	0.978	0.989	0.992	0.994	0.994	0.991	**0.994**
TOTAL mse	**5.875**	**5.712**	**5.425**	**5.153**	**4.180**	**3.824**	**3.824**
TOTAL cos	**9.388**	**9.477**	**9.573**	**9.657**	**10.034**	**10.044**	**10.044**
TOTAL cka	**11.196**	**11.784**	**12.186**	**12.554**	**12.140**	**12.290**	**12.554**

Table 12: Validation scores for all our `REVDICT` (RD) approaches. For each score, comparative results are shown in color. Green is used for the best and red for the worst-performing solution per row (a metric defines whether higher or lower values are better). The total score is the sum of the values over all datasets and embeddings.

METRICS	RD 1	RD 2	RD 3	RD 4	RD 5	RD 6	BEST
mse-en-sgns	1.024	0.964	1.021	1.085	1.170	1.119	**0.964**
mse-en-char	0.169	0.169	0.186	0.162	0.195	0.172	**0.162**
mse-en-electra	1.723	1.685	1.690	1.768	1.863	1.988	**1.685**
mse-it-sgns	1.076	1.160	1.100	1.156	1.211	1.318	**1.076**
mse-it-char	0.366	0.383	0.376	0.370	0.399	0.399	**0.366**
mse-fr-sgns	1.068	1.119	1.134	1.147	1.250	1.319	**1.068**
mse-fr-char	0.409	0.419	0.418	0.390	0.447	0.434	**0.390**
mse-fr-electra	1.339	1.347	1.414	1.358	1.554	1.566	**1.339**
mse-es-sgns	0.941	0.883	0.924	0.965	1.031	1.020	**0.883**
mse-es-char	0.526	0.545	0.546	0.532	0.582	0.635	**0.526**
mse-ru-sgns	0.568	0.604	0.596	0.601	0.667	0.653	**0.568**
mse-ru-char	0.145	0.141	0.142	0.140	0.170	0.144	**0.140**
mse-ru-electra	0.911	0.944	0.956	0.961	1.105	1.049	**0.911**
cos-en-sgns	0.250	0.260	0.250	0.245	0.231	0.214	**0.260**
cos-en-char	0.761	0.761	0.743	0.770	0.734	0.765	**0.770**
cos-en-electra	0.821	0.828	0.824	0.818	0.812	0.792	**0.828**
cos-it-sgns	0.380	0.358	0.370	0.361	0.361	0.339	**0.380**
cos-it-char	0.724	0.713	0.717	0.721	0.709	0.711	**0.724**
cos-fr-sgns	0.342	0.336	0.333	0.330	0.319	0.255	**0.342**
cos-fr-char	0.744	0.738	0.739	0.756	0.725	0.734	**0.756**
cos-fr-electra	0.847	0.842	0.837	0.844	0.828	0.825	**0.847**
cos-es-sgns	0.362	0.367	0.361	0.349	0.350	0.354	**0.367**
cos-es-char	0.819	0.812	0.812	0.816	0.803	0.784	**0.819**
cos-ru-sgns	0.412	0.421	0.411	0.406	0.399	0.381	**0.421**
cos-ru-char	0.818	0.822	0.820	0.824	0.788	0.821	**0.824**
cos-ru-electra	0.724	0.712	0.715	0.712	0.683	0.702	**0.724**
rnk-en-sgns	0.247	0.234	0.246	0.231	0.252	0.262	**0.231**
rnk-en-char	0.438	0.439	0.419	0.448	0.438	0.444	**0.419**
rnk-en-electra	0.438	0.446	0.437	0.444	0.438	0.432	**0.432**
rnk-it-sgns	0.165	0.177	0.178	0.169	0.188	0.187	**0.165**
rnk-it-char	0.397	0.390	0.400	0.402	0.397	0.383	**0.383**
rnk-fr-sgns	0.214	0.203	0.212	0.193	0.229	0.262	**0.193**
rnk-fr-char	0.425	0.429	0.427	0.435	0.431	0.421	**0.421**
rnk-fr-electra	0.447	0.463	0.448	0.450	0.444	0.429	**0.429**
rnk-es-sgns	0.197	0.214	0.203	0.199	0.217	0.201	**0.197**
rnk-es-char	0.407	0.409	0.403	0.412	0.407	0.420	**0.403**
rnk-ru-sgns	0.161	0.153	0.175	0.154	0.166	0.150	**0.150**
rnk-ru-char	0.361	0.365	0.376	0.372	0.378	0.357	**0.357**
rnk-ru-electra	0.350	0.355	0.351	0.359	0.351	0.345	**0.345**
TOTAL mse	**10.266**	**10.363**	**10.504**	**10.634**	**11.645**	**11.817**	10.266
TOTAL cos	**8.004**	**7.971**	**7.931**	**7.951**	**7.742**	**7.677**	8.004
TOTAL rnk	**4.248**	**4.276**	**4.275**	**4.268**	**4.337**	**4.293**	4.248

Table 13: Test results for all our `REVDICT` (RD) approaches. For each score, comparative results are shown in color. Green is used for the best and red for the worst-performing solution per row (a metric defines whether higher or lower values are better). The total score is the sum of the values over all datasets and embeddings.

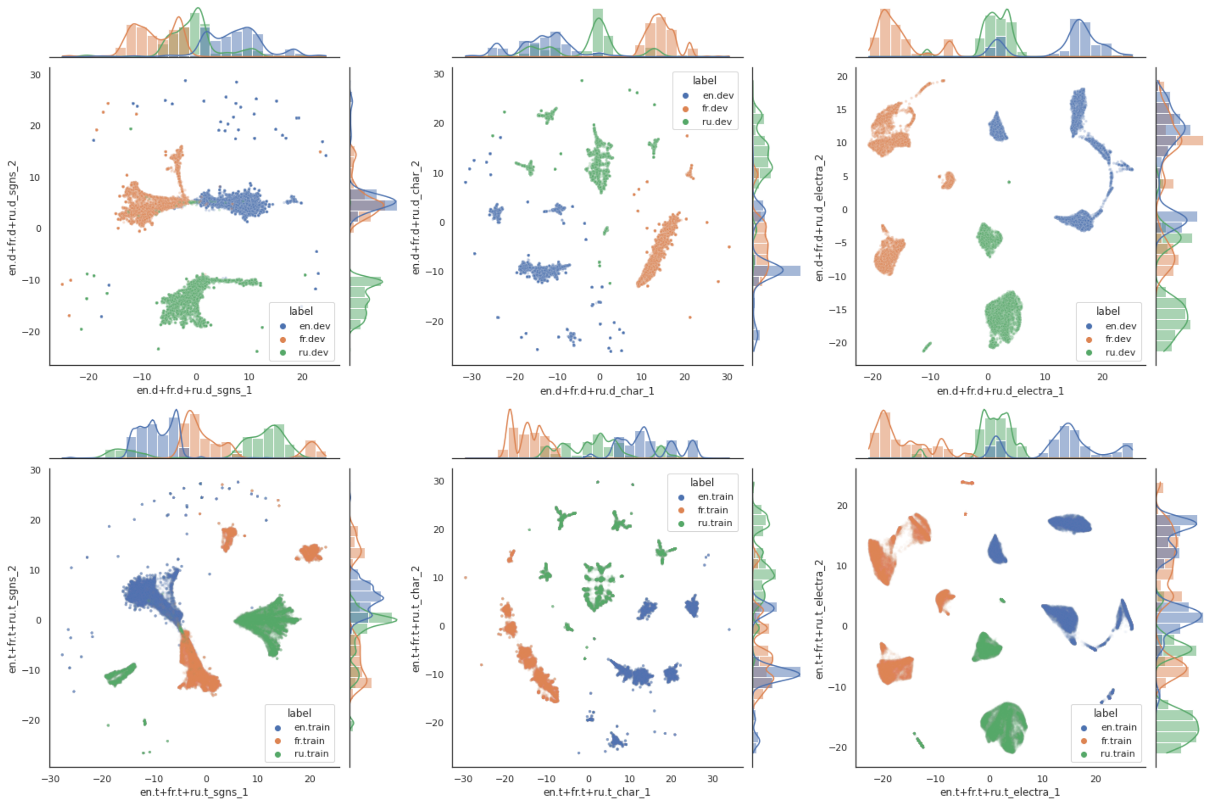


Figure 5: Distributions of all three embedding types in train (2nd row) and validation (development, 1st row) datasets after dimensionality reduction to 2D space. `sgns` (word2vec, 1st column), `char` (2nd column), and `electra` (3rd column) embeddings are depicted for English (orange), French (green), and Russian (blue).

TEAM	EN			ES		FR			IT		RU		
	sgns	char	electra	sgns	char	sgns	char	electra	sgns	char	sgns	char	electra
0	0.909			0.913		1.122			1.196		0.615		
1	**0.964**	**0.162**	**1.685**	**0.883**	**0.526**	**1.068**	**0.390**	**1.339**	**1.076**	**0.366**	**0.568**	**0.140**	**0.911**
2				0.911									
3	0.854												
5	0.864	0.143	1.310	0.860	0.467	1.026	0.335	1.066	1.031	0.334	0.538	0.116	0.828
6	0.900	0.143	1.340										
7	0.915	0.168		0.906	0.557	1.100	0.391		1.097	0.364	0.578	0.156	
10	0.875	0.141	1.301						1.087	0.355			
12	0.895	0.143	1.326	0.910	0.510	1.107	0.366	1.112	1.111	0.359	0.566	0.132	0.864
13	0.862	0.176	1.509	0.858	0.583	1.030	0.411	1.271	1.039	0.438	0.528	0.184	0.828

Table 14: MSE test scores for each team in `REVDICT` task. The results of our team are bold (team 1). For each task, comparative results are shown in color. Green is used for the best and red for the worst-performing solution per column.

TEAM	**EN**			**ES**		**FR**			**IT**		**RU**		
	sgns	**char**	**electra**	**sgns**	**char**	**sgns**	**char**	**electra**	**sgns**	**char**	**sgns**	**char**	**electra**
0	0.156			0.223		0.216			-0.004		0.006		
1	**0.260**	**0.770**	**0.828**	**0.367**	**0.819**	**0.342**	**0.756**	**0.847**	**0.380**	**0.724**	**0.421**	**0.824**	**0.724**
2				0.403									
3	0.248												
5	0.241	0.795	0.847	0.347	0.839	0.312	0.789	0.862	0.374	0.747	0.383	0.852	0.735
6	0.185	0.796	0.846										
7	0.194	0.792		0.262	0.820	0.228	0.769		0.260	0.739	0.335	0.836	
10	0.204	0.798	0.843						0.274	0.734			
12	0.166	0.795	0.844	0.252	0.824	0.212	0.770	0.858	0.246	0.728	0.298	0.830	0.721
13	0.243	0.782	0.846	0.353	0.824	0.328	0.752	0.859	0.360	0.681	0.424	0.791	0.734

Table 15: COS test scores for each team in `REVDICT` task. The results of our team are bold (team 1). For each task, comparative results are shown in color. Green is used for the best and red for the worst-performing solution per column.

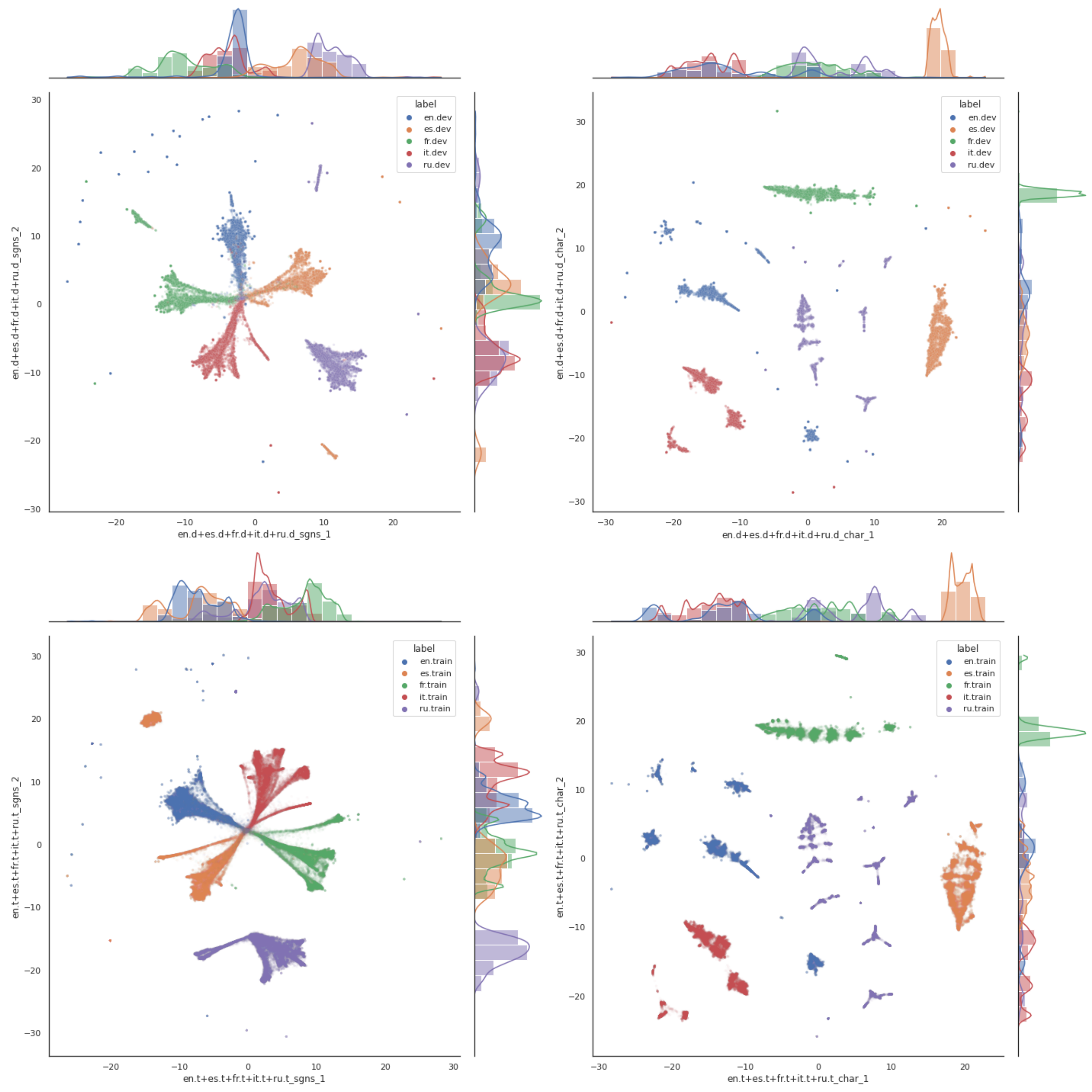


Figure 6: Distributions of `sgns` (1st column) and `char` (2nd column) embeddings in train (2nd row) and validation (development, 1st row) datasets after dimensionality reduction to 2D space. The embeddings are depicted for English (blue), Spanish (orange), French (green), Italian (red), and Russian (violet).

TEAM	**EN**			**ES**		**FR**			**IT**		**RU**		
	sgns	**char**	**electra**	**sgns**	**char**	**sgns**	**char**	**electra**	**sgns**	**char**	**sgns**	**char**	**electra**
0	0.499			0.495		0.498			0.499		0.499		
1	**0.231**	**0.419**	**0.432**	**0.197**	**0.403**	**0.193**	**0.421**	**0.429**	**0.165**	**0.383**	**0.150**	**0.357**	**0.345**
2				0.167									
3	0.319												
5	0.326	0.500	0.490	0.271	0.424	0.302	0.428	0.476	0.197	0.428	0.247	0.389	0.417
6	0.500	0.500	0.500										
7	0.374	0.478		0.375	0.410	0.439	0.416		0.384	0.438	0.291	0.377	
10	0.394	0.483	0.478						0.386	0.478			
12	0.312	0.450	0.434	0.253	0.412	0.314	0.428	0.442	0.247	0.417	0.290	0.410	0.399
13	0.329	0.486	0.478	0.251	0.500	0.282	0.502	0.478	0.230	0.496	0.187	0.472	0.420

Table 16: RNK test scores for each team in `REVDICT` task. The results of our team are bold (team 1). For each task, comparative results are shown in color. Green is used for the best and red for the worst performing solution per column.

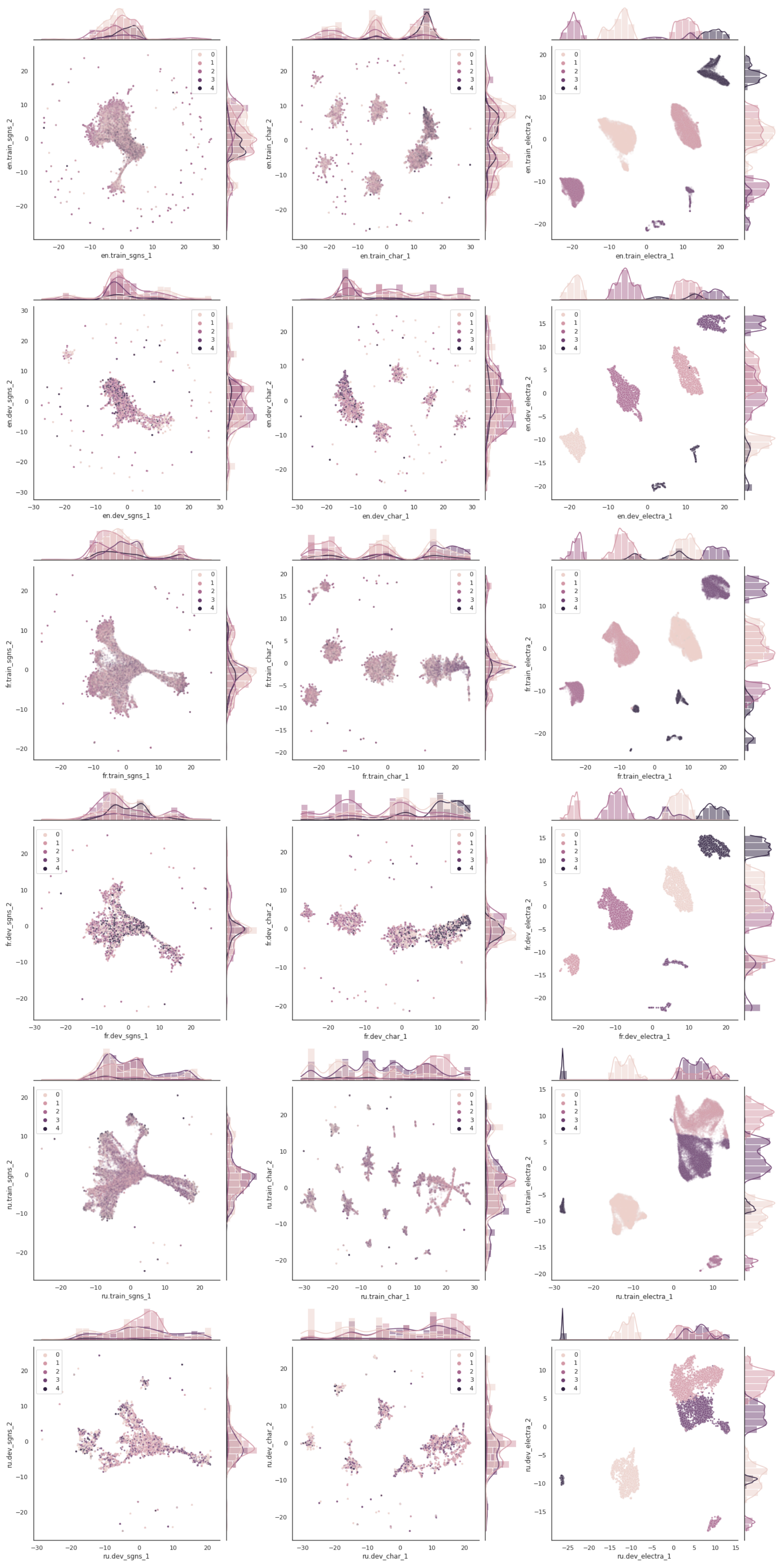


Figure 7: Projection of the clusters in the `electra` embedding space (3rd column) to the spaces of the other two embedding types `sgns` (1st column) and `char` (2nd column). The analysis if performed for English (rows 1–2), French (rows 3–4), and Russian (rows 5–6) train and validation (development) datasets, after dimensionality reduction to 2D space.

TLDR at SemEval-2022 Task 1: Using Transformers to Learn Dictionaries and Representations

V. Harsha Vardhan[†]
IIIT Hyderabad
harshavardhan.v
@research.iiit.ac.in

Aditya Srivastava[†]
IIIT Hyderabad
aditya.srivastava
@research.iiit.ac.in

Abstract

We propose a pair of deep learning models, which employ unsupervised pretraining, attention mechanisms and contrastive learning for representation learning from dictionary definitions, and definition modeling from such representations. Our systems, the **T**ransformers for **L**earning **D**ictionaries and **R**epresentations (**TLDR**), were submitted to the *SemEval 2022 Task 1: Comparing Dictionaries and Word Embeddings (CODWOE)*, where they officially ranked **first on the definition modeling subtask**, and achieved competitive performance on the reverse dictionary subtask. In this paper we describe our methodology and analyse our system design hypotheses.

1 Introduction

Dictionaries are some of the linguistically richest resources available for a language, in addition to being extremely clean and unbiased in comparison to most naturally occurring language data, which is noisy and shows domain specific bias based on its source. Thus, there has been considerable interest towards using NLP models to harness this knowledge, especially for low-resource languages. Broadly there are two sets of approaches towards the same - the first is to use dictionaries for representation learning and using these representations for transfer learning in other tasks, such as in the work by Bosc and Vincent (2018) where they use an LSTM based auto-encoder to learn rich representations from dictionary definitions such that the definitions can also be generated back from the representations. Tissier et al. (2017) also used dictionary definitions to build sets of 'strong' and 'weak' pairs of words to get improved word representations with greater interpretability by moving words which show a stronger semantic-relatedness closer together in the embedding space.

[†] *Authors contribute equally to this work.*

The second approach has been to move in the opposite direction, such as in the work by Chang and Chen (2019) where the authors try to map contextualized word representations to their dictionary definitions in an effort towards word-sense disambiguation. This has been further explored by Noraset et al. (2016), where they use an RNN based language model to generate definitions for representations. Recent work by Bevilacqua et al. (2020) improves upon this by leveraging pre-trained encoder-decoder models like BART (Lewis et al., 2019) in order to generate the definitions of words. These pre-trained language models, significantly outperform the RNN based models. The ability to generate definitions for representations makes these contextual representations of words explainable.

NLP has advanced leaps and bounds within the past decade, with a major push coming from the advent and utilization of transfer learning via representation learning techniques such as Word2Vec (Mikolov et al., 2013a) and ELMO (Peters et al., 2018). The more recent methods to employ transfer learning use large pretrained language models, such as BERT (Devlin et al., 2019) and XLM (Conneau et al., 2020). These models are jointly used with unsupervised training objectives such as MLM and Causal-LM, to transform natural language into information rich meaning representations which are then used for many different downstream tasks. We aim to replicate the same in our experiments to give a better prior for the model, before it learns to generate desired representations.

One of the characteristics common to all the language models mentioned above is that they are all based on transformers. Transformers use the multi-headed attention and self-attention mechanisms to learn extremely effective language representations. Transformers also scale well since unlike their predecessors, the RNNs, they process information in parallel and thus are much faster.

For SemEval 2022's Task 1, Comparing Dictionaries and Word Embeddings (Mickus et al., 2022), the participants were asked to design systems for the following two subtasks;

1. Subtask 1: Reconstruct SGNS (Mikolov et al., 2013b), character and ELECTRA (Clark et al., 2020) embeddings from their dictionary glosses.
2. Subtask 2: Reconstruct the dictionary glosses from their SGNS, character and ELECTRA embeddings.

The subtasks are called the Reverse Dictionary and Definition Modeling subtasks respectively. The first subtask is evaluated on the Mean Squared Error (MSE), Cosine Similarity and Cosine Ranking between the generated representations and the gold representations. The second subtask is evaluated on Mover score, Sense level BLEU score (S-BLEU) and Lemma level BLEU score (L-BLEU).

In this system description paper we detail our model architectures, training, evaluation and testing methodologies, and try to analyse our hypotheses and their impact on the final scores. For the reverse dictionary subtask we designed a simple BERT-like model, pretrained it on the MLM objective, and finetuned it for the subtask on a combination of cosine embedding loss and MSE loss, alongside negative sampling of the dataset to add a contrastive loss to the overall objective function. For the definition modeling subtask we designed a model based on transformer decoders for natural language generation, with masked self-attention over the inputs in addition to multi-head attention over the representations from the three embeddings spaces. We submitted our systems for evaluation over English data and our systems demonstrated very good performance in the contest itself, with the definition model system outperforming all other submissions and taking first place, and the the reverse dictionary system achieving the fifth, sixth and seventh place on the character, ELECTRA and SGNS targets respectively. Lastly, we also show some post-contest improvements on the reverse dictionary system. The code for our experiments has been open sourced and is available on GitHub.[1]

2 Subtask 1: Reverse Dictionary

2.1 Data Preprocessing

We maintain the data splits provided by the task organizers (43608 training samples, 6375 dev samples and 6221 test samples) with each sample containing the dictionary gloss and its SGNS, character and ELECTRA representations. All models were trained on the training split, with the best model picked from evaluation over the dev split.

The dictionary glosses were lower-cased and stripped of all whitespace characters except those essential to maintaining word boundaries for tokenization. We also padded and truncated all sequences to a maximum sequence length of 256 tokens.

For contrastive learning we performed negative sampling to augment each gloss-embedding pair with three other embeddings from the same semantic space and the same data split.

2.2 System Overview

We designed our system around three hypotheses - firstly, using pretraining would work better than starting from scratch since it would give the model a good prior for the downstream tasks (transfer learning). Secondly, training individual models for each representation would outperform training a single multitask model, as the representations do not reside in a common semantic space. Thirdly, optimizing over a combination of losses for each of the metrics that we're being evaluated upon, i.e. MSE loss for MSE, Cosine Embedding (CE) loss for cosine similarity, contrastive loss for cosine-rank, would work better than using any of them individually.

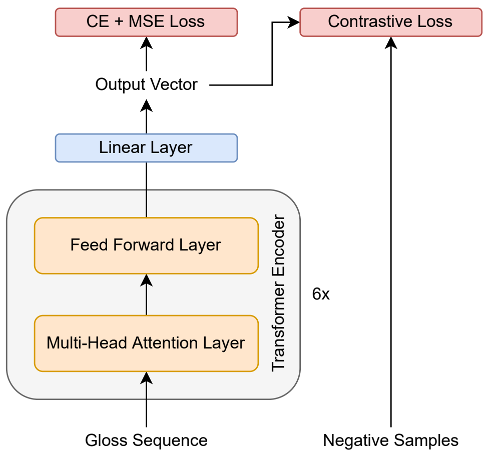


Figure 1: System architecture for the Reverse Dictionary subtask.

The foundation of our representation learner is based on the DistilBert (Sanh et al., 2019) architecture and comprises of a stack of 6 transformer

[1]https://www.github.com/IamAdiSri/tldr-semeval22

encoders, with 12 attention heads each, hidden dimension of 3072 and embedding dimension of 768.

We start by pretraining the model via unsupervised masked-language-modeling (Devlin et al., 2018) over only the texts from the dictionary glosses in the task dataset. Individual instances of this pretrained model are then appended with a linear layer of dimension 256 to project outputs in the required dimensions, and fine-tuned for each semantic space, optimized over the following loss function;

$$
\begin{aligned}
L = e^{-\log(p_0)} * \mathit{MSE}(\phi(g), v_p) + \\
e^{-\log(p_1)} * \mathit{CE}(\phi(g), v_p) + \\
e^{-\log(p_2)} * \mathit{CL}(\phi(g), v_{n0}, v_{n1}, v_{n2})
\end{aligned} \tag{1}
$$

where, $\phi(g)$ is the sentence embedding for a gloss g, v_p is the true (or *positive*) embedding, v_{n0}, v_{n1} and v_{n2} are the negative samples and p_0, p_1 and p_2 are trainable parameters for weighting the different loss functions. **MSE** and **CE** are the **Mean Squared Error** and **Cosine Embedding Loss** respectively, which function as the reconstruction loss between the generated representation and true representation. Lastly, **CL** is the **Contrastive Loss** between the generated representations and the false representations from negative sampling. The equations for the three are given below;

$$\mathit{MSE}(a, b) = \frac{1}{d}\sum_{i=0}^{d}(a_i - b_i)^2 \tag{2}$$

$$\mathit{CE}(a, b) = 1 - \frac{a.b}{|a||b|} \tag{3}$$

$$\mathit{CL}(p, n_0, n_1, n_2) = \sum_{i=0}^{2} 1 - \mathit{CE}(p, n_i) \tag{4}$$

where a, b, p and n are all vectors of size d.

2.3 Experimental Setup

We used Pytorch (Paszke et al., 2019) and the HuggingFace (Wolf et al., 2019) library to write our experiments in Python. Though the HuggingFace library does provide ready-to-go pretrained language models, they were not used in our experiments or submissions. We did however use a ready-made pretrained tokenizer from the library to tokenize our texts.[2] This was permitted by the organizers and we believed would be a significant advantage over training a new tokenizer from scratch, considering that the dataset is rather small and homogeneous in its linguistic variation. All models were trained on Nvidia's GTX-1080 Ti and RTX-2080 Ti GPUs.

We pretrained our model on the unsupervised MLM objective, with a learning rate (LR) of $5e-5$, masking probability of 0.15, and an effective batch size of 64 samples. For fine-tuning we use an LR of $5e-3$ with a linear LR scheduler and an effective batch size of 64 samples per batch. Any remaining hyperparameters were left as their default values. We selected the language model that attained the lowest perplexity score (55.25) over the dev split.

For models that were multitask trained, instead of a single output layer (as shown in figure 1) we have three output layers (one for projecting into each semantic space), all receiving the same output vector from the transformer encoders below them.

While finetuning we train all models for 50 epochs in total, and select a model for each semantic space and metric based on the epoch that attains the best score over the dev split. The same model is used for producing results over the test split.

2.4 Results and Analysis

The experiments here were designed with the intent to evaluate the three modeling hypotheses outlined at the beginning of section 2.2, with tests on pretraining, mutitasking and the objective function. The results for these experiments are given in tables 1, 3 and 4.

Repr.	Metric	Pretrained	RndInit.
sgns	mse	0.8990793	**0.8987827**
sgns	cos	**0.1805207**	0.1799421
sgns	rnk	**0.5004269**	0.5004292
char	mse	0.1465276	**0.1454727**
char	cos	0.7897581	**0.7916019**
char	rnk	**0.5004282**	0.5004290
electra	mse	1.5150244	**1.3510013**
electra	cos	**0.8452746**	0.8455241
electra	rnk	**0.5000801**	0.5000807

Table 1: Comparison between using pretraining versus starting from a randomly initialized model.

As we can see from table 1, there is no significant difference (mostly under $1e-4$) between

[2] https://huggingface.co/distilbert-base-uncased

Repr.	MSE	CSim.	Rank	PreTr.	MltTsk.	Objs.
sgns	558.96838	**0.18531**	0.50043			CE
sgns	**0.89953**	0.17866	**0.50043**	✓	✓	MSE, CE, CL
char.	125.12206	**0.79565**	0.50043			CE
char.	**0.14337**	0.79560	**0.50043**	✓	✓	MSE, CE, CL
electra	49.98565	**0.84564**	0.50008			CE
electra	**1.34014**	0.84563	**0.50008**	✓	✓	MSE, CE, CL

Table 2: Best submissions on the Reverse Dictionary leaderboard.

finetuning a pretrained model versus training the model from scratch. We did however notice that our pretrained models seemed to converge in fewer epochs than when the models were trained from scratch, indicating that the pretraining did have some positive effect.

Repr.	Metric	Individual	Multitask
sgns	mse	**0.8984921**	0.8990793
sgns	cos	0.1786035	**0.1805207**
sgns	rnk	0.5004290	**0.5004269**
char	mse	**0.1431219**	0.1465276
char	cos	**0.7955332**	0.7897581
char	rnk	0.5004292	**0.5004282**
electra	mse	**1.3292400**	1.5150244
electra	cos	0.8451633	**0.8452746**
electra	rnk	**0.5000672**	0.5000801

Table 3: Comparison between individual models for each semantic space versus a single multitask model.

Table 3 displays the results of our tests comparing a single multitask model for all three semantic spaces versus training individual models for each. Again the results are quite inconclusive whether one reliably outperforms the other, however considering differences of over $1e-3$ to be significant (as they affect the leaderboard rankings) we can see that individual models outperform multitask models on a greater number of metrics.

Finally, from the ablation tests in table 4 we can clearly see that while optimizing over the combination of losses or MSE alone gives comparable scores, optimizing over CE loss alone causes the MSE score to worsen manyfold. This seems to imply that MSE contributes significantly more than CE and CL losses to the performance of the models. The scores also demonstrate that tuning over MSE tends to improve the MSE metric, while tuning over CE improves the cosine similarity, agreeing with our hypothesis about the same. Contrary to our expectations however, we can also observe that adding contrastive learning by negative sampling does not improve the cosine-ranking by much.

3 Subtask 2: Definition Modeling

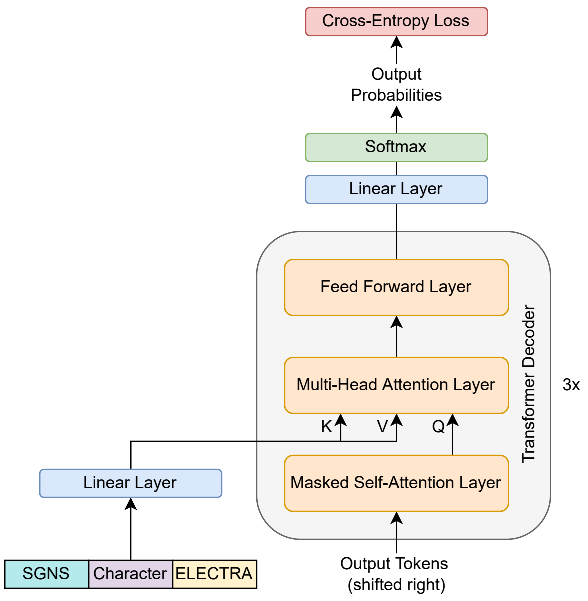


Figure 2: System architecture for the definition modeling subtask.

3.1 Data Preprocessing

The dictionary glosses were first normalized for punctuation and then tokenized using the Moses scripts (Koehn et al., 2007). We then learn a subword tokenizer on the training data, in order to create a fixed vocabulary of subwords. The Moses tokenized sentences were used to learn Byte Pair Encodings (BPE) with a vocabulary size of 10,000 using the subword-nmt library (Sennrich et al., 2016).

Repr.	Metric	All Losses	Only MSE	Only CE
sgns	mse	0.8991734	**0.8956623**	694685.50
sgns	cos	0.1787637	0.1547712	**0.1852663**
sgns	rnk	**0.5004290**	**0.5004290**	0.5004290
char	mse	**0.1430358**	0.6462155	169896.47
char	cos	0.7955229	0.3045915	**0.7955511**
char	rnk	**0.5004290**	0.5004294	**0.5004290**
electra	mse	**1.3285819**	3.5256984	82728.45
electra	cos	0.8451307	0.0650083	**0.8453690**
electra	rnk	0.5000807	**0.5000807**	0.5000808

Table 4: Ablation tests on the reverse dictionary objective function.

3.2 System Overview

The definition modeling problem is posed as one of text generation, generating the definition D_* of a word w_* autoregressively, given the semantic representation of the word in the form a vector v_*. Therefore, we maximise the likelihood of the definition $D_* = \{w_0, w_1, .., w_n\}$, where w_i corresponds to the ith word of the definition, given the vector v_*.

$$P(D_*/v_*) = \prod_{i=0}^{n} P(w_i/w_0, ..w_{i-1}, v_*) \tag{5}$$

The definition modeling architecture that has been used in this system is a transformer decoder, whose output softmax layer approximates the above likelihood.

In a vanilla transformer-seq2seq architecture (Vaswani et al., 2017), the decoder in the self-attention layer projects the decoder states into three matrices called Query (Q), Key (K) and Value (V) and using the below equation, attention values are computed.

$$\mathit{Attn.}(Q, K, V) = \mathit{Softmax}\left(\frac{Q \cdot K^T}{\sqrt{d}}\right) \cdot V \tag{6}$$

This is followed by a cross-attention layer where the decoder attends to the encoder states by projecting the encoder states as the Keys, Values and using the decoder states as the Queries.

In our model, since the input vectors are not in the same space as the decoder embeddings, a linear layer is used to learn a projection between them. The output of this layer is then projected into the Key (K) and Value (V) matrices which are used along with the decoder self-attention outputs to compute the cross attention values.

$$f = \mathit{linear}(v_{sgns} \oplus v_{char} \oplus v_{electra}) \tag{7}$$

where v_{sgns}, v_{char}, $v_{electra}$ represent the SGNS, character, ELECTRA vectors respectively and $\oplus$ denotes concatenation.

$$\begin{aligned} K &= f \cdot W^K \\ V &= f \cdot W^V \\ Q &= o \cdot W^Q \end{aligned} \tag{8}$$

where W^K, W^V are the matrices learnt to project the output of linear layer (f) to Key, Value matrices and W^Q is the matrix to project the output of the decoder self-attention layer o to the Query matrix.

3.3 Experimental Setup

The experiments for the defmod subtask were carried out using the Fairseq framework (Ott et al., 2019) in Python. All the models were trained on Nvidia GTX 1080 Ti GPUs.

The transformer decoder model consists of 3 decoder layers, with each layers consisting of 8 attention heads and an embedding dimension of 512 with the input and the output embeddings of the decoder being tied. The model is trained using label smoothed cross entropy loss with label smoothing of 0.2. A learning rate of 5e-4 using an inverse square root scheduler with a weight decay of 0.0001 was used to optimize over a batch size of 4096 tokens.

The model was trained for over 50 epochs and the checkpoint with the least validation perplexity

Word	Gloss	Predicted Gloss
farming	A farming operation ; a farm , or instance of farming on a piece of land .	The act or process of producing food
hazard	The chance of suffering harm ; danger , peril , risk of loss .	Something that causes trouble or destruction
hardware	Equipment .	Of or relating to a computer system
transition	The process of change from one form , state , style or place to another .	The act or process of converting
hastily	In a hasty manner ; quickly or hurriedly	In a hurried manner
chesty	Not dry ; involving the coughing of phlegm .	Of, pertaining to, or characteristic of a dove
Solarian	Of or relating to the Solar System .	Of or pertaining to the planet Mars.
valid	Well grounded or justifiable , pertinent .	Not able to be true.
alt-left	The extreme or radical left of the political spectrum .	A member of the United States of the United States of the United States of the United States of the United States.

Table 5: Examples of generated glosses.

was selected to generate definitions for the test set using beam search with a beam size of 10.

3.4 Results and Analysis

The ablation study results in table 6 shows the impact of each vector, namely SGNS, char and ELECTRA, against the best performing model where features were extracted using a concatenation of all three.

Repr.	Mover Score	S-BLEU	L-BLEU
All	**0.12847**	**0.03278**	**0.04250**
Electra	0.11008	0.02957	0.03629
Char	0.10403	0.02884	0.03643
SGNS	0.03622	0.01743	0.02114

Table 6: DefMod scores using all vs individual representations

We can clearly see that the ELECTRA representations outperform char and SGNS, with the SGNS vectors falling significantly behind on all three metrics. It can also be observed that by concatenating all three representations and extracting useful features using attention significantly boosts performance. This allows one to infer that char and SGNS vectors do contain semantic information that ELECTRA does not.

The submission utilizing all representations outperformed all other submissions in the defmod subtask and ranked first for English.

3.4.1 Observations Post Dataset Release

After the passing of the task deadline, the organizers released the full dataset, complete with all annotations. With this data we were able to make further observations about our model by comparing the generated glosses to the actual glosses in the dataset. A few examples are shown in table 5. Examples marked in red indicate wrong or irrelevant definitions, and the ones marked in green describe the relevant ones.

From the generated glosses in green, we can see that the model shows a remarkably good mapping between the words and their representations, and makes close approximations of their meanings when generating glosses. In the example of *farming*, we can see that the model is able to correctly associate the act of farming with that of producing food. The words *hazard* and *transition* are also correctly associated with *trouble (danger)* and *conversion (change)* respectively, demonstrating that the model is able to recall similar concepts. The model shows good language proficiency as well, with syntactically correct utterances.

In case of words like *hardware*, which have multiple meanings, we can see that the model outputs one of the secondary definitions that it has learnt from the dataset. From this, we can infer that in the absence of appropriate context (i.e. how the word is being used in a sentence), the model has difficulty in disambiguating the word's meaning, although it still recognizes the word and picks one of the correct definitions.

Finally, the model tends to learn definition templates from the training data, such as *"Of, pertaining to"* or *"Of or relating"*, that it reuses during generation. As a result of being a sequence to sequence model, it also occasionally exhibits degenerate repetition, as seen in the *"alt-left"* example.

4 Conclusion

In this paper, we explored a variety of approaches towards dictionary definitions and embeddings generation, especially under the constraint of being unable to use external monolingual data. We have analyzed the effectiveness of each of these methods, performing ablation studies showing the impact that various objective functions like MSE, cosine embedding loss and contrastive loss have in reconstructing representations from text, and coming up with an attention mechanism utilizing all the provided representations to generate definitions to produce exceptional results.

As part of future work, we plan to explore the performance of our models in multilingual settings. We would also use test the performance of these models against pre-trained language models like BART, BERT etc. to gauge the impact language model pre-trainig since we did not have enough monolingual data in the tasks to train them. Finally, we plan to experiment with a joint training mechanism where instead of training on each subtask independently, the models can inform and improve each other by collaboratively learning both the subtasks.

References

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221, Online. Association for Computational Linguistics.

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1532, Brussels, Belgium. Association for Computational Linguistics.

Ting-Yun Chang and Yun-Nung Chen. 2019. What does this word mean? explaining contextualized embeddings with natural language definition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6064–6070, Hong Kong, China. Association for Computational Linguistics.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, page 177–180, USA. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Efficient estimation of word representations in vector space.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2016. Definition modeling: Learning to define word embeddings in natural language.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Julien Tissier, Christophe Gravier, and Amaury Habrard. 2017. Dict2vec : Learning word embeddings using lexical dictionaries. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Copenhagen, Denmark. Association for Computational Linguistics.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing.

MMG at SemEval-2022 Task 1: A Reverse Dictionary approach based on a review of the dataset from a lexicographic perspective

Alfonso Ardoiz[1,2]
alfonso.ardoiz@dezzai.com
aardoiz@ucm.es

Miguel Ortega-Martín[1,2]
m.ortega@dezzai.com
m.ortega@ucm.es

Óscar García-Sierra[1,2]
oscar.garcia@dezzai.com
oscarg02@ucm.es

Jorge Álvarez[1]
jorge.alvarez@dezzai.com

Ignacio Arranz[1]
ignacio.arranz@dezzai.com

Adrián Alonso[1,3,4]
a.alonso@dezzai.com
adrian.barriuso@urjc.es

1 dezzai by MMG **2** Universidad Complutense de Madrid
3 Universidad Rey Juan Carlos **4** Data Science Laboratory - Universidad Rey Juan Carlos

Abstract

This paper presents a novel and linguistic-driven system for the Spanish Reverse Dictionary task of SemEval-2022 Task 1. The aim of this task is the automatic generation of a word using its gloss. The conclusion is that this task results could improve if the quality of the dataset did as well by incorporating high-quality lexicographic data. Therefore, in this paper we analyze the main gaps in the proposed dataset and describe how these limitations could be tackled.

1 Introduction

The CODWOE (Comparison of Word Glosses and Word Embeddings) task at SemEval-2022 (Mickus et al., 2022) encouraged participants to analyze the relation between two types of semantic descriptions, word embeddings and dictionary glosses, by proposing two sub-tasks: Reverse Dictionary (RD) (Hill et al., 2016), in which participants must generate vectors from glosses, and Definition Modeling (DM) (Noraset et al., 2017), in which participants must generate glosses from vectors. These subtasks aim to be useful for explainable Artificial Intelligence (AI) by including human-readable and machine-readable data.

Given the didactic nature of these tasks, the output generated by these models should be as accurate as the most prestigious dictionaries. Hence, the process of selecting a quality dataset is a critical phase, as Garg et al. (2020) state: "a small number of data examples prevents an effective convergence to the task, while noisy data leads to incorrect convergence". In this case, tasks require that the glosses used in the training represent the exact meaning of the word being defined in the context that the embeddings were extracted. However, as per our understanding, coherence, rigour and lexicographical prestige of the provided dataset should be improved; although accessing a prestigious dictionary is not an easy task.

A Reverse Dictionary takes a description in natural language and generates a list of words satisfying it (Siddique and Sufyan Beg, 2018). First RD were Information Retrieval systems for Turkish (El-Kahlout and Oflazer, 2004) and Japanese (Bilac et al., 2004). Other approaches used lexical graphs (Thorat and Choudhari, 2016; Ortega-Martín, 2021) that capture the relationships between the words of the definition itself and between these and others similar to them at different levels (synonymy, hyperonymy, etc.). Other systems create a vector space from these lexical resources, such as Wordnet (Dutoit and Nugues, 2002; Calvo et al., 2016; Méndez

ID: Definition
es.train.212: “Biología.— Se dice de los microorganismos que no aceptan los colorantes habituales.”
es.train.250: “Zoología.— Cualquiera de los colibríes del género Chlorostilbon .”
es.train.119: “Servirse , darse ayuda mutuamente .”
es.train.120: “Trabajar (uso pronominal de ...)”

Table 1: Examples of glosses

et al., 2013). As in many other NLP fields, more recently have appeared approaches based on Neural Networks (NNs) such as Long Short-Term Memory (LSTMs) (Malekzadeh et al., 2021; Zhang et al., 2020b) or Transformers (Qi et al., 2020; Yan et al., 2020). Language Models (LMs) based on Recurrent Neural Networks (RNNs) (Hill et al., 2016) have also been used. Finally, from a linguistic point of view, Shaw et al. (2011) add syntactic knowledge, Zock and Schwab (2008) try to replicate the model of the mental dictionary and Zhang et al. (2020b) use morphological knowledge in their system.

Definition Modeling is a relatively new task based on using distributed word representations to generate its definition. Noraset et al. (2017) use an RNN to compute the probability of a word being part of the definition. Different approaches have been proposed to this Natural Language Generation (NLG) task. Usually these new methods are heavily focused on the importance of the context of the word being defined, like fine-tuning a BART model to define groups of words (Bevilacqua et al., 2020), using attention and a Skip-gram model to smooth the problems of word selection in the generation step (Gadetsky et al., 2018), or exploring new ways to understand the embeddings and their capabilities, resulting these in a new task named “usage modeling” (Zhang et al., 2020a). There have been few attempts to use pure linguistics traits to improve definition generation, like accounting polysemy as a generative target using multi-sense word embeddings (Kabiri and Cook, 2020) or using sememes to condense the semantic core of generated sentences (Yang et al., 2020).

This paper has the following structure. Chapter 2 contains a review of the data along with some linguistic knowledge we consider relevant. In chapter 3 the RD approach and results are presented. Finally, chapter 4 contains the conclusions and future work. Our contributions are the following:

- We point out the main problem of this task, the lack of high-quality lexicographic data.
- We present the third best model for the Spanish “sgns” embeddings Reverse Dictionary task, which due to the use of external resources is not valid for the challenge.
- We compare various approaches for the previous task, analysing different preprocessing strategies, model architectures, loss functions and embedding initialization tactics.

2 Data analysis

This section contains a review of the Spanish dataset structure, an introduction about relevant lexicography concepts and the RD task preprocessing techniques.

2.1 The data

The dataset can be used for both subtasks. It is stored in a JSON file where each element contains four or five keys: its ”ID”, its ”gloss” or definition, the character-based embeddings (”char”), the Word2Vec Skip-gram Negative Sampling embeddings (”sgns”) and, just for some languages, the ”ELECTRA” (Clark et al., 2020) embeddings. All of these embeddings have a dimensionality of 256. For the development of the Spanish RD model “sgns” embeddings were used, since it was considered that using a more static approach such as “char” would lower the performance of the model and ”ELECTRA” embeddings were not available. However, as it will be explained later, the model was found out to be scalable to other embedding types and languages. Table 1[1] contains some words from the Spanish dataset that will be useful in the subsequent analysis.

2.2 Linguistic analysis

Even though this is not a linguistic paper, there are lexicographic concepts that should be explained in order to reach a deeper understanding of the dataset flaws. One of the most common approaches to

[1]Appendix A contains the translation of these examples

classify lexical dictionaries distinguishes between general dictionaries (also known as usage dictionaries) and specific dictionaries (this classification includes, to name a few, encyclopedic, synonym and scientific dictionaries). Therefore, given the significant number of different possible uses of the CODWOE tasks, the dataset should only include generic term definitions found in usage dictionaries, not specific ones.

In second place, there is no consensual standard structure for definitions. However, two principles must be followed (del Teso Martín, 1987).

1. The definitions must specify the hyperonym of the word being defined.
2. The definition should explain the main distinction between the class and its instance.

In other words, the definition has to include a hyperonym which clusters the word being defined into a category (for instance, "person who...", "a block of rock that...") and the definition should specify the specific traits of its meaning (following the last examples, "...plays badminton", "...shines even at night").

Lastly, a given definition is not the only way that a word could be defined, but just a context related meaning. This is why, from our perception, static embeddings should be avoided in modern Natural Language Processsing (NLP) tasks. For that reason, "ELECTRA" embeddings, used in other languages but not available for Spanish, could be more representative than "sgns" or "char" embeddings. Furthermore, these contextual embeddings should have been extracted from solid examples of use which represent the exact meaning of the gloss.

2.3 Dataset review

Dataset review revealed that glosses did not only go against the previously explained notions, but also lack coherence and exactitude. As seen in table 1, many definitions include the category which they belong (for instance, "Zoology"), or some grammatical information ("pronominal use"). Although dictionaries usually include this kind of information, it should not be in the definition (García and José, 2017).

Another drawback is that the dataset combines generic and specific definitions. Generic definitions usually can be found in usage dictionaries like "DLE" for Spanish or "Oxford Dictionary" for English, meanwhile specific definitions include terms from a certain domain, like zoology or linguistics. In our opinion, using specific definitions in this phase of the task just add noise to the training and evaluation.

It should also be noted that the terms glosses have not global coherence, and most of them do not follow the hyperonym and main distinction principles. There are plenty of synonym definitions (not optimal for these tasks as the definition length is too low) and encyclopedic definitions (which add a lot of noise as they have to fully describe the word being defined). Data could be improved if basic lexicographic notions were applied, but it is understood that being a multilingual dataset and given the available resources, CODWOE team has done a great work.

2.4 Data preprocessing

Data was preprocessed by deleting stopwords and category words (a term at the beginning of the definition that indicates its semantic category). In the second RD approach, the lexical graph, which is explained in section 3, words from the definitions were also lemmatized using the Spacy Spanish models lemmatizer [2]. Evaluation showed that the definitions preprocessing has been the most useful factor in the RD task, which indicates that the quality of the original definitions is what has penalized the model the most.

3 Our approach

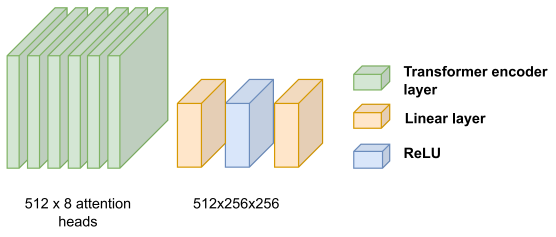


Figure 1: Model architecture

For the RD task, model was trained trying to make the definition embeddings as similar as possible as the defined word ones, focusing just on Spanish "sgns" (Word2Vec Skip-gram Negative Sampling embeddings of 256 dims) embeddings, al-

[2]`https://spacy.io/api/lemmatizer`

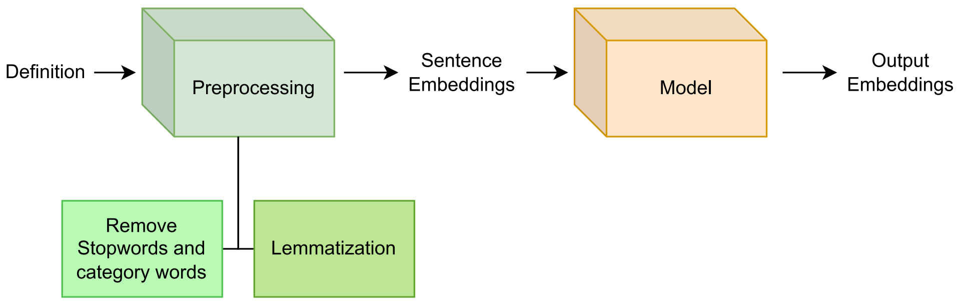


Figure 2: Full Reverse Dictionary pipeline

though system architecture was found scalable to other languages and embedding types. After test phase the model was applied also to Spanish “char” and to English “sgns” embeddings, achieving proportionally similar results which will be mentioned later.

Different tactics to initialize definition embeddings were also used. In our first approach we use Sentence-Transformers (Reimers and Gurevych, 2019). The “distiluse-base-multilingual-cased-v2”[3] was found to be the most appropriate model for Spanish tasks. Secondly, a lexical graph was built with the training dataset, in such a way that each defined word is related to the words in its definition. Then a SAGE Graph Neural Network (GNN) (Hamilton et al., 2017) with 2 SAGE layers with dimensions of sizes 256, 512 and 512 was used to perform message passing from every defined word to the words in its definition. To train this GNN we used 1 negative example for every positive edge, and trained the model for 50 epochs. Adam with 0.001 as learning rate was used as optimizer.

More than one model architecture were compared. As seen in figure 1, the final model was built with a Transformer encoder and an additional Multi-Layer Perceptron (MLP) with two linear layers with dimensions of sizes 512, 256, 256 and a ReLU layer between them, which during evaluation in the development set achieved better results than models based just on Transformer encoders or MLPs. Adam was used as optimizer. During training two loss functions from PyTorch were compared: Cosine Embedding Loss and Mean Square Error (MSE) Loss, which correspond to two of the task evaluation metrics. Regarding the hyperparameters, the following optimized the evaluation on the development set: 8 attention heads, 6 encoder layers, batch size=2048, learning rate=0.001. Model converges after around 10 epochs. Models were trained using 4 Nvidia Tesla v100 32GB.

As seen in figure 2, during training and prediction the process was the following. For a given sentence, stopwords and category tokens were removed. In the lexical graph model every remaining word was also lemmatized. After that the sentence embeddings are initialized, either with the Sentence-Transformers model, either by the mean of the lexical graph embeddings of every word in the definition. These initial embeddings are fed into the model along with some negative examples for the Cosine Embedding Loss model, receiving these a target label of 0. In the case of MSE Loss, negative sampling was not performed.

As stated in the CODWOE task guidelines, Reverse Dictionary submissions were evaluated using three metrics:

- mean squared error between the predicted embedding and the word embedding.
- cosine similarity between the predicted embedding and the word embedding.
- cosine-based ranking between the predicted embedding and the word embedding, which means how many other predicted embeddings have a higher cosine similarity with the target word than the right predicted one.

Since the Sentence-Transformer model was faster at generating the initial definition embeddings, it was used to initialize the definition embeddings in the final training and predictions. We understand that because of this our results in the task are not valid. However, the lexical graph approach can achieve almost similar results without the use of external data.

As seen in table 2, two different loss functions were used separately during training. Therefore,

[3]`https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased`

	MSE	Cosine Similarity	Cosine-based ranking
Cosine Embedding Loss	2.0157	0.4029	0.1665
Mean Square Error Loss	0.9106	0.2274	0.5003

Table 2: Reverse Dictionary results

two models were eventually presented. During evaluation, the first model trained with Cosine Embedding Loss reached more than 0.4 in cosine score and 0.16 in cosine ranking, which we consider remarkable and, according to the rankings[4], better than the top result for these particular Spanish embeddings. However, this model reached more than 2 in MSE, which considerably worsens the baseline (0.92) and the top results (0.85). On the other hand, the MSE Loss trained model slightly improves the baseline test MSE (0.91) and cosine score (0.22) but worsens the cosine ranking score. Other attempts to combine both loss functions did not success and achieved worse results in each of the evaluation metrics.

In the end, these results were found to be scalable to another languages and embedding types by using this same model architecture, and encountering in the way the same issues as for the Spanish "sgns" embeddings, that is, trouble combining MSE and cosine metrics. A Cosine Embedding Loss model for English ”sgns” embeddings achieves 0.34 cosine and 1.58 MSE, and using Spanish ”char” embeddings it reached 0.84 cosine and 1.66 MSE. As for the case of Spanish ”sgns”, compared to the top-ranked participants, a better cosine was achieved in exchange of a worse MSE. This leads to the opinion that the system architecture is easily scalable to other inputs, but it suffers from the same issues that with other languages and embeddings: a higher cosine similarity score can be achieved by using Cosine Embedding Loss, but in exchange of a worse MSE score.

4 Conclusions

For these tasks a combination of Machine Learning techniques and linguistic knowledge was proposed, in order to achieve good results and to understand the problems and the future challenges of these tasks.

In this paper, the main gaps in the dataset from a lexicographic perspective and its lack of coherence and exactitude were explained, and then a preprocessing solution was proposed, which was finally used in the RD system to avoid the problems that the dataset could carry in the model. Eventually, MMG team has presented a novel approach with an architecture that is easily scalable to other languages and embedding types. We understand that due to the use of external resources our results in the task are not valid for the challenge. However, we would like to remark that the lexical graph approach, which in the end we did not submit due to speed issues, achieved almost similar results.

These tasks are considered to represent an excellent starting point for research on the relationships between dictionaries and word embeddings. Both subtasks in general and our research on them in particular open up many options for further investigation. In our case, our intention is to use more linguistic knowledge at different levels, further exploring the power of linguistic graphs and putting into practice what we have learned in the Reverse Dictionary task to create quality Definition Modeling systems.

[4]`https://competitions.codalab.org/competitions/34022#results`

References

Bevilacqua, M., Maru, M., and Navigli, R. (2020). Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221.

Bilac, S., Watanabe, W., Hashimoto, T., Tokunaga, T., and Tanaka, H. (2004). Dictionary search based on the target word description. In *Proc. of the Tenth Annual Meeting of The Association for Natural Language Processing (NLP2004)*, pages 556–559.

Calvo, H., Méndez, O., and Moreno-Armendáriz, M. A. (2016). Integrated concept blending with vector space models. *Computer Speech & Language*, 40:79–96. ISBN: 0885-2308 Publisher: Elsevier.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

del Teso Martín, E. (1987). En torno a la definición lexicográfica. *Contextos*, (10):29–56.

Dutoit, D. and Nugues, P. (2002). A lexical database and an algorithm to find words. In *ECAI 2002: 15th European Conference on Artificial Intelligence, July 21-26, 2002, Lyon France: Including Prestigious Applications of Intelligent Systems (PAIS 2002): Proceedings*, volume 77, page 450. IOS Press.

El-Kahlout, I. D. and Oflazer, K. (2004). Use of wordnet for retrieving words from their meanings. In *Proceedings of the global Wordnet conference (GWC2004)*, pages 118–123.

Gadetsky, A., Yakubovskiy, I., and Vetrov, D. (2018). Conditional generators of words definitions. *arXiv preprint arXiv:1806.10090*.

García, J. and José, E. (2017). Forma y función del diccionario: hacia una teoría general del ejemplo lexicográfico. *Forma y función del diccionario*, pages 1–151.

Garg, S., Vu, T., and Moschitti, A. (2020). Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7780–7788.

Hamilton, W., Ying, Z., and Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hill, F., Cho, K., Korhonen, A., and Bengio, Y. (2016). Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Kabiri, A. and Cook, P. (2020). Evaluating a multi-sense definition generation model for multiple languages. In *International Conference on Text, Speech, and Dialogue*, pages 153–161. Springer.

Malekzadeh, A., Gheibi, A., and Mohades, A. (2021). Predict: Persian reverse dictionary. *arXiv preprint arXiv:2105.00309*.

Méndez, O., Calvo, H., and Moreno-Armendáriz, M. A. (2013). A reverse dictionary based on semantic analysis using wordnet. In *Mexican International Conference on Artificial Intelligence*, pages 275–285. Springer.

Mickus, T., Paperno, D., Constant, M., and van Deemter, K. (2022). SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Noraset, T., Liang, C., Birnbaum, L., and Downey, D. (2017). Definition modeling: Learning to define word embeddings in natural language. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ortega-Martín, M. (2021). *Grafos de vinculación semántica a partir del definiens del DUE*. PhD thesis, Universidad Complutense de Madrid.

Qi, F., Zhang, L., Yang, Y., Liu, Z., and Sun, M. (2020). Wantwords: an open-source online reverse dictionary system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–181.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Shaw, R., Datta, A., VanderMeer, D., and Dutta, K. (2011). Building a scalable database-driven reverse dictionary. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):528–540.

Siddique, B. and Sufyan Beg, M. M. (2018). A review of reverse dictionary: Finding words from concept description. In *International Conference on Next Generation Computing Technologies*, pages 128–139. Springer.

Thorat, S. and Choudhari, V. (2016). Implementing a reverse dictionary, based on word definitions, using a node-graph architecture. *arXiv preprint arXiv:1606.00025*.

Yan, H., Li, X., and Qiu, X. (2020). Bert for monolingual and cross-lingual reverse dictionary. *arXiv preprint arXiv:2009.14790*.

Yang, L., Kong, C., Chen, Y., Liu, Y., Fan, Q., and Yang, E. (2020). Incorporating sememes into chinese definition modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1669–1677.

Zhang, H., Du, Y., Sun, J., and Li, Q. (2020a). Improving interpretability of word embeddings by generating definition and usage. *Expert Systems with Applications*, 160:113633.

Zhang, L., Qi, F., Liu, Z., Wang, Y., Liu, Q., and Sun, M. (2020b). Multi-channel reverse dictionary model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 312–319.

Zock, M. and Schwab, D. (2008). Lexical access based on underspecified input. In *COLING 2008: Proceedings of the Workshop on cognitive Aspects of the Lexicon (COGALEX 2008)*, pages 9–17.

Appendix A Translation of Table 1

es.train.212: "Biology.— Said of microorganisms that do not accept the usual dyes ."
es.train.250: "Zoology.— Any of the hummingbirds of the genus Chlorostilbon ."
es.train.119: "To serve, to help each other."
es.train.120: "Work (pronominal use of ...)"

Edinburgh at SemEval-2022 Task 1: Jointly Fishing for Word Embeddings and Definitions

Pinzhen Chen **Zheng Zhao**
School of Informatics, University of Edinburgh
{pinzhen.chen, zheng.zhao}@ed.ac.uk

Abstract

This paper presents a winning submission to the SemEval 2022 Task 1 on two sub-tasks: reverse dictionary and definition modelling. We leverage a recently proposed unified model with multi-task training. It utilizes data symmetrically and learns to tackle both tracks concurrently. Analysis shows that our system performs consistently on diverse languages, and works the best with *sgns* embeddings. Yet, *char* and *electra* carry intriguing properties. The two tracks' best results are always in differing subsets grouped by linguistic annotations. In this task, the quality of definition generation lags behind, and BLEU scores might be misleading.

1 Introduction

We describe the University of Edinburgh's participation in SemEval 2022 Task 1 on comparing dictionaries and word embeddings (CODWOE), organized by Mickus et al. (2022).[1] The task features two directions: *reverse dictionary* and *definition modelling*. The former is to construct the embedding of a word given its definition gloss, and the latter is to generate the definition from a word embedding. The organizers provide datasets of word embedding-definition pairs across three types of embeddings and five languages. The training data has a size of 43.6k for each language, which is smaller than the data released in prior research (Hill et al., 2016; Chang et al., 2018). However, it provides a precious chance for a comprehensive study of lower-resourced reverse dictionary and definition modelling on languages other than English, as well as on different embedding architectures.

As our system architecture, we use a recently proposed unified model, which deals with both tracks concurrently and achieves superior results (Chen and Zhao, 2022). The model enables multi-task training by using word embeddings and definitions symmetrically. We also create ensembles and handcrafted phrases. Our code implementation builds on the organizers' and is publicly available.[2]

We submit to both reverse dictionary and definition modelling tracks, and cover all language and embedding combinations. Furthermore, we examine model generations and scores from three aspects: embedding architectures, languages, and linguistic annotations, aiming to figure out how these affect performance, subject to the models we have adopted. We finally show the information captured by different word embeddings and discuss the limitations in task evaluation and ranking.

Regarding the shared task outcome, we are the team with the most "gold medals": out of 18 sub-tracks, we attain first place in 8, second place in 4 and third place in 4. Our final ranks in the sub-tracks are detailed in Table 1.

Langauge		en	es	fr	it	ru
Reverse dictionary	sgns	2	4	3	2	3
	char	3	1	1	1	1
	electra	1	n/a	1	n/a	1
Definition modelling		4	3	2	2	1

Table 1: Our ranks in each sub-track.

2 Background

2.1 Datasets

The organizers provide datasets for five languages: English (*en*), Spanish (*es*), French (*fr*), Italian (*it*), and Russian (*ru*). Also, they supply 256d word embeddings from three architectures:

- *sgns*: static (non-contextualized) embeddings learned using skip-gram with negative sampling (Mikolov et al., 2013);
- *char*: character-based embeddings from an autoencoder trained on the spelling of a word;
- *electra*: contextualized embeddings produced by a generator-discriminator model (Clark et al., 2020).

[1] https://competitions.codalab.org/competitions/34022

[2] https://github.com/PinzhenChen/UnifiedRevdicDefmod

Despite that *electra* is not available for *es* and *it*, the data still covers 13 combination. All embedding architectures are trained on comparable corpora for all languages. Participants are not allowed to use any external resources, and words are provided as embeddings rather than actual words.

For each language, data is split into train, validation, test, and trial sets, at sizes 43.6k, 6,4k, 6.2k, and 0.2k. Human annotations are included in the trial split for analysis, but only word embeddings and definition glosses can be used for training. The snippet below exemplifies a single data instance with all possible fields. Training, validation, and test sets consist of only the bolded key-value pairs; all fields are found in the tiny trial set.

```
{"id":"en.trial.2",
 "sgns": [2.08729, 0.26177, ...],
 "char": [0.38789, 0.19716, ...],
 "electra": [-1.47715, -0.47424, ...],
 "gloss": "A mixture of other substances or things .",
 "word": "cocktail",
 "pos": "noun",
 "example": "a cocktail of illegal drugs",
 "type": "hypernym-based",
 "counts": 4187,
 "f_rnk": 13245,
 "concrete": 1,
 "polysemous": 0}
```

2.2 Evaluation metrics and ranking

Reverse dictionary is evaluated by three metrics:

- *MSE*: mean squared error between references and generated embeddings;
- *cosine*: cosine similarity between references and generated embeddings;
- *ranking score*: a percentage score measuring how many other test instances have a higher cosine similarity with a generated embedding than its reference does.

The definition modelling performance is measured by three too:

- *sense-BLEU*: sentence-BLEU implemented in NLTK with smoothing method 4 (Papineni et al., 2002; Chen and Cherry, 2014);
- *lemma-BLEU*: the maximum sense-BLEU between a generated gloss and all possible references of the same word and part of speech;
- *MoverScore*: a neural distance measure based on multilingual BERT (Zhao et al., 2019).

Finally, participants are ranked by rank scores instead of scalar numbers from the above metrics. A rank score is simply the rank of a particular submission among all submissions. For each sub-track, the average rank score of all three metrics is used to rank each team.

3 System Overview

3.1 Model Architecture

We select Chen and Zhao (2022)'s model as our system architecture because it has demonstrated great success on previous datasets for reverse dictionary and definition modelling. It is a "unified" model as it learns both tasks simultaneously, based on the intuition that a word and its corresponding definition share the same meaning, thus can be cast into the same neural semantic space.

We attach a diagram of this architecture as Figure 1. Technically, the model encodes glosses or word embeddings as the input, maps it into a shared representation, then generates embeddings or glosses accordingly. The shared representation serves as an autoencoding of both a word and its definition. Specifically, Linear layers (L) transform embeddings, and Transformer (Vaswani et al., 2017) blocks (T) encode or decode definitions.

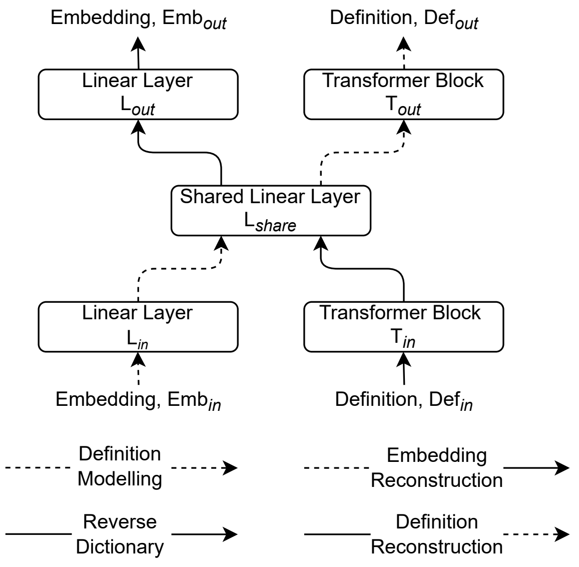


Figure 1: Chen and Zhao's illustration of the unified model.

3.2 Multi-task training

At the bottom of Figure 1, four trainable objectives are depicted: definition modelling, reverse dictionary, along with word embedding and definition reconstruction. The first two are CODWOE tasks, and the rest are auxiliary autoencoding tasks. Besides, another objective is to bring the vector representations of a word and its definition close in the shared layer. Our overall objective function combines the five objectives with equal weights.

3.3 Ensembling for reverse dictionary

Ensembling is a commonly employed technique to enhance machine learning performance. Specifically for reverse dictionary, we perform average ensembling: for each test instance, its final prediction is obtained by averaging all the corresponding predictions from different models. We ludicrously ensemble up to 21 models, of the same unified architecture, trained with various random seeds.

3.4 Handcrafting for definition modelling

Upon our initial inspection of definition modelling on the trial set, the generated definitions are mostly meaningless hallucinations, scoring a very low sense-BLEU of about 3. To understand how indicative BLEU is in this case, we handcraft a nonsensical n-gram submission. The rule is that for each test instance, we simply concatenate the most frequent bigram with the most frequent unigram, computed on all definitions in the training data. The phrases we prepare for each language are:

en	es	fr	it	ru
, or .	de la .	) (.	) (.	в . ,

4 Experiments and Results

4.1 Experimental setup

We tokenize glosses by whitespaces, add tokens into an open vocabulary, and embed them using one-hot. Word embeddings are used as provided. Loss functions are cross-entropy for tokens and MSE for embeddings. We also try cosine similarity for embeddings, but the model fails to converge. For definition modelling, we do not combine various embeddings as the input; this might put us at disadvantage in the team ranking.

While Transformer components are connected to form a unified model, most hyperparameters remain the same as in the provided baseline, which we specify in Appendix A. Following the original work, we tie Transformer embeddings and add a residual connection. We follow the same configurations for all language-embedding combinations. Training a unified model on an Nvidia GeForce RTX 2080 Ti takes roughly three hours.

4.2 Results

During the evaluation, we submit the provided baseline and our unified model. Also, we add ensembles of 17 and 21 models, as well the handcrafted n-grams. The submission scores, computed by the task organizers, are reported in Table 2 and 3. In the direction of reverse dictionary, the unified model steadily beats the baseline; ensembling adds a cherry on top for some languages but not all.

In definition modelling, our n-grams surpass genuine models on *en* BLEU scores, and even rank first in *fr* sense-BLEU among all participants' entries. This implies that either BLEU scores are not informative, or the model outputs are as embarrassing as the n-grams. On contrary, MoverScore is effective in downing the n-grams, probably by penalizing disfluency or semantic mismatch. Sadly, our manual review suggests that most model-generated

	en			es			fr			it			ru		
	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank
baseline	0.884	0.189	0.439	0.905	0.241	0.462	1.06	0.275	0.360	1.10	0.245	0.451	0.561	0.295	0.432
unified	0.871	**0.241**	**0.326**	0.868	0.339	**0.271**	**1.03**	**0.312**	**0.302**	1.05	0.371	**0.197**	0.553	0.327	0.340
ensemble 17	**0.864**	0.225	0.374	**0.860**	**0.347**	**0.271**	**1.03**	0.305	0.334	**1.03**	0.373	0.206	**0.538**	0.381	0.251
ensemble 21	0.865	0.225	0.374	**0.860**	**0.347**	**0.271**	**1.03**	0.306	0.330	**1.03**	**0.374**	0.205	**0.538**	**0.383**	**0.247**

(a) *sgns* as target embeddings

	en			es			fr			it			ru		
	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank
baseline	0.161	0.795	0.500	0.551	0.820	0.499	0.404	0.764	0.495	0.400	0.720	0.499	0.144	0.829	0.496
unified	0.143	0.795	0.500	0.480	0.834	0.431	0.347	0.782	0.448	0.337	0.745	**0.428**	0.119	0.849	0.395
ensemble 17	**0.142**	0.795	0.500	**0.467**	**0.839**	**0.424**	0.336	0.788	0.429	**0.334**	**0.747**	0.429	**0.116**	0.851	0.390
ensemble 21	**0.142**	0.795	0.500	**0.467**	**0.839**	0.425	**0.335**	**0.789**	**0.428**	**0.334**	**0.747**	0.429	**0.116**	**0.852**	**0.389**

(b) *char* as target embeddings

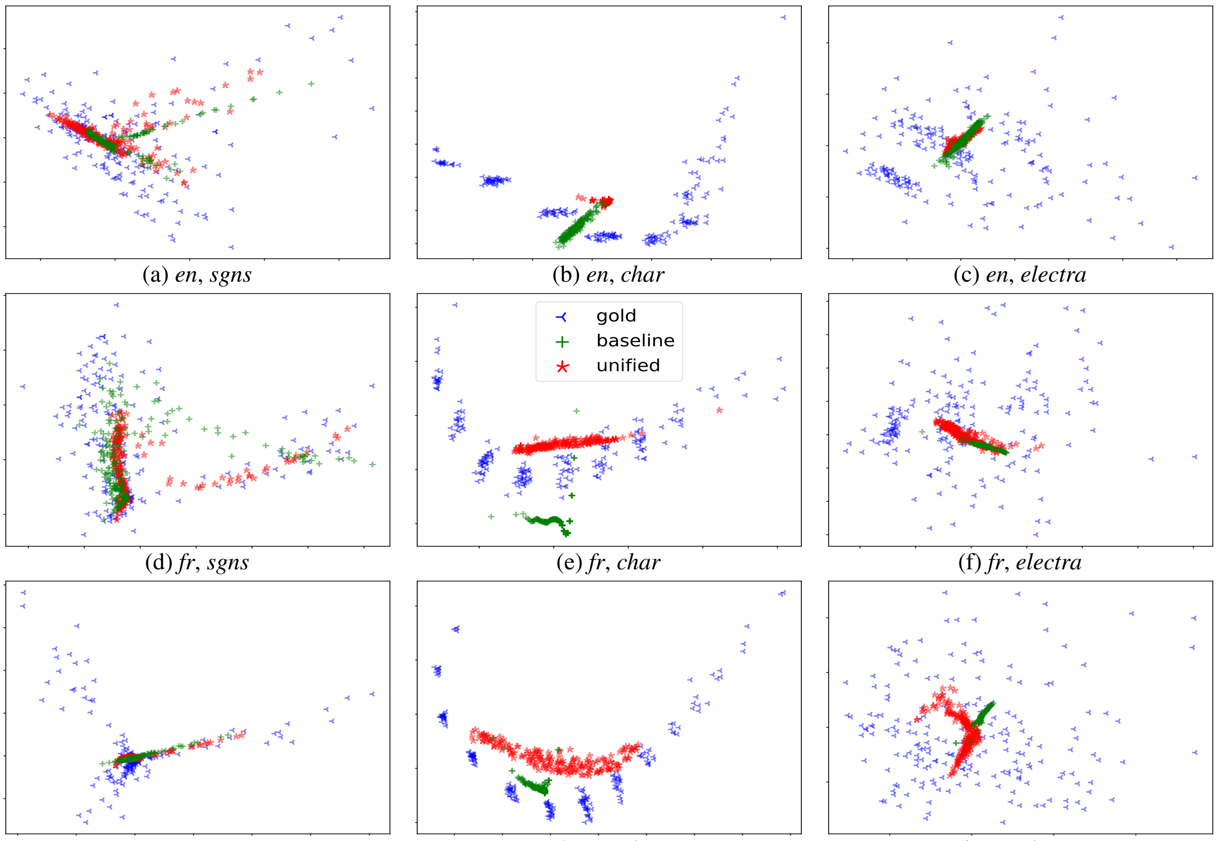
	en			fr			ru		
	MSE	cosine	rank	MSE	cosine	rank	MSE	cosine	rank
baseline	1.34	0.842	0.497	1.18	0.853	0.497	0.898	0.718	0.498
unified	1.32	0.844	0.495	1.08	0.861	0.476	0.846	0.731	0.421
ensemble 17	**1.31**	**0.847**	**0.490**	**1.07**	**0.862**	**0.479**	**0.829**	**0.735**	**0.417**
ensemble 21	**1.31**	**0.847**	0.491	**1.07**	0.861	0.480	**0.829**	0.734	0.419

(c) *electra* as target embeddings

Table 2: Reverse dictionary test performance, measured by MSE (↓), cosine similarity (↑), and ranking score (↓).

	source embed.	en MvSc	en s-B	en l-B	es MvSc	es s-B	es l-B	fr MvSc	fr s-B	fr l-B	it MvSc	it s-B	it l-B	ru MvSc	ru s-B	ru l-B
n-grams	n/a	-0.004	**3.06**	**3.81**	-0.032	2.73	3.67	-0.176	**2.95**	3.56	-0.164	1.89	2.74	-0.006	2.65	3.31
baseline	sgns	0.100	2.91	3.67	0.088	**3.47**	**5.28**	-0.019	2.34	3.38	0.046	4.62	6.97	**0.109**	**4.91**	7.14
unified	sgns	0.098	3.01	3.80	**0.101**	3.42	5.14	-0.064	1.59	2.38	**0.107**	**6.01**	**9.17**	0.095	4.59	6.82
baseline	char	0.101	2.47	3.02	0.064	2.06	2.88	-0.186	0.11	0.11	0.019	2.09	2.99	0.092	4.01	5.87
unified	char	**0.104**	2.83	3.40	0.065	2.14	2.96	**0.026**	2.42	**3.82**	0.044	2.93	4.29	0.085	4.80	**7.24**
baseline	electra	0.070	2.53	3.26	n/a			-0.075	1.38	1.93	n/a			0.090	3.78	5.45
unified	electra	0.094	2.75	3.43	n/a			-0.045	1.60	2.29	n/a			0.088	4.08	5.86

Table 3: Definition modelling test results, in MoverScore (↑), sense-BLEU (↑), and lemma-BLEU (↑).



(a) *en*, *sgns* (b) *en*, *char* (c) *en*, *electra*

(d) *fr*, *sgns* (e) *fr*, *char* (f) *fr*, *electra*

(g) *ru*, *sgns* (h) *ru*, *char* (i) *ru*, *electra*

Figure 2: Visualization of gold and output embedding distributions across languages and embedding architectures.

glosses are inaccurate. The dissatisfying results might be due to the modest training data size.

5 Performances across *embeddings*

Reverse dictionary MSE and cosine are incomparable across different embedding types, whereas ranking scores can tell which embedding architecture is preferred for indexing and retrieving a word. A random baseline ranking score is 0.5, and most *char* and *electra* figures, unfortunately, fall between 0.4 and 0.5. On the other hand, *sgns* is more useful as its baseline scores start at around 0.45, and our models can improve these up to 0.25.

We employ principal component analysis (PCA) to reduce the gold and output embeddings to 2 dimensions. Then in Figure 2 we visualize *en*, *fr*, and *ru*, which come with all embeddings. The unified model usually outputs to a larger space than the baseline, hinting at a positive correlation between output spread and performance. Gold *electra* has the most isotropic space, but neither model could imitate the distribution. *Char* has a crescent shape with several clusters inside, which is unlikely to be cosine-friendly. These problems are alleviated on *sgns*, which witnesses the best ranking scores.

Definition modelling *Sgns* is again the winner, as models trained with it reach the top in many metrics. *Char* is also favourable. This is counterintuitive as *electra* should be fitter, for it retains more sense-specific knowledge. A possible reason is that *electra* needs to go through more training data than *sgns* and *char* to reach perfection.

6 Performances across *languages*

As seen in the result and rank tables, our system's behaviour is relatively consistent on various languages, except that English is more challenging. Assuming that the datasets are of similar quality, it is questionable to conclude that our model suits other languages more than English. Moreover, Figure 2 confirms that the English embeddings are not more peculiar than those of other languages.

We guess that other teams have focused on English (e.g. only submitted English), as it is a centred language in the research community. Instead, our hyperparameter search is based on the average loss from all languages, neglecting that the losses are not directly comparable.

7 Performances across *linguistic features*

We look into the unified model's trial set predictions, to interpret how scores vary across diverse linguistic annotations: polysemy, part of speech (POS), word length in characters, definition length in words, and word frequency. For categorical features, we group data by annotations; for numerical features, we divide the data into three subsets, by percentile ranges: 0-33, 33-67, and 67-100. Statistics of the subsets are in Table 4. We list cosine similarity for reverse dictionary, and lemma-BLEU for definition modelling. A generic discovery is that, the best scores of the two tracks emerge in differing subsets, regardless of what the feature is.

Linguistic feature	**Category / Range**	**No. of instances**
Polysemy	Yes	65
	No	135
Part-of-speech	Adj	56
	Adv	11
	Verb	37
	Noun	96
Word frequency (frequency rank in the whole corpus)	67 – 11145	67
	11146 – 44416	66
	44417 – 905726	67
Word length	3 – 5	85
	6 – 7	60
	8 – 17	55
Definition length	1 – 6	71
	7 – 10	65
	11 – 39	64

Table 4: Statistics of the different subsets grouped by features.

Polysemy Table 5 exhibits the results for the words with either one or multiple definitions. It is slightly easier to achieve better cosine similarity for unambiguous words. Polysemous words have better BLEU, and *electra* has worse BLEU than *sgns*. This is illogical, as defining a polysemous word is harder, especially without context. We hypothesize that BLEU is not reflective, and *electra* embeddings might be of sub-optimal quality.

Polysemy	sgns		char		electra	
	cosine	l-B	cosine	l-B	cosine	l-B
Yes	0.232	**4.34**	0.804	**3.20**	0.836	**3.61**
No	**0.360**	2.82	**0.813**	2.53	**0.845**	3.09

Table 5: Performances across polysemy annotations for *en*.

Part of speech Next, numbers for the four POS tags that exist in *en* trial, are laid out in Table 6. Strong cosine similarity is associated with verbs, although cosine numbers are close, except for adverbs. Adverbs, which have a small sample size, dominate high lemma-BLEU, perhaps because they are the least ambiguous.

POS	sgns		char		electra	
	cosine	l-B	cosine	l-B	cosine	l-B
Adj	0.319	3.36	0.801	2.76	0.811	2.81
Adv	0.134	**6.56**	0.798	**5.45**	0.815	**5.93**
Verb	**0.383**	3.20	**0.839**	2.50	0.853	3.83
Noun	0.314	2.97	0.806	2.53	**0.860**	2.99

Table 6: Performance across POS tags for *en*.

Word length We then make three partitions according to different word length ranges. Results in Table 7 suggest that shorter words have higher cosine, while longer words have higher lemma-BLEU. Numbers are closer for *sgns* and *electra*; we further investigate on *char* in Section 8.1.

Word length	sgns		char		electra	
	cosine	l-B	cosine	l-B	cosine	l-B
short	**0.332**	3.19	**0.845**	2.58	0.817	3.10
medium	0.314	3.19	0.842	2.74	**0.867**	**3.41**
long	0.327	**3.66**	0.694	**3.00**	0.854	3.33

Table 7: Performances across word lengths for *en*.

Definition length Likewise in Table 8, we separate the trial data by the gold definition length. Much higher BLEU is seen when the model defines words linked with a shorter gold gloss, as generating a shorter sequence is easier. As we anticipate, when the model produces word embeddings for longer glosses, results are better too, potentially because more information can be encoded.

Definition length	sgns		char		electra	
	cosine	l-B	cosine	l-B	cosine	l-B
short	0.280	**4.51**	0.796	**3.60**	0.824	**4.89**
medium	0.318	3.48	0.814	2.73	0.848	2.76
long	**0.361**	1.83	**0.822**	1.80	**0.856**	1.93

Table 8: Performances across definition lengths for *en*.

Word frequency Finally, Table 9 summarizes the results of the low, medium, and high frequency word groups. From the results, we cannot establish an explicit trend across different task directions, embeddings, or word frequencies. This implies that the embedding quality and model performance might be word frequency-agnostic.

Frequency	sgns		char		electra	
	cosine	l-B	cosine	l-B	cosine	l-B
low	0.250	3.53	0.805	**2.82**	0.850	3.30
medium	0.348	**3.54**	0.786	2.76	**0.864**	**3.38**
high	**0.357**	2.89	**0.839**	2.66	0.814	3.10

Table 9: Performances across word frequencies for *en*.

8 Qualitative Analysis and Discussions

8.1 Observing the crescent with a telescope

After PCA retains the most distinguishing components, Figure 2 shows interesting patterns, especially for *char*. We randomly label 25 English words and present them in Figure 3 and Figure 4, respectively for *char* and *electra*. The sub-clusters in *char*'s crescent are perfectly in tune with word lengths; for *electra*, more frequent words are closer to the origin. We do not notice a clear trend for *sgns*, for which a plot is attached as Figure 5.

We attribute the distinct patterns to the training paradigms: character-level word autoencoding for *char*, and contextualized modelling for *electra*. This accounts for the largest cosine gap on *char* between long and short words, seen earlier in Table 7. Intuitively, it is more difficult to train *char* autoencodings for longer words, so, in turn, embeddings for longer words possess inferior quality.

Within *char* embeddings, words are grouped by lengths, so we may utilize this for word retrieval in future work. Nonetheless, we are unsure of how length or frequency information aids sense-based tasks, like definition generation in our context.

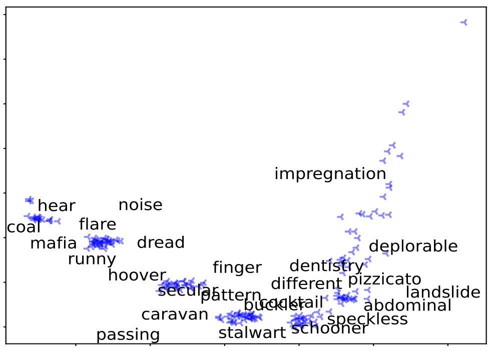


Figure 3: Gold English *char* embeddings with word labels.

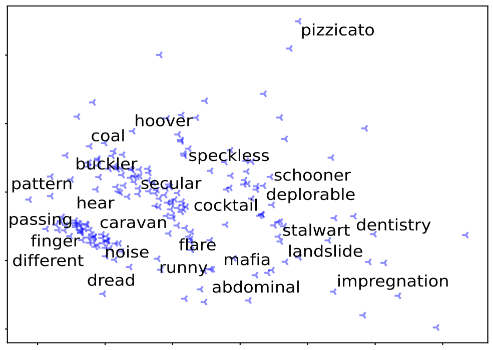


Figure 4: Gold English *electra* embeddings with word labels.

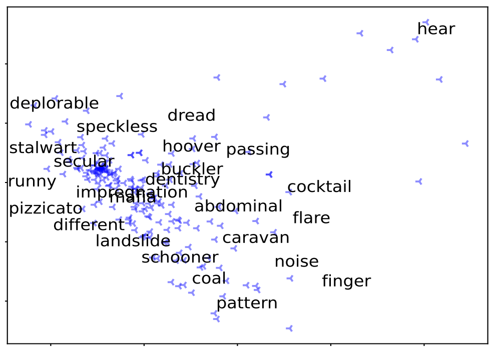


Figure 5: Gold English *sgns* embeddings with word labels.

8.2 Sense-BLEU with no sense

We design a sanity check on the representativeness of BLEU. On the English trial set, we remove punctuation marks and NLTK-defined stop words from both references, and our unified model's definitions generated from *sgns*. Sense-BLEU drops from 3.31 to 0.39, and surprisingly, it worsens to 0 with smoothing disabled. Evidently, sense-BLEU and thereby lemma-BLEU are hugely inflated by functional tokens as well as smoothing.

8.3 Evaluating task evaluation and ranking

We point out the limitations associated with the evaluation and ranking process, which can benefit from a rethink. First, as shown above, the two BLEU metrics may not be practical. Second, some metrics are correlated, i.e., cosine with the ranking score, and sense-BLEU with lemma-BLEU. These problems are amplified by the team ranking protocol, which averages a team's ranks in individual metrics to produce a final standing. It might not be meaningful to compare the individual metric ranks, not to mention averaging them since metrics are not equally weighted.

Nonetheless, we are not in a knowledgeable position to propose a better approach, other than clumsily displaying ranks in individual metrics.

Acknowledgements

Pinzhen Chen is supported by funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825303 (Bergamot). Zheng Zhao is supported by the UKRI Centre for Doctoral Training in Natural Language Processing (UKRI grant EP/S022481/1). This paper reflects the view of the authors and not necessarily that of the funders.

References

Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. 2018. xSense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *arXiv*.

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level BLEU. In *Proc. of WMT*.

Pinzhen Chen and Zheng Zhao. 2022. A unified model for reverse dictionary and definition modelling. *arXiv*.

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

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *TACL*, 4:17–30.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: CODWOE – comparing dictionaries and word embeddings. In *Proc. of SemEval*.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL*.

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

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proc. of EMNLP-IJCNLP*.

A Hyperparameters

Hyperparameter	**Value**
learning rate	1e-4
optimizer	Adam
beta1, beta2	0.9, 0.999
weight decay	1e-6
batch size	256
decoding beam size	6
early stopping	5 non-improving validations
embedding loss	mean squared error
token loss	cross-entropy
Transformer depth	4
Transformer head	4
Transformer dropout	0.3
linear dropout	0.2
shared layer dim.	256
word embed.	*sgns*, *char*, *electra*
word embed. dim.	256
definition embed.	one-hot
definition embed. dim.	256
vocabulary size	open, all training tokens

Table 10: Model hyperparameters.

RIGA at SemEval-2022 Task 1: Scaling Recurrent Neural Networks for CODWOE Dictionary Modeling

Eduards Mukans
mukans.work@gmail.com
University of Latvia

Gus Strazds
gs15014@students.lu.lv
University of Latvia

Guntis Barzdins
guntis@latnet.lv
University of Latvia, IMCS

Abstract

Described are our two entries "emukans" and "guntis" for the definition modeling track of CODWOE SemEval-2022 Task 1. Our approach is based on careful scaling of a GRU recurrent neural network, which exhibits double descent of errors, corresponding to significant improvements also per human judgement. Our results are in the middle of the ranking table per official automatic metrics.

1 Introduction

The definition modeling track of SemEval-2022 Task 1: CODWOE - COmparing Dictionaries and WOrd Embeddings (Mickus et al., 2022) challenged participants to generate dictionary glosses from individual word embedding vectors. This paper describes two CODWOE submissions, "emukans" and "guntis", where the first focuses on the automatic CODWOE scores, but the second attempts to gauge the relationships between scaling laws, the automated metrics, and human evaluation. Our submissions achieved competitive results (see Figure 3) on the MoverScore official metric - scoring 1st for French, 2nd for Spanish, and 3rd for Russian.

Our approach was to apply classical recurrent networks, such as Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRU) (Cho et al., 2014), to definition modeling and investigate how model scaling impacts performance. The scaling effect is well investigated for transformers, but not so much for RNNs. Recently, the main focus in deep learning has skewed from searching for new model architectures to investigating how various factors impact the training process and overall system performance (Nakkiran et al., 2019; Kaplan et al., 2020; Gordon et al., 2021). The main factors are: the amount of data, the amount of compute, and the size of the model (parameter count). In the competition the data amount is fixed and no use of external data is permitted, thus we investigate how scaling model size and training time impacts training progress and model performance for recurrent models.

In our experiments we did observe deep double descent effects: epoch-wise double descent with respect to both cross-entropy loss and prediction accuracy on a validation data set, more pronounced with increasing model size.

We also investigated the automatic metrics used for evaluating submissions and their correlation with human evaluation, focusing primarily on the MoverScore metric (Wei Zhao, 2019). MoverScore does correlate with human evaluation, but not necessarily very strong, at least for this dataset. We find that the double descent effect seen with respect to prediction accuracy can also be observed for MoverScore.

2 Background

The CODWOE shared task invites participants to compare two types of semantic descriptions: dictionary glosses and word embedding representations. The task consists of 2 subtracks: definition modeling and reverse dictionary. In definition modeling participants have to generate glosses from word embedding representations. The reverse dictionary task is the inverse: reconstruct a word embedding from the corresponding gloss. Considering results achieved by the baseline models provided by the organisers, we decided to participate only in the definition modeling track, as it seems the more challenging task, with more room for potential improvement.

For the definition modeling track, inputs are 256-dimensional embedding vectors and outputs are plain text. Data is provided for 5 languages: English, French, Spanish, Italian and Russian. Every language is scored separately. We submitted for all 5 languages. The provided word embedding vectors are of 3 types: CHAR, SGNS, and ELECTRA.

3 System overview

For the definition modeling task, we used classical recurrent networks, experimenting with both LSTM and GRU architectures. We added an initial fully connected input layer to scale a given word embedding vector to higher dimensions. We used the ADAM optimizer (Kingma and Ba, 2014) in the training process. The learning rate was set in the range ∈ [1e-5, 3e-5, 1e-4]. A linear learning rate decay schedule with warm-up over 0.01 was used. No preprocessing was applied to training data. The code is available on GitHub [1]

The very first step is creating a tokenizer and building its vocabulary. We use SentencePiece tokenization (Kudo and Richardson, 2018), trained on the training dataset only. We carried out experiments across a range of vocabulary sizes.

We used a classical approach and a decoder only part of standard seq2seq (sequence to sequence) recurrent neural network models without attention. The GRU/LSTM state vector is initialized from the given defmod embedding vector. In our case, we use a single word embedding vector type. For the first time step we pass a single <seq> token to model as input. The model outputs a single predicted token and a new state vector. To avoid exploding gradients, the outputs of the model are normalized. The token selected by the model is appended to the generated gloss, and is also used as input to the model for the next time step. This process is repeated until reaching the iteration limit.

At every time step, the model can make mistakes. If the initial part of the input sequence fed to a seq2seq model is bad, most likely the subsequent output sequence will also be wrong. To mitigate this accumulation of errors and speed up the training process, we use the teacher forcing technique (Williams and Zipser, 1989). With teacher forcing, the model is trained by supplying input tokens from the target sequence of the dataset and using the network's one-step-ahead predictions to do multi-step sampling. We also tried a more advanced teacher forcing technique: scheduled sampling (Duckworth et al., 2020), where input sequence tokens are given ground-truth values only with some probability. Unfortunately, scheduled sampling did not give good results - the loss plot was very noisy. It is likely that the CODWOE definition modeling task itself is a very hard task with too much variability relative to the amount of provided training data; scheduled sampling might be better suited for language model fine-tuning when the model weights are pretrained on a large corpus and already correlate fairly well with natural language syntax and semantics.

After each training epoch, the model is evaluated on a validation dataset using the same cross-entropy loss function as used for training. We also use an accuracy metric for evaluating model performance, as it correlates with perplexity and human judgement for large language models. The accuracy is calculated by dividing the count of correctly predicted tokens (under teacher-forcing) by the number of total tokens.

For "emukans" submissions, model training is stopped using early stopping (Prechelt, 2012) based on the accuracy score for the validation data, while "guntis" submissions were intentionally trained long past the overfitting point to observe scaling and double descent effects.

4 Experimental setup

For our experiments, we have 5 Tesla v100 16GB GPUs provided by our institute. During the competition, our focus was on exploring different training effects and model tuning. Most of the experiments were focused on primary factors of "scaling laws": model size and the amount of compute (training epochs).

For simplicity and consistency of presentation, in most of the following tables and figures (all except for Figure 3) we report experimental performance evaluated against a trial dataset provided by the CODWOE organizers, which consists of only 200 glosses. Apart from the automatic metrics, our focus was on (informal) manual evaluation of generated glosses.

4.1 Vocabulary size

The vocabulary of distinct tokens available for use by an NLP model is generally built during a data preparation stage, and the size of this vocabulary is a key factor in model performance. Therefore we started our experiments by tuning the size of the vocabulary.

We build our token vocabulary from the training dataset only. Taking into account the relatively small training dataset - only 43k glosses and 18k unique words, we reasoned that the token vocabu-

[1] https://github.com/emukans/codwoe

Vocab size	MoverScore	S-BLEU	L-BLEU
250	0.09702	0.02504	0.02508
500	0.10662	0.02452	0.02455
800	0.11754	0.02469	0.02470
1500	0.13045	**0.02726**	**0.02726**
3000	0.13379	0.02679	0.02681
5000	**0.13625**	0.02593	0.02596
15000	0.09638	0.02053	0.02056

Table 1: Influence of vocabulary size on the automatic metrics

lary size should be fairly small. Therefore we set our hypothesis as the following:

Hypothesis 1 (H1): *Optimal vocabulary size is around 10% of the unique word tokens.*

During initial training experiments, we noticed a tendency of the model to repeat the same gloss for many different word embeddings. We speculate that repeating such 'most popular' glosses might give the model higher chances of matching frequently occurring words or phrases in the dataset.

In the vocabulary size optimization experiment, we used the GRU model with 2 layers, hidden dimension 768, and 30 tokens limit during training. Table 1 summarizes our results on the trial set. We selected vocabulary size *1500* as it has the highest BLEU scores, relatively good MoverScore and the most promising glosses during manual evaluation. 1500 tokens are 8.3%, the result is close to 10%, confirming hypothesis 1 experimentally.

4.2 Model size scaling

Recent trends in deep learning suggest that bigger models increase performance on most tasks (Brown et al., 2020; Rae et al., 2021). However, the focus in these cited papers is given to Transformer (Vaswani et al., 2017), Convolutional (ConvNets) or Residual networks (ResNets). Classical recurrent neural networks (RNN) such as GRU or LSTM have been left out of the mainstream investigation of scaling effects. In the following experiments, we show that scaling RNNs also gives similar positive effects as for other network architectures. Our approach could be formulated with the following hypothesis:

Hypothesis 2 (H2): *Scaling RNNs in depth or width improves their performance.*

We summarize our experiments in tables 2, 3 and 4. The results tentatively confirm hypothesis 2. We observe that no matter how one scales the model, in width (higher hidden dimension) or depth (more layers), the performance does increase in both cases. Of course, these results are only for relatively small models fitting into our compute capacity (trained using a single Nvidia V100 GPU).

Layers	MoverScore	S-BLEU	L-BLEU
1	0.11458	0.02564	0.02561
2	0.11312	0.02427	0.02426
4	0.12454	0.02548	0.02548

Table 2: Scaling GRU model layers with fixed hidden size: 3072 dim.

Hidden	MoverScore	S-BLEU	L-BLEU
512	0.10851	0.02439	0.02437
1024	0.10880	0.02342	0.02341
3072	0.11312	0.02427	0.02426
4096	0.11071	0.02453	0.02450

Table 3: Scaling hidden dimensions for 2 layer GRU model.

4.3 Double descent

Classical machine learning theory says that increasing the model size or training time beyond some optimum, while keeping the amount of data constant, will eventually lead to the model overfitting. (i.e., bigger models would give worse performance than optimally sized smaller models). Recently, a new effect was discovered (Nakkiran et al., 2019) which contradicts, or amends, this traditional wisdom. The double descent effect states that increasing the model size (i.e., model-wise double descent) or compute resources invested into training (i.e., epoch-wise double descent) indeed leads to overfitting at first, but further increasing the size of the model or the training time can, at some critical point, reverse the trend, so that performance starts increasing again.

During our model scaling experiments we aimed

Hidden	MoverScore	S-BLEU	L-BLEU
1024	0.07284	0.02018	0.02014
2048	0.11915	0.02430	0.02427
3072	0.12454	0.02548	0.02548

Table 4: Scaling hidden dimensions for 4 layer GRU model.

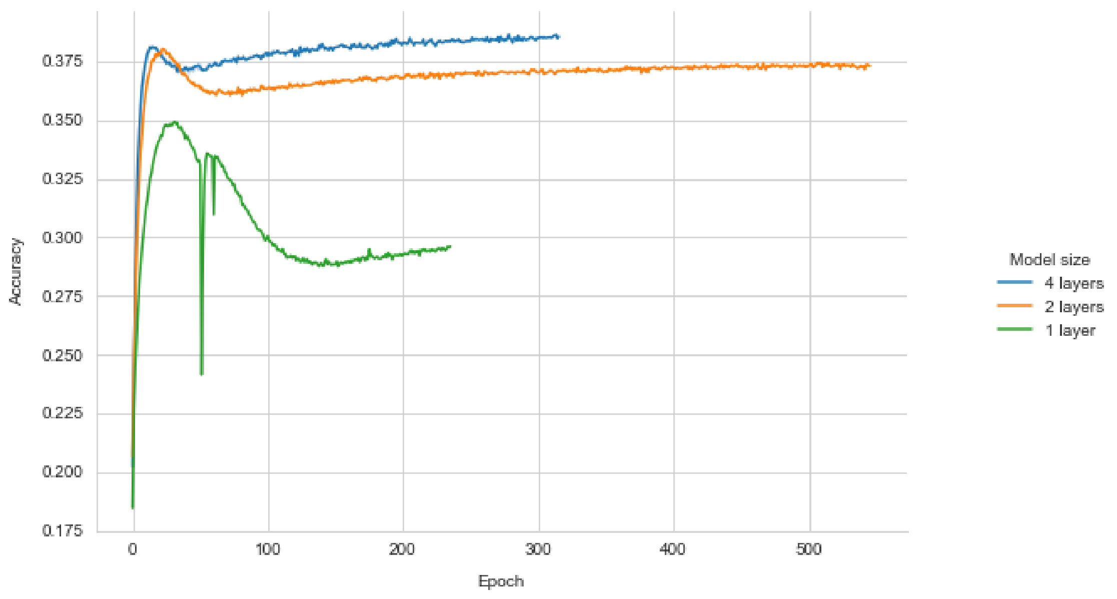


Figure 1: Accuracy: Correctly predicted word tokens, assuming that all previous word tokens were correct. The scores are calculated on the development (validation) dataset. In all cases, the hidden layer size is 3072 dim.

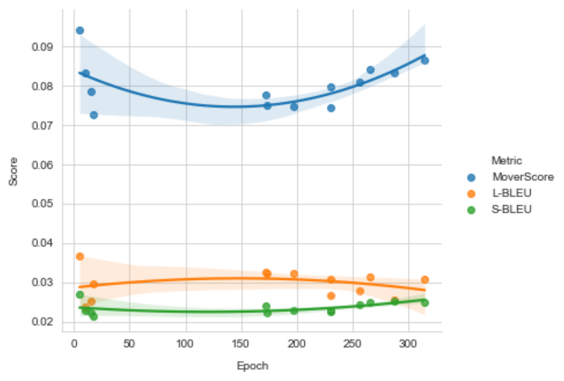


Figure 2: Automatic metric change during the 4-layer model training. The scores are calculated on the trial dataset.

to replicate the double descent effect and bring the model quality to a new level after initial overfitting. Since our compute resources were limited and we could not scale our model size endlessly, we investigate the following hypothesis:

Hypothesis 3 (H3): *Training the GRU model longer leads to an epoch-wise double descent effect.*

For our experiments, besides automatic evaluation metrics for the defmod task we introduced also an accuracy score.

Definition 1 (Accuracy): *Percent of correctly predicted tokens when all previous input tokens are correct.*

In figure 1 are 3 plots for 1-, 2- and 4-layer GRU models. The 4-layer model shows a clear epoch-wise double descent effect. We can also observe that, as previously demonstrated for other kinds of models, the effect occurs only when the model size is big enough relative to the training set. The 1- and 2-layer models are apparently too small for this training set and the task complexity.

Figure 2 is for the same 4-layer model, but in this case plotting scores on the metrics used for the CODWOE defmod task. We can see some correlation with the accuracy plot in figure 1, but these metrics seem to be less sensitive overall.

In the table 5 we illustrate the continuing gloss quality improvement according to human judgement after the first accuracy spike in the automatic metrics (epoch 5). Glosses become semantically closer to the original word. Hence, we conclude that hypothesis 3 is empirically confirmed.

5 Results

Our team "emukans" and "guntis" placed in the middle of the final ranking table. However, if we inspect the scores in figure 3, we see that our solution (a green line) does outperform others in some metrics for some languages (i.e., top score for French, 2nd for Spanish, 3rd for Russian).

Analysing our submission results, we noticed that MoverScore can give even negative scores and is quite variable from one example to another. The score is generally very low if the generated gloss length differs substantially (either too long or too

Word	Ground-truth	Epoch	Predicted	MoverScore
scraggy	Lean or thin, scrawny.	5	A slightly used to slightly.	0.11885
		193	Adorned with one or more gauntlets	0.06659
		315	Ase, slender, thin .	0.12597
coal	A glowing or charred piece of coal, wood, or other solid fuel.	5	slightly; to slightly.	0.00863
		193	A blust or furnished vehicle .	0.17182
		315	supply with energy, especially of a person's size.	0.10929
beautiful	Pleasant; clear.	5	having been (a person); to suggest or despons.	0.11287
		193	sufficient attention or thought, especially concerning the avoidance of harm.	0.03470
		315	suitable or proper; extraordinary; epic.	0.19444
thirsty	Craving something.	5	having been used to suggest or slightly.	0.04727
		193	Causing by a sensation of alcohol or narcotics.	-0.02330
		315	Causing by anger or excitement.	0.05691

Table 5: The evolution of gloss prediction during training. **N.B.** The word in column one is informational only, it was not available in the train/dev datasets and was not used during training nor prediction.

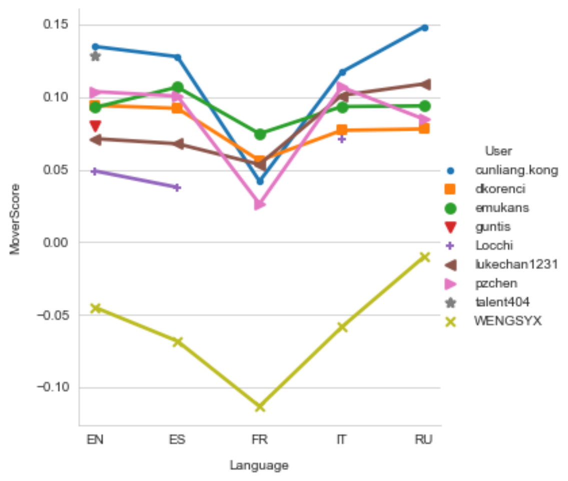


Figure 3: Best MoverScore results for all participants in all languages.

short) from the length of the ground-truth gloss, irrespective of whether a human can perceive some semantic alignment between the two.

Analysing the available data, we see that many of the glosses are relatively short: up to 20 tokens (but there are also very long examples). We conjecture that one strategy for increasing MoverScore might be to simply limit all generated glosses to 20 tokens or less.

6 Conclusion

In this competition, we tried a classical recurrent neural network approach for the CODWOE definition modeling task, and obtained positive results.

Several topics require deeper investigation. A good metric for automatically measuring how semantically close are two sentences is still an unsolved problem. MoverScore is still too far from human judgement. Taking into account even the best scores for the definition modeling task, the task is still in very early stages, and models that are trained only on the provided data cannot generate any practically useful outputs. This could possibly be addressed with much larger training datasets, or by allowing the use of external data (or of large pretrained language models). In general, it seems that a word semantic could not be represented using a single vector. The task requires more context to capture the semantics. Maybe the task could be changed to generating a gloss for a set of synonyms or semantically close words.

Acknowledgements

This research was supported by the Latvian Council of Science, project No. lzp-2021/1-0479, and by the H2020 project SELMA (under grant agreement No. 957017).

References

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

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Daniel Duckworth, Arvind Neelakantan, Ben Goodrich, Lukasz Kaiser, and Samy Bengio. 2020. Parallel scheduled sampling.

Mitchell A Gordon, Kevin Duh, and Jared Kaplan. 2021. Data and parameter scaling laws for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5915–5922, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: Codwoe – comparing dictionaries and word embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2019. Deep double descent: Where bigger models and more data hurt. *CoRR*, abs/1912.02292.

Lutz Prechelt. 2012. *Early Stopping — But When?*, pages 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446.

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

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

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

Uppsala University at SemEval-2022 Task 1: Can Foreign Entries Enhance an English Reverse Dictionary?

Rafal Černiavski and **Sara Stymne**
Department of Linguistics and Philology
Uppsala University
rafal.cerniavski.2286@student.uu.se, sara.stymne@lingfil.uu.se

Abstract

We present the Uppsala University system for SemEval-2022 Task 1: Comparing Dictionaries and Word Embeddings (CODWOE). We explore the performance of multilingual reverse dictionaries as well as the possibility of utilizing annotated data in other languages to improve the quality of a reverse dictionary in the target language. We mainly focus on character-based embeddings. In our main experiment, we train multilingual models by combining the training data from multiple languages. In an additional experiment, using resources beyond the shared task, we use the training data in Russian and French to improve the English reverse dictionary using unsupervised embeddings alignment and machine translation. The results show that multilingual models occasionally but not consistently can outperform the monolingual baselines. In addition, we demonstrate an improvement of an English reverse dictionary using translated entries from the Russian training data set.

1 Introduction

In a reverse dictionary, one can look up a gloss, an explanation of a word's meaning, to find the most relevant word or word form. The applications of reverse dictionaries are numerous, as they can help language learners in expanding their vocabulary, authors and writers in looking for the most suitable word, and avid cruciverbalists in taking on some of the most challenging crosswords.

Reverse dictionary modelling has seen approaches ranging from traditional information retrieval using relevance scores (Zock and Bilac, 2004) to ones involving node-graph architectures (Zhang et al., 2020). As a general rule, the quality of a reverse dictionary appears to largely depend on the availability of annotated data. However, annotated data are scarcely available and expensive to produce for low-resource languages. We therefore explore the viability of multilingual approaches to improve the quality of a reverse dictionary.

This work is performed in the context of the reversed dictionary subtask of the SemEval 2022 task 1, COmparing Dictionaries and WOrd Embeddings (Mickus et al., 2022). Unlike standard reverse dictionaries, the target is to predict a word embedding vector for each gloss, rather than a word form. Three types of word embeddings are available: character-based embeddings (*char*), Skip-grams (*sgns*), and contextual embeddings (*electra*). No additional resources are allowed in the shared task. In this paper, we do present additional experiments, though, where we also used an external machine translation engine. While five languages were made available in the shared task, we mainly focus on English, but also give some results for Russian and French.

The main research question of this study is thus whether the performance of a monolingual reverse dictionary can be improved using data in other language(s) in a low supervision setup. We first explore what are the most suitable type of embeddings for a Transformer-based reverse dictionary. Having found the best-performing embeddings, we use them to train a joint model for multilingual reverse dictionary, which can map glosses to words in multiple languages. Finally, we use the training data in French and Russian to improve the quality of an English reverse dictionary by means of unsupervised embeddings alignment and machine translation.

We did not submit our results in the evaluation period since in one of the experiments we used a pre-trained neural machine translation model, which is prohibited in the shared task. Nevertheless, we report the performance of our jointly trained multilingual models on the test sets, as no additional data or pre-trained models were involved in training. For character-level embeddings, our best multilingual models, when tested on English, would rank 25th in terms of mean squared error (MSE), 20th in terms of cosine similarity (COS), and 9th in terms of cosine-based ranking (CRK); on French: 22nd (MSE), 10th (COS), 3rd (CRK); on Russian: 7th (MSE), 7th (COS), 13th (CRK).

2 Related Work

Recent research has explored bilingual and cross-lingual reverse dictionaries, the task of which is to map a gloss in a source language to a word in target language. An implementation by Qi et al. (2020) involved a machine translation API and bilingual dictionaries to re-direct a query in the source language through the target language pipeline. Yan et al. (2020) implemented the first cross-lingual reverse dictionary based on mBERT (Devlin et al., 2019), a Transformer-based language model trained on Wikipedia articles in 104 languages. Their study revealed that unaligned cross-lingual reverse dictionary achieves best performance when mBERT is tuned on unaligned multilingual data; its quality is substantially worse than that of a monolingual model. Yan et al. (2020) thus concluded that it remains unclear how multilingual data is to be utilized to improve the quality of unaligned reverse dictionary, which is to be explored in this project.

Joint multilingual models, which are trained on multiple languages at once, offer a solution for low-resource languages that often have little to none annotated data. This has for example been explored for dependency parsing, with positive results (Kondratyuk and Straka, 2019; Smith et al., 2018).

Cross-lingual embeddings are of central importance in word meaning similarity across languages (Jimenez et al., 2017), and are thus a crucial component of cross-lingual reverse dictionaries. As noted by Ruder et al. (2019), the applicability of cross-lingual embeddings relies on their quality, which, in turn, depends on the availability of bilingual corpora and dictionaries. Nevertheless, an unsupervised cross-lingual embeddings alignment method proposed by Lample et al. (2018) enables high quality cross-lingual embeddings with no or little supervision, further allowing for unsupervised machine translation. Unsupervised cross-lingual embeddings alignment thus offers a solution for both mapping the word embeddings and its glosses from one language to another.

3 System Description

We focus on the strategies of utilizing the data in foreign languages to improve reverse dictionary rather than the choosing of most suitable model. Therefore, we use the SemEval 2022 task 1 baseline system, a Transformer-based architecture with all parameters unchanged for all of our models.

3.1 Methodology

The methodology adopted can be divided into a preparatory step and two main experiments. The initial step sought to learn the most suitable type of embeddings for a Transformer-based English reverse dictionary. A baseline model was trained and tested three times on each type of embedding to learn whether there were notable deviations between the runs and the official baseline scores of the shared task. This was done to select the best performing type of embedding to be used in further experiments, thus avoiding spending the computational resources on numerous models with different embeddings.

The two main experiments build on the research of He et al. (2017), as they investigate joint training of multilingual models as well as cross-lingual embedding alignment. In the first experiment, the French and Russian training sets are concatenated to the English training set, one or both at a time. The joint models are then trained with a joint development set containing entries in all languages used in training. We choose the source languages, namely French and Russian, so as to investigate whether the similarities between the source and target language, such as shared words, similar script, and typological proximity can affect the performance of a multilingual reverse dictionary.

In the second experiment, the embeddings of source entries (in French and Russian) are firstly aligned to the target embedding space (English) with no supervision using the MUSE library (Lample et al., 2018). To ensure a fully unsupervised setup, the refinement and evaluation steps involving bilingual corpora are disabled. The alignment is conducted in five epochs using all standard parameters. In the process, the target embeddings are anchored. Their values are not updated in order to preserve the quality of the pre-trained embeddings. Secondly, the glosses of the first 4,500[1] entries from the now-aligned source training set are

[1]A relatively small number of glosses were translated due to the limited access to the tool used for machine translation.

Embeddings	MSE	σ	COS	σ	CRK	σ
sgns	1.193	0.009	0.259	0.007	**0.405**	0.012
char	**0.156**	0.014	0.810	0.003	0.469	0.003
electra	1.846	0.172	**0.840**	0.001	0.483	0.002

Table 1: The baseline performance of a Transformer-based English reverse dictionary trained on different types of pre-trained embeddings, averaged over 3 runs. The standard deviation (σ) shows the fluctuation of the scores over the three runs.

translated and attached to the target (English) training set. Since the word forms are masked in the training data, we were unable to train an unsupervised machine translation model. The glosses are thus translated using a pre-trained neural machine translation model, namely Watson API[2]. Lastly, the translated glosses are tokenized using the spaCy tokenizer to mirror the tokenization in the original data sets provided by the organizers of the shared task.

3.2 Evaluation

All models are primarily evaluated on the trial data set. This is due to the fact that Experiment 2 used a pre-trained machine translation model, which goes against the rules of the contest. We, however, additionally evaluate our jointly trained multilingual models on the test set, as the model does not use any additional resources.

The models were evaluated based on the three official metrics of the shared task: mean squared error (MSE), cosine similarity (COS), and cosine-based ranking (CRK) (Mickus et al., 2022).

4 Results and Discussion

4.1 Choice of Embeddings

The performance of the baseline models trained on the English training data set with different embeddings can be seen in Table 1. The scores are highly similar to the baselines published by Mickus et al. (2022) and are primarily included to estimate the stability of the performance of a Transformer architecture on each type of embeddings.

Individually, each type of embedding achieves the highest score on one of the parameters, with *char* achieving lowest MSE, *electra* securing highest cosine similarity, and *sgns* having best cosine-based ranking. Overall, *char* embeddings demonstrate the most stable and good performance across all three parameters. The *char* embeddings also had a relatively low standard deviation between runs for all metrics, as opposed to *electra* on MSE.

The results seem to have several implications. Firstly, the three evaluation parameters favour divergent information encoded by the three types of embeddings. Most notably, character-level information stored in *char* embeddings substantially minimizes MSE of the predicted embeddings of a word. This might be because character-level embeddings are effective in addressing out-of-vocabulary words (Polatbilek, 2020). In other words, they seem to enable the Transformer model to learn the mapping between glosses and characters that add up to words denoting the glosses. However, such mapping suffers from a major limitation, as character-level embeddings do not differentiate between the senses of a word. Most effective in handling this task are the contextualized embeddings (*electra*), for they encode a word depending on the surrounding context. Depending on the context, the sense might differ, thus leading to completely different values in the embeddings space. It can thus be argued that both character-level and contextualized features are important for a reverse dictionary model; an ideal solution could perhaps utilize using both types of embeddings for fine-grained retrieval of words.

Seeing as *char* embeddings had a good and stable performance overall, we further explore them in the following experiments.

4.2 Multilingual Model

The performance of multilingual models jointly trained for two or three languages at a time is reported in Tables 2 and 3.

The multilingual models perform similarly to the monolingual baselines. As can be seen from comparing the models' performance across trial and test sets, some differences are likely due to chance and fall within the range of a standard deviation reported in 1. Nevertheless, it is rather surprising that the English reverse dictionary seems to bene-

[2]https://developer.ibm.com/components/watson-apis/

	English (E)				French (F)			Russian (R)		
Metric	E (Base)	E+F	E+R	E+F+R	F (Base)	F+E	F+E+R	R (Base)	R+E	R+E+F
MSE	0.17893	0.18897	**0.14708**	0.19417	**0.39491**	0.43406	0.51295	**0.13858**	0.15327	0.25199
COS	0.79591	0.78978	**0.80659**	0.79472	**0.78361**	0.77169	0.77499	**0.84409**	0.83503	0.83073
CRK	**0.45771**	0.46748	0.49775	0.48978	0.47125	0.45235	**0.45225**	0.42565	0.41385	**0.40665**

Table 2: The performance of a multilingual reverse dictionary jointly trained on *char* embeddings in the source and target language evaluated on the **trial** set. The performance of multilingual model (joint) is reported alongside its monolingual baseline.

	English (E)				French (F)			Russian (R)		
Metric	E (Base)	E+F	E+R	E+F+R	F (Base)	F+E	F+E+R	R (Base)	R+E	R+E+F
MSE	0.17893	0.22773	**0.17821**	0.21932	**0.45808**	0.50001	0.53045	0.16775	**0.16075**	0.24864
COS	**0.79591**	0.76452	0.78445	0.77336	**0.77978**	0.75831	0.76917	**0.84044**	0.83220	0.83349
CRK	0.45771	**0.45639**	0.46198	0.46480	0.45006	**0.42284**	0.43047	0.42073	**0.40115**	0.40776

Table 3: The performance of a multilingual reverse dictionary jointly trained on *char* embeddings in the source and target language evaluated on the **test** set. The performance of multilingual model (joint) is reported alongside its monolingual baseline.

fit from the Russian data more than it does from the French data. In addition, when trained on both English and Russian, the model performs better on Russian.

In the case of multilingual models, it might be productive to focus on the lack of losses rather than the lack of gains. The results indicate that the performance of Transformer-based English reverse dictionary remains unaffected by both a relatively close language (French), and a distant language (Russian). This might be due to the fact that the high-quality pre-trained embeddings exist in different vector spaces. Despite the fact that the data are concatenated, the Transformer architecture learns to differentiate between the two and only retrieve words from the relevant vector space.

The shared space of models like mBERT is arguably the main reason why the joint tuning of models on data in multiple languages at once leads to best performance of a cross-lingual reverse dictionary for Yan et al. (2020). Overall, it is debatable whether there is reason to train a multilingual reverse dictionary on several unaligned languages. Such a model takes longer to train and tune, occupies more space, and does not offer much apart from the convenience of not having to switch between multiple models.

4.3 Embeddings Alignment and Machine Translation

The last experiment involved unsupervised embeddings alignment and machine translation of the glosses from source language (French and/or Russian) to target language (English). During alignment, the target embeddings were anchored to retain the values of the pre-trained embeddings. However, due to system constraints, the target embedding values changed from ten decimal points to five. To address this and to see whether this could affect the results in a negative way, an additional model was trained with the restored original values (with ten decimal points) of the embeddings in English, while the source (French and Russian) embeddings were kept at five decimal points. The results are presented in Table 4 alongside the baseline results.

Alignment without translation of glosses in most cases affected the model in a negative way, as it only introduced noisy foreign data. However, the machine translated glosses attached to the aligned values from source language seemed to have a positive effect on the English reverse dictionary when the source language was Russian. In the case of French, the approach failed completely. The retention of the original embedding values as opposed to the last five digits being lost led to mixed results. Though in most cases the difference is small and might have occurred by chance, the results could also indicate that it is crucial for the source and target embeddings to be similar in terms of the quality

		English + French			English + Russian		
Metric	Baseline	Al	Al+T	Al+TR	Al	Al+T	Al+TR
MSE	0.156	0.184	0.171	0.184	0.162	**0.135**	0.171
COS	0.810	0.799	0.810	0.801	0.807	**0.811**	0.810
CRK	0.469	0.474	0.500	0.483	0.477	0.501	**0.463**

Table 4: The performance of a Transformer-based English reverse dictionary trained on aligned and joined data (Al), aligned with target embeddings cut off past five digits and machine translated glosses (Al+T), as well as aligned with recovered target embeddings and machine translated glosses (Al+TR).

in a cross-lingual space.

A rather surprising finding of the experiment was the improvement of an English reverse dictionary using the data in Russian. Contrary to the findings of Yan et al. (2020), a more substantial improvement for English was observed with a distant source language, which uses a completely different script. The Russian language has been previously proposed as a generally good source language across several tasks and target languages, though (Turc et al., 2021). As for this experiment, perhaps the alignment produced with no supervision was of higher quality with Russian, allowing to correctly project the foreign source entries in the target space. It is also possible, though unlikely, that the translations of glosses from Russian to English were of higher quality than those of French to English.

5 Conclusions

This project has investigated whether an English reverse dictionary can be improved using data in foreign languages. This research question was addressed by firstly determining the most suitable type of embeddings for a Transformer-based reverse dictionary. Secondly, multilingual joint models were trained to see the affects on the performance of English as target language and two source languages, namely French and Russian. Lastly, the embeddings from source language were aligned to the target embedding space, followed by machine translation of the respective glosses.

Three key findings emerged. Firstly, character-level features lead to best performance of an English Transformer-based reverse dictionary. Secondly, multilingual reverse dictionaries perform comparably with monolingual ones, as no substantial improvement or decline was observed. Thirdly, an English reverse dictionary can be improved using the available data in foreign languages, such as French and Russian, though the improvement is rather small. In the reported experimental setup, Russian was found to be a more suitable source language in enhancing an English reverse dictionary.

There are numerous possible extensions of the present study. One could, for instance, recreate the study in a fully supervised or fully unsupervised setup so as to see to what extent the lack of supervision affected the results. It would also be interesting to investigate whether combinations of embeddings, e.g. contextual and character-level, would lead to better performance of reverse dictionary models. Overall, the improvements recorded in this study were, arguably, hardly significant. It may therefore be productive to search for more successful ways of using data in foreign languages in creating or improving reverse dictionaries.

Acknowledgements

Computations were enabled by resources in project UPPMAX 2020/2-2 at the Uppsala Multidisciplinary Center for Advanced Computational Science.

References

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

Junqing He, Long Wu, Xuemin Zhao, and Yonghong Yan. 2017. HCCL at SemEval-2017 task 2: Combining multilingual word embeddings and transliteration model for semantic similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 220–225, Vancouver, Canada. Association for Computational Linguistics.

Sergio Jimenez, George Dueñas, Lorena Gaitan, and

Jorge Segura. 2017. RUFINO at SemEval-2017 task 2: Cross-lingual lexical similarity by extending PMI and word embeddings systems with a Swadesh's-like list. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 239–244, Vancouver, Canada. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations (ICLR)*, Vancouver, Canada.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2022. SemEval-2022 Task 1: CODWOE – COmparing Dictionaries and WOrd Embeddings. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ozan Polatbilek. 2020. *Enriching Contextual Word Embeddings with Character Information*. Ph.D. thesis, Izmir Institute of Technology (Turkey).

Fanchao Qi, Lei Zhang, Yanhui Yang, Zhiyuan Liu, and Maosong Sun. 2020. WantWords: An open-source online reverse dictionary system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–181, Online. Association for Computational Linguistics.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, 65(1):569–630.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. 82 treebanks, 34 models: Universal Dependency parsing with multi-treebank models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 113–123, Brussels, Belgium. Association for Computational Linguistics.

Iulia Turc, Kenton Lee, Jacob Eisenstein, Ming-Wei Chang, and Kristina Toutanova. 2021. Revisiting the primacy of English in zero-shot cross-lingual transfer. arXiv preprint, arXiv:2106.16171.

Hang Yan, Xiaonan Li, Xipeng Qiu, and Bocao Deng. 2020. BERT for monolingual and cross-lingual reverse dictionary. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4329–4338, Online. Association for Computational Linguistics.

Lei Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2020. Multi-channel reverse dictionary model. arXiv preprint, arXiv:1912.08441.

Michael Zock and Slaven Bilac. 2004. Word lookup on the basis of associations : from an idea to a roadmap. In *Proceedings of the Workshop on Enhancing and Using Electronic Dictionaries*, pages 29–35, Geneva, Switzerland. COLING.

BL.Research at SemEval-2022 Task 1: Deep networks for Reverse Dictionary using embeddings and LSTM autoencoders

Nihed Bendahman[†], **Julien Breton**[†], **Lina Nicolaieff**[†],
Mokhtar Boumedyen Billami[†], **Christophe Bortolaso**[†], **Youssef Miloudi***[†]
[†]Berger-Levrault, 64 Rue Jean Rostand, 31670 Labège, France
*CARL Berger-Levrault, 361 All. des Noisetiers, 69760 Limonest, France
{nihed.bendahman, julien.breton, lina.nicolaieff, mb.billami,
christophe.bortolaso}@berger-levrault.com, {youssef.miloudi}@carl.eu

Abstract

This paper describes our two deep learning systems that competed at SemEval-2022 Task 1 "CODWOE: Comparing Dictionaries and WOrd Embeddings". We participated in the subtask for the reverse dictionary which consists in generating vectors from glosses. We use sequential models that integrate several neural networks, starting from Embeddings networks until the use of Dense networks, Bidirectional Long Short-Term Memory (BiLSTM) networks and LSTM networks. All glosses have been preprocessed in order to consider the best representation form of the meanings for all words that appears. We achieved very competitive results in reverse dictionary with a second position in English and French languages when using contextualized embeddings, and the same position for English, French and Spanish languages when using char embeddings. Our source code can be found at GitHub[1].

1 Introduction

Distributed representations of words (or word embeddings) (Bengio *et al.*, 2003; Mikolov *et al.*, 2013; Pennington, Socher and Manning, 2014) have shown to provide useful features for various tasks in natural language processing (NLP) and computer vision. While there seems to be a consensus concerning the usefulness of word embeddings and how to learn them, this is not yet clear with regard to representations that carry the meaning of a full sentence. That is, how to capture the relationships among multiple words and phrases in a single vector remains a question to be solved.

Much recent research in computational semantics has focused on learning representations of arbitrary-length phrases and sentences. The reverse dictionary represents one of the most common cases to solve this problem of learning sequence representations. That said, the reverse dictionary is the task to find the proper target word given the word description (Hill *et al.*, 2016; Hedderich *et al.*, 2019; Zhang *et al.*, 2019; Yan, Li and Qiu, 2020). For example, the composed meaning of the words in a dictionary definition (*A mixture of other substances or things*) should correspond to the meaning of the word that define it (*cocktail*). As mentioned by Hill *et al.* (2016), this bridge between lexical and phrasal semantics is useful because high quality vector representations of single words can be used as a target when learning to combine the words into a coherent phrasal representation.

In this paper, we present our contributions to solve the reverse dictionary problem using very specific neural architectures and applying supervised learning. For more information on task 1 of SemEval-2022 as described by its organizers, we invite the reader to consult the paper of Mickus, Timothee *et al.* (2022).

Our approaches require a model able of learning to map between arbitrary-length phrases and fixed-length continuous-valued word vectors. For this purpose, we experiment with two broad classes of neural language models (NLMs): Recurrent Neural Networks (RNNs) with (Bidirectional) Long-Short

[1] https://github.com/jln-brtn/BL.Research-at-SemEval-2022-Task-1

Term Memory (BiLSTM and LSTM), which naturally encode the order of input words (or characters), and simpler (feedforward) lexical units embedding models. These lexical units can be bag-of-words (BOW) or a sequence of characters.

After having described in section 2 in more detail the problem to be solved in the SemEval-2022 evaluation campaign and the data provided, we present in section 3 the previous works of reverse dictionary. Then, in section 4, we present our neural architectures with all data preprocessing having been performed. Thereafter, in section 5, we describe the experimental setup implemented before presenting the results in section 6 and concluding in section 7.

2 Background

2.1 Problem Description

The CODWOE shared task[2] consists of compare two types of semantic descriptions: dictionary glosses and word embedding representations. The problem can be defined as follows: given a definition, can we generate the embedding vector of the target word? That said, there are several questions to be solved: (1) How should we compare two very different types of semantic representation? (2) Will contextualized embeddings help better define polysemous (ambiguous) words that have multiple senses? and (3) Can we have the same evaluation performances of the same neural architecture for different natural languages? In other words, five natural languages are studied in this task, namely: English, French, Spanish, Italian and Russian language. Our goal is to answer the following question: can the same model or the same neural network architecture be beneficial for all languages?

2.2 Data Description

The organizing members of the reverse dictionary task proposed different JSON files that contain definitions and their vector representations. Each JSON file describes information about a one natural language for a list of five languages, namely: French, English, Spanish, Italian and Russian language. The corpus is therefore multilingual. Before having the test corpus, the data have been split in different sets: trial, train, and development corpus. All the models we will present have been trained and validated on train and development corpus.

Before describing the data, we can do a quick focus on the size of these data. As we mentioned, the data are split into 3 groups. For each language, the trial dataset contains 200 elements pairs (definitions and their embedding vectors), the train dataset contains 43,608 elements pairs, and the development dataset contains 6,375 elements pairs. Regarding the test corpus, we have 6,208 definitions for each language. The organizers have provided different vector representations for the definitions. All these representations are continuous vectors (embedding vectors). They have 256 dimensions and are built with well-known three techniques:

- "char" corresponds to character-based embeddings, computed using an auto-encoder on the spelling of a word.
- "sgns" corresponds to Skip-Gram with negative sampling embeddings, aka. Word2Vec (Mikolov *et al.*, 2013).
- "electra" corresponds to Transformer-based contextualized embeddings.

As mentioned above, all datasets of SemEval-2022 task 1 are multilingual. This is an important point because we can imagine and create a system that manages these multilingual datasets, or several systems specialized in one language. We will explore this option in the “Experimental Setup”.

3 Related Work

In the field of natural language processing, word embeddings have been the subject of several research problems for many years. Indeed, a text contains various information, and the idea is to resorb the target information in a continuous vector representation. Embedding’s methods improve significatively the results against standard statistical approaches and justified the interest these last years. In order to create embeddings, the scientific research community is experimenting with two approach types: unsupervised and supervised learning of embeddings.

The unsupervised approaches are the most common and consist in using a pretrained language model on a large corpus such as Word2Vec (Mikolov *et al.*, 2013) or Glove (Pennington, Socher and Manning, 2014). From the word

[2] https://competitions.codalab.org/competitions/34022

embeddings we can obtained, we must then choose a technique in order to find the right combination of words that will best convey the desired information. For example, it's possible to apply combinations such as: average, sum, or centroid to obtain a vector that reflect the representation for a sequence of words. We can find an example of centroid usage with Lwin and Nwet (2019) for news summarization extraction or a centroid vector weighted by IDF (Inverse Document Frequency) (Arora, Liang and Ma, 2017).

For the supervised approaches to produce sequence text vector representation, the idea is consisting of modeling the link between a content and an embedding representation. The reverse dictionary is a common case of sequence or short content representation. Some state-of-the-art models are used to perform this task, for example neural networks LSTM (Sherstinsky, 2018) and Bert (Pre-training of Deep Bidirectional Transformers for Language Understanding) (Devlin *et al.*, 2018). Notably in the paper of Yan, Li and Qiu (2020), they experiment with word generation from a definition using Bert multilingual architecture. As it's mentioned in their paper, the use of a Bert model is a great idea and can, at least, achieves state-of-the-art performances for both monolingual and cross-lingual reverse dictionary task. Even better, the proposed framework here can perform cross-lingual reverse dictionary task without aligned data.

We can also talk about the work of Morinaga and Yamaguchi (2020), Malekzadeh, Gheibi and Mohades (2021) which are based on a Long Short-Term Memory (LSTM) architecture. With always our objective to produce a vector of a contextualized text, the LSTM offers great prospect in this field of research. Indeed, the recurrent neural network architecture of the LSTM allow models to perform on sequential data which is exactly our case study in this task.

4 System Overview

In this section, we describe the models we proposed in the CODWOE – Reverse dictionary shared task. In order to keep comparable and linguistically significant the results submitted by the different participants, the organizers of CODWOE disallowed any use of external resources, including standard datasets as well as pretrained models that could be used for this task (such as Word2Vec models or contextual pretrained models based on Transformer's architectures like Bert). Given this condition, we decided to explore the sequential models, and particularly the LSTM and BiLSTM models. All the following models' architecture we created are based on TensorFlow[3] and Keras[4] (Chollet and others, 2015) libraries.

4.1 Data preprocessing

Before introducing our models, we want to mention that we have performed preprocessing on the content data. By using Stanza[5] (Qi *et al.*, 2020), we lemmatized all definitions and removed all punctuations. We decided to do this to minimize alternative words for the same concept and help our models to correctly process the vocabulary.

To optimize our workflow, we worked on the data before and independently from the neural network architectures. In this way, we built new files based on the lemmatization of the main corpus. This process is possible because all languages are covered by Stanza[6].

4.2 Baseline Model

In this section, we will introduce our first model, called: Baseline Model. This model is intentionally simple in order to create baseline scores and introduce manipulation on the datasets. The figure 1 presents our first architecture.

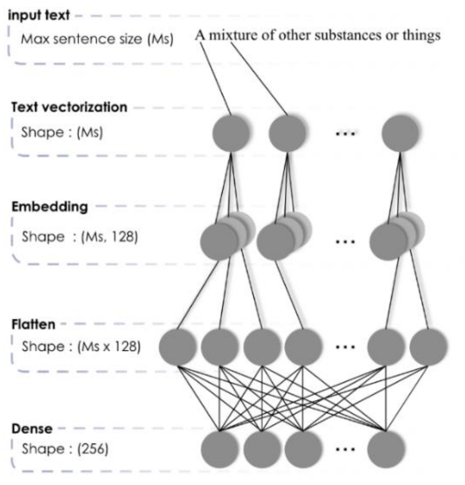


Figure 1 : Baseline Model Architecture.

[3] https://www.tensorflow.org/

[4] https://keras.io/

[5] https://stanfordnlp.github.io/stanza/

[6]https://stanfordnlp.github.io/stanza/available_models.html

As you can see on the figure 1, our model created with Keras contains four different layers and starts with the text vectorization. This first layer will transform the input text in a vector to be process by the next layers. To perform this operation, we must give an identifier to each different word in our corpus. After that, each sentence will be represented as a vector of identifiers. To be processed, the vector of a sentence must have the same size for all sentences. That said, we take the maximum sentence size (Ms in figure 1) and normalize all the vectors by adding zero values in the end.

Now, each vector's sentence is ready to be processed by the embedding layer. This layer turns positive integers (indexes) into dense vectors of fixed size. After this operation, the flatten layer will change the dimensionality of the data from two dimensions to one dimension without losing any value. The shape of this layer will be the multiplication of the two dimensions of the previous layer. Finally, we model the output data by using a fully connected layer (Dense layer) with 256 dimensions to match with the gloss embeddings gave by the organizers.

4.3 Advanced Model

LSTMs are a Recurrent Neural Networks (RNN) (Medsker and Jain, 2001) which have an internal memory that allows them to store the information learned during training. LSTMs are frequently used in the reverse dictionary task (Sherstinsky, 2018) and in word and sentence embeddings tasks in general (Augustyniak, Kajdanowicz and Kazienko, 2019; Liu *et al.*, 2020), as they can learn long-term dependencies between existing words in the sentence and thus compute context representation vectors for each word. BiLSTM for its part, is a variant of LSTMs, it allows a bidirectional representation of words (Augustyniak, Kajdanowicz and Kazienko, 2019).

Our second model, named Advanced Model, is therefore a BiLSTM-LSTM network. As for the baseline model, we use a sequential model which can be provided by Keras. The figure 2 presents our second architecture. This last one starts with a text vectorization layer, followed by an Embedding layer and a dense layer producing vectors of the words passed as an input, the vectors have (length of the longest sentence, 128) dimensions. Then we added a BiLSTM layer, which takes a recurrent layer (the first LSTM of our network) which in turn takes the "merge mode" as an argument. This mode specifies how the forward and reverse outputs should be combined; in our case the average of the outputs is taken.

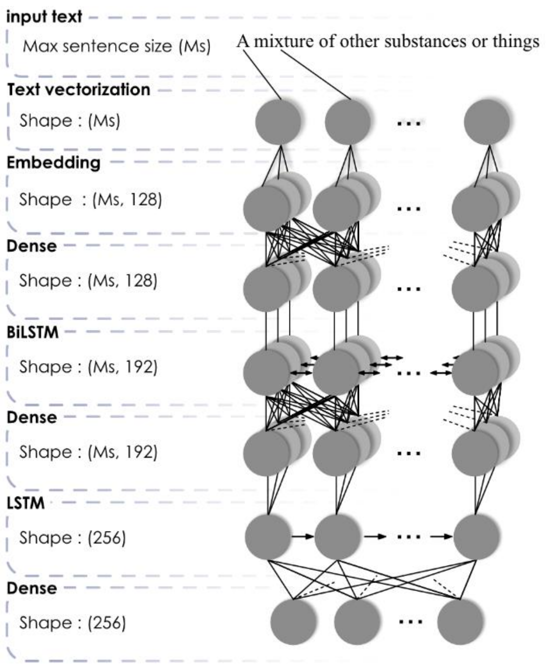


Figure 2 : Advanced Model Architecture.

To these three layers, we added another fully connected Dense layer and a LSTM layer of 256 dimensions corresponding to the dimensions of the output vectors and a final Dense layer with the same dimensions as illustrated in figure 2. We use the Softmax as an activation function. For the hyper-parameters to train the model, we use the following: epochs = 10; batch size = 192; learning rate = 1e-3 and AdamW as an optimizer.

5 Experimental Setup

In this section, we describe different variants we tested. Since there were 3 types of vector representations proposed to us in this shared task, we used the same architectures to produce the 3 types of vectors. However, the data format given as input to the model is not the same for the 3 types. For the 'electra' and 'sgns' representation types, we prepare a vocabulary containing the words of the glosses of the 'training dataset', the words of this vocabulary were obtained by following the preprocessing described in section 4.1.

For the 'char' vector type, we construct a vocabulary of all the characters used in the glosses without preprocessing the data. The idea being that, for the 'char' type representation, the model encodes the characters of the glosses into vectors and then produce the vectors encoding the glosses based on the vectors of the characters constituting the glosses.

The model that we propose is a monolingual model, i.e., we trained it separately on the training dataset of each language provided. However, in order to evaluate the impact of using a multilingual model, we trained the same neural networks on five languages (with character vector) at the same time and compared the results obtained with those obtained by the monolingual models.

For the 'sgns' and 'electra' representation types, we built a vocabulary containing the words of all glosses on the five languages, which contains in total 121,147 words. We did the same with the 'char' vectors but with preparing a vocabulary of characters instead containing 405 characters, in total. The table 1 describes the vocabulary size for each monolingual model and for multilingual model.

Model Type	Language	Vocab Type	
		Words	Chars
Monolingual Model	English	21,001	139
	French	24,089	170
	Spanish	**29,383**	**229**
	Italian	25,414	162
	Russian	29,289	212
Multilingual Model	All languages	**121,147**	**405**

Table 1: Vocabulary size of models.

We can see that there are common words between the different languages since the multilingual model has a vocabulary of 121,147 words instead of 129,176. That said, there are 8,029 common words between at least two language vocabularies. Moreover, we find that the vocabulary of the Spanish language is the best represented in the train dataset.

6 Results and Analysis

In this section, we present the performance results on using architectures that we described in section 4 and try to give clue to understand them. Our main goal was to outperform the organizers' baseline model and results (Mickus, Timothee *et al.*, 2022).

For our first model (our baseline), the model is not better than the state-of-the-art models for this task. However, we can analyze an interesting point: this simple model surprisingly produces better results on the rank cosine (Rank) measure. To illustrate this remark, we can look the model results in table 2. On results for the MSE measure, only 3 cases outperform the organizers' baseline model. Moreover, every rank cosine measure is better. At this point, we can reach our first analyze, it's hard to perform in the MSE and Cosine (Cos) with, at the same time, trying to obtain good results in Rank (and vice-versa). This analyze is supported by the following table 3 based on advanced model.

Language	Emb-type	MSE	Cos	Rank
EN	char	**0.216**	0.709	0.449
	electra	1.638	**0.805**	0.433
	sgns	1.217	0.165	0.311
FR	char	0.501	0.690	0.428
	electra	1.394	**0.813**	0.441
	sgns	1.867	0.166	0.314
ES	char	0.632	0.787	0.411
	sgns	1.089	0.251	**0.253**
IT	char	0.691	0.572	0.417
	sgns	1.329	0.245	**0.246**
RU	char	**0.165**	0.787	0.409
	electra	0.946	0.694	0.398
	sgns	0.690	0.219	0.289

Table 2: Our Baseline model results.

Language	Emb-type	MSE	Cos	Rank
EN	char	**0.143**	0.795	0.500
	electra	1.326	**0.843**	0.500
	sgns	0.895	0.153	0.500
FR	char	0.365	0.769	0.500
	electra	1.112	**0.857**	0.500
	sgns	1.106	0.211	0.500
ES	char	0.510	0.824	0.500
	sgns	0.910	0.227	0.500
IT	char	0.358	0.728	0.500
	sgns	1.111	0.227	0.500
RU	char	**0.132**	0.829	0.500
	electra	0.864	0.719	0.500
	sgns	0.566	0.298	0.425

Table 3: Advanced model results.

With our second model, the results are completely opposite. We performed in MSE and Cosine measure. With these two measures, we're doing better than the organizers' baseline model. On the other hand, the Rank cosine seems to be stuck on 0.5. We can also compare our results with the other participants. Our BiLSTM-LSTM architecture is efficient on 'char' and 'electra' embeddings. For example, in 'char' with English, French and Spanish languages, we obtain the second-best score over the seven participants in SemEval-2022 campaign at task 1. We can conclude this analyze for the advanced architecture with this open-ended question: Why our architecture performs on English, French and Spanish but seam to give worse results on the Italian and Russian languages?

As we mentioned earlier, we tried to create a multilingual model. Unfortunately, after trained this model on all five languages and test on French Character embeddings, the model gave us poor results: 0.67 for MSE and 0.48 for Cosine measure. These are the worst results we've had in this competition, so we decided to drop this architecture and focus on the models presented in the system overview section.

Given the set of results obtained, we find that the best cosine score was obtained by using electra (contextualized) vector embeddings and the best MSE score was obtained by using character vector embeddings. More generally, the use of BiLSTM-LSTM architecture neural network has been beneficial in having results that surpass baselines when cosine and MSE are used as evaluation measures.

7 Conclusion

In this paper, we have presented our contributions to solve the task 1 problem of the semeval-2022 evaluation campaign. We studied the effects of training sentence embeddings with supervised data by testing on five different languages, namely: English, French, Spanish, Italian and Russian language. We showed that models learned with char embeddings or contextualized embeddings can perform better than models learned with Skip-Gram word embeddings. By exploring various architectures, we showed that the combination of Embedding/Dense/BiLSTM/ Dense/LSTM layers can be beneficial than the simple use of Embedding layer.

We believe that the neural architecture of our advanced model can be used to solve other tasks such as Definition Modeling (Noraset *et al.*, 2017), where the objective would be to reverse the inputs/outputs of the model, or other natural language processing tasks where the objective is to add a specific output layer to adopt the specific problem like sequence classification, for example.

Acknowledgments

We would like to thank Berger-Levrault for giving us the opportunity to participate in the 2022 edition of SemEval. Our thanks also go to the research and technological innovation department (DRIT) which is an important part of Berger-Levrault.

References

Arora, S., Liang, Y. and Ma, T. (2017) 'A Simple but Tough-to-Beat Baseline for Sentence Embeddings', in *International Conference on Learning Representations (ICLR)*.

Augustyniak, L., Kajdanowicz, T. and Kazienko, P. (2019) 'Aspect Detection using Word and Char Embeddings with (Bi)LSTM and CRF', *CoRR*, abs/1909.01276. Available at: http://arxiv.org/abs/1909.01276.

Bengio, Y. et al. (2003) 'A Neural Probabilistic Language Model', *J. Mach. Learn. Res.*, 3, pp. 1137–1155.

Chollet, F. and others (2015) Keras. GitHub. Available at: https://github.com/fchollet/keras.

Devlin, J. et al. (2018) 'Bert: Pre-training of deep bidirectional transformers for language understanding', arXiv preprint arXiv:1810.04805 [Preprint].

Hedderich, M.A. et al. (2019) 'Using Multi-Sense Vector Embeddings for Reverse Dictionaries', arXiv:1904.01451 [cs] [Preprint]. Available at: http://arxiv.org/abs/1904.01451 (Accessed: 17 February 2022).

Hill, F. et al. (2016) 'Learning to Understand Phrases by Embedding the Dictionary', *Transactions of the Association for Computational Linguistics*, 4, pp. 17–30. doi:10.1162/tacl_a_00080.

Liu, Y. et al. (2020) 'Depth-Adaptive Graph Recurrent Network for Text Classification', CoRR, abs/2003.00166. Available at: https://arxiv.org/abs/2003.00166.

Lwin, S.S. and Nwet, K.T. (2019) 'Extractive Myanmar News Summarization Using Centroid Based Word Embedding', in *2019 International Conference on Advanced Information Technologies (ICAIT)*, pp. 200–205. doi:10.1109/AITC.2019.8921386.

Malekzadeh, A., Gheibi, A. and Mohades, A. (2021) 'PREDICT: Persian Reverse Dictionary', arXiv:2105.00309 [cs] [Preprint]. Available at: http://arxiv.org/abs/2105.00309 (Accessed: 24 February 2022).

Medsker, L.R. and Jain, L.C. (2001) 'Recurrent neural networks', *Design and Applications*, 5, pp. 64–67.

Mickus, T. et al. (2022) 'SemEval-2022 Task 1: CODWOE -- COmparing Dictionaries and WOrd Embeddings', in *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mikolov, T. et al. (2013) 'Efficient estimation of word representations in vector space', arXiv preprint arXiv:1301.3781 [Preprint].

Morinaga, Y. and Yamaguchi, K. (2020) 'Improvement of Neural Reverse Dictionary by Using Cascade Forward Neural Network', *Journal of Information Processing*, 28(0), pp. 715–723. doi:10.2197/ipsjjip.28.715.

Noraset, T. et al. (2017) 'Definition Modeling: Learning to Define Word Embeddings in Natural Language', in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. AAAI Press (AAAI'17)*, pp. 3259–3266.

Pennington, J., Socher, R. and Manning, C.D. (2014) 'Glove: Global Vectors for Word Representation.', in *EMNLP*, pp. 1532–1543.

Qi, P. et al. (2020) 'Stanza: A Python Natural Language Processing Toolkit for Many Human Languages', CoRR, abs/2003.07082. Available at: https://arxiv.org/abs/2003.07082.

Sherstinsky, A. (2018) 'Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network', CoRR, abs/1808.03314. Available at: http://arxiv.org/abs/1808.03314.

Yan, H., Li, X. and Qiu, X. (2020) 'BERT for Monolingual and Cross-Lingual Reverse Dictionary', arXiv:2009.14790 [cs] [Preprint]. Available at: http://arxiv.org/abs/2009.14790 (Accessed: 17 February 2022).

Zhang, L. et al. (2019) 'Multi-channel Reverse Dictionary Model', arXiv:1912.08441 [cs] [Preprint]. Available at: http://arxiv.org/abs/1912.08441 (Accessed: 17 February 2022).

JSI at SemEval-2022 Task 1: CODWOE - Reverse Dictionary: Monolingual, multilingual, and cross-lingual approaches

Hanh Thi Hong TRAN[1,2,3], **Matej MARTINC**[1], **Matthew PURVER**[4], **Senja POLLAK**[1]
[1]Jožef Stefan Institute, Slovenia
[2]Jozef Stefan International Postgraduate School, Slovenia
[3]University of La Rochelle, France
[4]Queen Mary University of London, UK

Abstract

The *reverse dictionary* is a sequence-to-vector task in which a gloss is provided as input, and the model is trained to output a semantically matching word vector. The reverse dictionary is useful in practical applications such as solving the tip-of-the-tongue problem, helping new language learners, etc. In this paper, we evaluate the Transformer-based model with the added LSTM layer for the task at hand in a monolingual, multilingual, and cross-lingual zero-shot setting. Experiments are conducted in five languages in the CODWOE dataset, namely English, French, Italian, Spanish, and Russian. Our work partially improves the current baseline of the CODWOE competition and offers insight into the feasibility of the cross-lingual methodology for the reverse dictionary task. The code is available at https://github.com/honghanhh/codwoe2021.

1 Introduction

The CODWOE 2021 shared task on dictionary glosses and word embedding representations, organized as part of the SemEval workshop, presented one of the first opportunities to systematically study and compare these semantic descriptions by two sub-tracks: model definition and reverse dictionary.

While definition modeling consists in using the vector representation of e.g. “giraffe” to produce the associated gloss, e.g. “a tall, long-necked, spotted ruminant of Africa”, the reverse dictionary is the mathematical inverse: reconstruct an embedding for the word “giraffe” from the corresponding gloss. In this paper, we dive into the reverse dictionary task modelling to learn the ability to infer word embeddings from dictionary resources.

A reverse dictionary is useful in real-world applications. First of all, it can effectively solve the tip-of-the-tongue problem (Brown and McNeill, 1966): the inability to retrieve a word from memory. People who suffer from this problem such as copywriters, novelists, researchers, students, etc. can quickly and easily find the words they need thanks to reverse dictionary. Furthermore, new language learners who grasp a limited number of words can also take advantage of the reverse dictionary to express correctly. Besides, it plays an important role in word selection for anomia patients (Benson, 1979), who can recognize and describe an object but fail to name it due to neurological disorder.

The contributions of this paper are as follows:

1. We evaluate the performance of the Transformer-based model with an additional LSTM, BiLSTM, and the combination of both additional layers on separate languages as well as the performance of a multilingual model trained on the concatenated corpus containing text for all five given languages.

2. We analyze the effectiveness of zero-shot learning by training the model on a particular language and apply it for prediction on the rest.

This paper is organised as follows: Section 2 presents the related works in reverse dictionary. Next, we introduce our methodology in Section 3, and the experimental details in Section 4. The results are discussed in Section 5, before we conclude and present future works in Section 6.

2 Related Work

The reverse dictionary systems tend to employ two distinct approaches. The first approach takes advantage of sentence matching (Bilac et al., 2004; Zock and Bilac, 2004; Méndez et al., 2013; Shaw et al., 2011) to return the words whose dictionary definitions are most similar to the corresponding gloss.

The second approach focuses on neural language models to encode the glosses into a vector representation and returns the words with the closest embeddings to the vector of the glosses (Hill et al.,

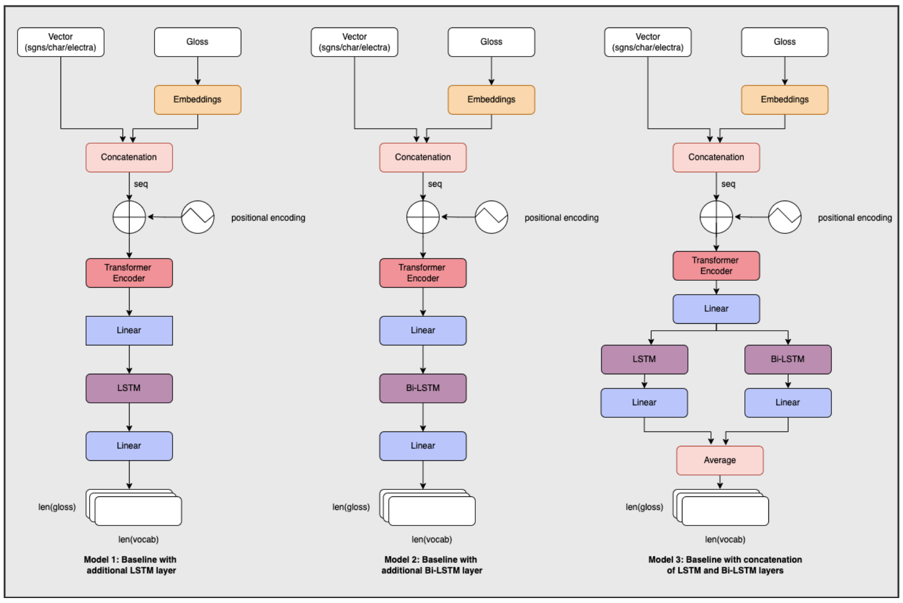

Vector (sgns/char/electra)
Gloss
Embeddings
Concatenation
seq
positional encoding
Transformer Encoder
Linear
LSTM
Bi-LSTM
Average
len(gloss)
len(vocab)
Model 1: Baseline with additional LSTM layer
Model 2: Baseline with additional Bi-LSTM layer
Model 3: Baseline with concatenation of LSTM and Bi-LSTM layers

Figure 1: The overall model architecture.

2016; Kartsaklis et al., 2018; Morinaga and Yamaguchi, 2018; Hedderich et al., 2019; Pilehvar, 2019). As a result, the performance depends largely on the word representation's quality. However, many words are low-frequency and usually have poor embeddings regarding Zipf's law.

To tackle the above issue, a multi-channel reverse dictionary model has been proposed (Zheng et al., 2020; Qi et al., 2020). The system includes a sentence encoder (e.g. a BiLSTM (Hochreiter and Schmidhuber, 1997), BERT (Devlin et al., 2018)) with attention (Bahdanau et al., 2014), and diverse characteristic predictors that are useful to find the target words with poor representations and exclude wrong words with similar embeddings to the target words, for example, antonyms.

In terms of production, OneLook[1] and ReverseDictionary[2] are two successful commercial English reverse dictionary systems. However, their architectures are undisclosed and their performance is far from perfect. Meanwhile, open-sourced WantWords[3] (Qi et al., 2020) is a rising star with state-of-the-art (SOTA) performance in English and even competitive results in a cross-lingual Chinese-English and English-Chinese setting.

3 Methodology

As the competition does not allow the use of external data or pretrained language models in order to make approaches easily comparable, we start by experimenting with the simplest form of Transformer, a deep learning model that adopts the self-attention mechanism, differentially weighting the significance of each part of the input data. This is also the baseline shared by CODWOE's organizers. Then we experiment by adding an additional LSTM layer (Model 1), BiLSTM layer (Model 2), and combining the prediction from these two mentioned layers (Model 3). The overall architecture is presented in Figure 1.

The objective of the model is to map the glosses to the vector representation of the word that the gloss defines. The target embeddings are learned by a skip-gram with negative sampling (sgns) approach (word2vec). During training, the input is the gloss, which is tokenized using the Byte Pair Encoding (BPE) algorithm[4] and then converted into word embeddings. The positional encoding is applied to each embedding to inject meaningful information about the position of the tokens in the sequence. After that, they are fed into a Transformer Encoder, which is a stack of four identical encoder blocks. As illustrated in Figure 2, each block includes the following layers in the same order: a multi-head self-attention layer that explores the word correlations followed by a normalization layer (both of them are surrounded by a residual connection), and then a linear layer followed by a second normalization layer (both of them are also

[1]https://onelook.com/thesaurus/
[2]https://reversedictionary.org/
[3]https://wantwords.thunlp.org/

[4]We employ the SentencePiece library: https://github.com/google/sentencepiece

surrounded by a residual connection). A dropout layer is then added to avoid overfitting. In the baseline model suggested by the CODWOE's organizers, the results from the above architecture are then passed into a linear layer to achieve the final model.

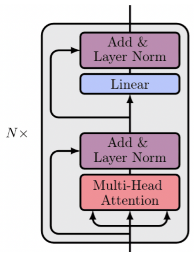


Figure 2: Transformer encoder (Vaswani et al., 2017).

We propose three settings regarding three different models constructed from the baseline architecture. We hypothesize that with an additional LSTM or BiLSTM layer, we can improve the modeling of the word-level sequential context, same as in (Wang et al., 2019), and therefore improve the performance of the model. In Model 1, we add one additional LSTM layer after the linear one. We take advantage of the BiLSTM layer in Model 2 to capture the information bidirectionally. We combine the result from the two mentioned layers by averaging their weights in Model 3. In the final step, we fed the LSTM or BiLSTM outputs into a linear layer to obtain the final vector representation. During the prediction phase, for each new data example, we feed the gloss into the trained model to obtain the vector presentation similar to the sgns.

The proposed three models are first tested in a monolingual setting, to determine which architecture achieves the best performance. Next, we explore if the target sgns embedding spaces may already be aligned to some degree across languages, even though the CODWOE organizers did not explicitly mention any cross-lingual alignment in the shared task description. We first attempt a multilingual experiment to examine the degree to which training in multiple languages affects performance. Finally, the best performing monolingual models are tested in a zero-shot cross-lingual setting, where we train the model in a specific language and evaluate it in different languages that the model has never seen before. The implementation details are in Section 4.2.

4 Experimental Setup

4.1 Dataset

The experiments were conducted on the dataset from the CODWOE 2021 competition. The data consists of glosses for five languages (English - **en**, Spanish - **es**, French - **fr**, Italian - **it**, and Russian - **ru** and three different word embedding representations for each gloss. In this paper, we focus only on skip-gram with negative sampling (sgns) embeddings trained on around 1 billion sentences in total with 50% of the sentences coming from Wikipedia, 40% coming from open subtitles, and the rest drawn from the corpora (e.g. Wikisource, gutenberg.org). All sentences were tokenized with the default NLTK's[5] tokenizer.

Each language contains 3 different sets, including the training set with 43,608 samples, the development set with 6,375 samples, and a test set containing 6,208 samples. Although the number of samples for each set is distributed equally among languages, a word can have a different number of glosses (polysemy), and vice versa, a gloss can belong to more than one word (synonymy).

Note that the training and development data hide the exact words matching each gloss and only release their sngs, char, and electra embeddings. However, on the full test set, the words are provided.

4.2 Experimental Settings

Due to time limitations, we have not conducted any hyperparameter search on the development sets over the space of possible model configurations, such as embedding dimension, learning rate, weight decay, size of hidden layers, etc. Alternatively, we decided to use a standard configuration based on previous research as well as suggested by the competition organizers for all the experiments. The configuration is presented in Table 1.

All models were implemented with Pytorch and trained on GPUs from Google Colab[6]. Further tuning and optimization will be left for future work.

4.3 Evaluation Metrics

The performance of the reverse dictionary system is evaluated by Mean squared error (MSE), Cosine similarity, and Cosine-based ranking (Dinu and Ionescu, 2012). These are the evaluation metrics suggested in the CODWOE 2021 competition,

[5]`https://www.nltk.org/`
[6]`https://colab.research.google.com/`

Table 1: Model configuration.

Settings	Values
Number of heads	4
Number of encoder layers	4
Number of epoches	20
Learning rate	1e-4
Weight decay	1e-6
Drop out	0.3
Optimizer	AdamW
Max length	512
Patience	5

which hereby facilitates the comparison between our approaches and the baseline. Further details about each evaluation metric can be found on the CODWOE 2021 website. Here, in this research, we aim to minimize the MSE and the cosine-based ranking, and maximize the cosine similarity.

5 Results

The test set results of our approach on the reverse dictionary task are presented in Table 2. We compare our three different models (LSTM, BiLSTM, and combined) with the baseline as well as with the winning approach on this shared task. In addition, we also present the results for a multilingual LSTM trained in all available languages.

In terms of MSE, the performance of the Transformer-based model with an additional LSTM layer is the most competitive for all languages except English when compared to our other approaches, namely BiLSTM and combined LSTM and BiLSTM. This model surpasses the baseline in Spanish and French according to most criteria. Meanwhile, the combination of the LSTM and BiLSTM layers after the Transformer encoder layer offers the best results on the English dataset, outperforming the baseline in terms of MSE. We also investigate a multilingual configuration where we train in all languages and employ the model on each language's test set. The results for the multilingual model are substantially lower compared to all other monolingual settings according to the MSE score. Compared to the best solution in the CODWOE competition proposed by WENGSYX team[7], the gap between our solution and theirs is on average 0.1 in terms of the MSE score.

In terms of Cosine similarity, the model with an additional LSTM layer proves to have better performance in English, Spanish, and French compared

[7]https://competitions.codalab.org/competitions/34022#results

Table 2: The evaluation results on the test dataset. We compare our models with additional LSTM, BiLSTM and combined LSTM and BiLSTM with the shared task baseline and the winning approach. We also test our multilingual approach trained on all languages of the train set. All the results above the baseline are in bold.

Language	Model	MSE	Cosine	Ranking
en	LSTM	0.913	**0.156**	0.499
en	BiLSTM	0.938	0.125	0.517
en	combined	**0.909**	0.139	0.513
en	multilingual LSTM	1.184	0.003	0.501
en	Baseline	0.911	0.151	0.490
en	#1 solution	**0.862**	**0.243**	**0.329**
es	LSTM	**0.914**	**0.223**	**0.499**
es	BiLSTM	1.031	0.005	**0.498**
es	combined	0.947	0.138	**0.495**
es	multilingual LSTM	0.978	**0.207**	**0.452**
es	Baseline	0.930	0.204	0.499
es	#1 solution	**0.858**	**0.353**	**0.251**
fr	LSTM	**1.123**	**0.216**	0.498
fr	BiLSTM	1.283	0.010	0.502
fr	combined	1.169	0.093	0.498
fr	multilingual LSTM	1.404	-0.005	0.524
fr	Baseline	1.140	0.198	0.491
fr	#1 solution	**1.030**	**0.328**	**0.282**
it	LSTM	1.201	-0.010	0.500
it	BiLSTM	1.287	-0.004	0.501
it	combined	1.208	-0.008	0.500
it	multilingual LSTM	1.305	-0.008	0.494
it	Baseline	1.125	0.204	0.477
it	#1 solution	**1.040**	**0.360**	**0.230**
ru	LSTM	0.616	0.006	0.500
ru	BiLSTM	0.795	-0.020	0.499
ru	combined	0.650	-0.016	0.499
ru	multilingual LSTM	0.934	-0.004	0.522
ru	Baseline	0.577	0.253	0.490
ru	#1 solution	**0.528**	**0.424**	**0.187**

to other tested models. This model also surpasses the baseline model on Spanish and French test sets. In addition, the multilingual model also achieves a slightly better Cosine similarity than the baseline on the Spanish test set.

In terms of Cosine ranking, all models demonstrate a slightly higher ranking in comparison to the baseline on the Spanish test set, with the multilingual model achieving the best ranking. In other languages, the baseline model performs the best.

Overall, training the additional LSTM layer on a multilingual training set does not seem to improve the results compared to the monolingual settings, the only exception being the performance of the multilingual model on the Spanish test set in terms of Cosine ranking.

Given the fact that the Transformer-based model with an additional LSTM performs the best in a monolingual setting, we use this model for the zero-shot cross-lingual experiments. The results

Table 3: Cross-lingual zero-shot evaluation on test set.

Train set	Metrics	en	es	fr	it	ru
en	MSE	**0.913**	0.914	1.208	1.201	0.616
	Cosine	**0.156**	**0.223**	-0.020	-0.010	**0.006**
	Ranking	**0.499**	**0.499**	0.500	0.500	**0.500**
es	MSE	0.963	0.914	1.208	1.201	0.616
	Cosine	-0.004	**0.223**	-0.020	-0.010	**0.006**
	Ranking	0.501	**0.499**	0.500	0.500	**0.500**
fr	MSE	0.962	0.916	**1.123**	**1.198**	**0.615**
	Cosine	-0.004	0.215	**0.216**	**-0.005**	0.002
	Ranking	0.500	**0.499**	**0.498**	**0.499**	0.501
it	MSE	0.962	0.916	1.208	1.201	**0.615**
	Cosine	-0.004	0.215	-0.024	-0.010	0.002
	Ranking	0.501	**0.499**	0.501	0.500	0.501
ru	MSE	0.964	**0.913**	1.204	1.196	0.616
	Cosine	-0.004	0.222	-0.021	-0.010	**0.006**
	Ranking	0.501	0.500	0.500	0.500	**0.500**

for these experiments are displayed in Table 3. The first column indicates the language used for training and development, the second column displays the evaluation metrics including MSE, Cosine similarity, and Cosine ranking. The rest demonstrate the evaluation results of each metric on a specific test dataset per language. For example, in the first row where the training set is *en*, we train on the English training and development set and predict each of the five language's test sets.

In general, if the model is trained on a language matching the language of the test data, it performs better except in the French corpus. However, the interesting exception is that, for example, the Spanish test set, on which all models, no matter on which language they were trained, offer very consistent performance according to all measures. It is also interesting that the models trained in English and Spanish have exactly the same results on French, Italian, and Russian test sets. This might suggest that these models were not able to make sense of the examples in the test set and that their performance is on par with a random baseline. Further analysis of this behavior will be left for the future.

6 Conclusion

In this paper, we have investigated the performance of monolingual and multilingual Transformer-based models on the reverse dictionary problem, a sequence-to-vector task where a word representation needs to be constructed from the corresponding gloss. We have experimented with two additions to the original architecture, namely adding either an additional LSTM or a BiLSTM layer on top of the original architecture. We have also explored whether combining these two architectures improves the performance. Besides that, we explored the cross-lingual performance of the monolingual models and compared them to monolingual and multilingual classifiers.

On the task of reconstructing sgns embeddings, the monolingual Transformer-based model with an additional LSTM layer in most cases offers the best performance for English, Spanish, and French according to MSE and Cosine similarity. The model also offers competitive performance in terms of MSE for Italian and Russian compared to the baseline. Therefore, the results to some extent confirm the initial hypothesis that with an additional LSTM layer, we can improve the modeling of the word-level sequential context. Nevertheless, the improvements are worse than expected and the multilingual and zero-shot experiments yield unexpected results that require further analysis. We can therefore summarize our findings by saying that the reverse dictionary task of restoring sgns embeddings seems to be very challenging, and none of our models (and also other models in the competition) were able to successfully solve it, at least according to the scores achieved during the competition.

This means that there remains a lot of room for improvement. In the future, we would like to investigate the effect of different text representations on the performance of the model, e.g., by feeding the model graph representations. Combinations of several text representations will also be explored. Furthermore, the effectiveness of multilingual models compared to monolingual ones should be additionally explored. Despite zero-shot learning not working well in our studies, it is worth evaluating the performance of one-shot learning and few-shot learning with the hypothesis that the models can understand new concepts from only one or a few examples. Further experiments on the topic of adapting the Transformer architecture for the specific task at hand will also be conducted.

Acknowledgements

We acknowledge the Slovenian Research Agency ARRS for the core programme funding P2-103 "Knowledge Technologies". We also thank the region of La Nouvelle Aquitaine for funding the first author of this research.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

D Frank Benson. 1979. Neurologic correlates of anomia. In *Studies in neurolinguistics*, pages 293–328. Elsevier.

Slaven Bilac, Wataru Watanabe, Taiichi Hashimoto, Takenobu Tokunaga, and Hozumi Tanaka. 2004. Dictionary search based on the target word description. In *Proc. of the Tenth Annual Meeting of The Association for Natural Language Processing (NLP2004)*, pages 556–559.

Roger Brown and David McNeill. 1966. The “tip of the tongue” phenomenon. *Journal of verbal learning and verbal behavior*, 5(4):325–337.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Liviu P Dinu and Radu-Tudor Ionescu. 2012. A rank-based approach of cosine similarity with applications in automatic classification. In *2012 14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pages 260–264. IEEE.

Michael A Hedderich, Andrew Yates, Dietrich Klakow, and Gerard De Melo. 2019. Using multi-sense vector embeddings for reverse dictionaries. *arXiv preprint arXiv:1904.01451*.

Felix Hill, KyungHyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

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

Dimitri Kartsaklis, Mohammad Taher Pilehvar, and Nigel Collier. 2018. Mapping text to knowledge graph entities using multi-sense lstms. *arXiv preprint arXiv:1808.07724*.

Oscar Méndez, Hiram Calvo, and Marco A Moreno-Armendáriz. 2013. A reverse dictionary based on semantic analysis using wordnet. In *Mexican International Conference on Artificial Intelligence*, pages 275–285. Springer.

Yuya Morinaga and Kazunori Yamaguchi. 2018. Improvement of reverse dictionary by tuning word vectors and category inference. In *International Conference on Information and Software Technologies*, pages 533–545. Springer.

Mohammad Taher Pilehvar. 2019. On the importance of distinguishing word meaning representations: A case study on reverse dictionary mapping. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2151–2156.

Fanchao Qi, Lei Zhang, Yanhui Yang, Zhiyuan Liu, and Maosong Sun. 2020. Wantwords: An open-source online reverse dictionary system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–181.

Ryan Shaw, Anindya Datta, Debra VanderMeer, and Kaushik Dutta. 2011. Building a scalable database-driven reverse dictionary. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):528–540.

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

Chenguang Wang, Mu Li, and Alexander J Smola. 2019. Language models with transformers. *arXiv preprint arXiv:1904.09408*.

Lei Zheng, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2020. Multi-channel reverse dictionary model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 312–319.

Michael Zock and Slaven Bilac. 2004. Word lookup on the basis of associations: from an idea to a roadmap. In *Proceedings of the Workshop on Enhancing and Using Electronic Dictionaries*, pages 29–35.

SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding

Harish Tayyar Madabushi[1], **Edward Gow-Smith**[1],
Marcos Garcia[2], **Carolina Scarton**[1],
Marco Idiart[3] and **Aline Villavicencio**[1]

[1] University of Sheffield, UK
[2] Universidade de Santiago de Compostela, Spain
[3] Federal University of Rio Grande do Sul, Brazil

{h.tayyarmadabushi, egow-smith1, c.scarton, a.villavicencio} @sheffield.ac.uk
marcos.garcia.gonzalez@usc.gal, marco.idiart@gmail.com

Abstract

This paper presents the shared task on *Multilingual Idiomaticity Detection and Sentence Embedding*, which consists of two Subtasks: (a) a binary classification task aimed at identifying whether a sentence contains an idiomatic expression, and (b) a task based on semantic text similarity which requires the model to adequately represent potentially idiomatic expressions in context. Each Subtask includes different settings regarding the amount of training data. Besides the task description, this paper introduces the datasets in English, Portuguese, and Galician and their annotation procedure, the evaluation metrics, and a summary of the participant systems and their results. The task had close to 100 registered participants organised into twenty five teams making over 650 and 150 submissions in the practice and evaluation phases respectively.

1 Introduction

Multiword Expressions (MWEs) are a challenge for natural language processing (NLP), as their linguistic behaviour (e.g., syntactic, semantic) differs from that of generic word combinations (Baldwin and Kim, 2010; Ramisch and Villavicencio, 2018). Moreover, MWEs are pervasive in all domains (Biber et al., 1999), and it has been estimated that their size in a speaker's lexicon of any language is of the same order of magnitude as the number of single words (Jackendoff, 1997; Erman and Warren, 2000), thus being of crucial interest for language modelling and for the computational representation of linguistic expressions in general.

One distinctive aspect of MWEs is that they fall on a *continuum* of idiomaticity (Sag et al., 2002; Fazly et al., 2009; King and Cook, 2017), as their meaning may or may not be inferred from one of their constituents (e.g., *research project* being a type of 'project', vs. *brass ring* meaning a 'prize'). In this regard, obtaining a semantic representation of a sentence which contains potentially idiomatic expressions involves both the correct identification of the MWE itself, and an adequate representation of the meaning of that expression in that particular context. As an example, it is expected that the representation of the expression *big fish* will be similar to that of *important person* in an idiomatic context, but closer to the representation of *large fish* when conveying its literal meaning.

Classic approaches to representing MWEs obtain a compositional vector by combining the representations of their constituent words, but these operations tend to perform worse for the idiomatic cases. In fact, it has been shown that the degree of idiomaticity of a MWE can be estimated by measuring the distance between a compositional vector (obtained from the vectors of its components) and a single representation learnt from the distribution of the MWE in a large corpus (Cordeiro et al., 2019).

Recent approaches to identify and classify MWEs take advantage of the contextualised representations provided by neural language models. On the one hand, some studies suggest that pre-training based on masked language modeling does not properly encode idiomaticity in word representations (Nandakumar et al., 2019; Garcia et al., 2021b,a). However, as these embeddings encode contextual information, supervised approaches using these representations tend to obtain better results in different tasks dealing with (non-)compositional semantics (Shwartz and Dagan, 2019; Fakharian and Cook, 2021; Zeng and Bhat, 2021).

As such, this shared task[1,2] presents two Subtasks: i) Subtask A, to test a language model's

[1]Task website: https://sites.google.com/view/semeval2022task2idiomaticity

[2]GitHub:https://github.com/H-TayyarMadabushi/SemEval_2022_Task2-idiomaticity

ability to detect idiom usage, and ii) Subtask B, to test the effectiveness of a model in generating representations of sentences containing idioms. Each of these Subtasks are further presented in two *settings*: Subtask A in the Zero Shot and One Shot settings so as to evaluate models on their ability to detect previously unseen MWEs, and Subtask B in the Pre Train and the Fine Tune settings to evaluate models on their ability to capture idiomaticity both in the absence and presence of training data. Additionally, we provide strong baselines based on pre-trained transformer-based language models and release our codetr which participants can build upon.

2 Related Tasks

The computational treatment of MWEs has been of particular interest for the NLP community, and several shared tasks with different objectives and resources have been carried out.

The SIGLEX-MWE Section[3] has organised various shared tasks, starting with the exploratory *Ranking MWE Candidates* competition at the MWE 2008 Workshop, aimed at ranking MWE candidates in English, German and Czech.[4] More recently, together with the PARSEME community, they have conducted three editions of a shared task on the automatic identification of verbal MWEs (Savary et al., 2017; Ramisch et al., 2018, 2020). In these cases, the objective is to identify both known and unseen verb-based MWEs in running text and to classify them under a set of predefined categories. Interestingly, these PARSEME shared tasks provide annotation guidelines and corpora for 14 languages, and include 6 categories (with additional subclasses) of verbal MWEs.

The *Detecting Minimal Semantic Units and their Meanings (DiMSUM 2016)* shared task (Schneider et al., 2016) consisted of the identification of *minimal semantic units* (including MWEs) in English, and labelling some of them according to a set of semantic classes (supersenses).

Focused on the interpretation of noun compounds, the *Free Paraphrases of Noun Compounds* shared task of SemEval 2013 (Hendrickx et al., 2013) proposed to generate a set of free paraphrases of English compounds. The paraphrases should be ranked by the participants, and the evaluation is performed comparing these ranks against a list of paraphrases provided by human annotators.

Similarly, the objective of the SemEval 2010 shared task on *The Interpretation of Noun Compounds Using Paraphrasing Verbs and Prepositions* (Butnariu et al., 2010) was to rank verbs and prepositions which may paraphrase a noun compound adequately in English (e.g., *olive oil* as 'oil *extracted from* olive', or *flu shot* as 'shot *to prevent* flu').

Apart from these competitions, various studies have addressed different tasks on MWEs and their compositionality, such as: classifying verb-particle constructions (Cook and Stevenson, 2006), identifying light verb constructions and determining the literality of noun compounds (Shwartz and Dagan, 2019), identifying and classifying idioms in running text (Zeng and Bhat, 2021), as well as predicting the compositionality of several types of MWEs (Lin, 1999; McCarthy et al., 2003; Reddy et al., 2011; Schulte im Walde et al., 2013; Salehi et al., 2015).

3 Dataset Creation

The dataset used in this task extends that introduced by Tayyar Madabushi et al. (2021), also including Galician data along with Portuguese and English. Here we describe the four step process used in creating this dataset.

The first step was to compile a list of 50 MWEs across the three languages. We sourced the MWEs in English and Portuguese from the Noun Compound Senses dataset (consisting of adjective-noun or noun-noun compounds) (Garcia et al., 2021b), which extends the dataset by Reddy et al. (2011) and provides human-judgements for compositionality on a Likert scale from 0 (non-literal/idiomatic) to 5 (literal/compositional). To ensure that the test set is representative of different levels compositionality, we pick approximately 10 idioms at each level of compositionality (0-1, 1-2, . . .). For Galician, we extracted noun-adjective compounds from the Wikipedia and the CC-100 corpora (Wenzek et al., 2020) using the following procedure: First, we identified those candidates with at least 50 occurrences in the corpus. They were randomly sorted, and a native speaker and language expert of Galician selected 50 compounds from the list. The language expert was asked to take into account both the compositionality of the compounds (including idiomatic, partly idiomatic, and literal expressions),

[3] https://multiword.org/

[4] http://multiword.sourceforge.net/mwe2008

and their ambiguity (trying to select potentially idiomatic examples, i.e. compounds which can be literal or idiomatic depending on the context).

In the second step of the dataset creation process, in English and Portuguese, annotators were instructed to obtain between 7 and 10 examples for each possible meaning of each MWE from news stories available on the web, thus giving between 20 and 30 total examples for each MWE. Each example consisted of three sentences: the target sentence containing the MWE and the two adjacent sentences. Annotators where explicitly instructed to select high quality examples, where neither of the two adjacent sentences were empty and, preferably, from the same paragraph. They were additionally required to flag examples containing novel meanings, so such new meanings of MWEs could be incorporated into the dataset. Sentences containing MWEs in Galician were directly obtained from the Wikipedia and the CC-100 corpora due to the sparsity of Galician data on the web. During this annotation step, we follow the method introduced by Tayyar Madabushi et al. (2021), and add two additional labels: 'Proper Noun' and 'Meta Usage'. 'Meta Usage' represents cases wherein a MWE is used literally, but within a metaphor (e.g. *life vest* in "Let the Word of God be our life vest to keep us afloat, so as not to drown.").

In the third phase, across all three languages, each possible meaning of each MWE was assigned a paraphrase by a language expert. For example, the compositional MWE *mailing list* had the associated paraphrase 'address list' added, whereas the idiomatic MWE *elbow room* had the associated paraphrases 'joint room', 'freedom' and 'space' added to correspond to each of its possible meanings. Language experts focused on ensuring that these paraphrases were as short as possible, so the resultant adversarial paraphrases could be used to evaluate the extent to which models capture nuanced differences in each of the meanings.

The final phase of the process involved the annotation of each example with the correct paraphrase of the relevant MWE. This was carried out by two annotators, and any disagreements were discussed (in the case of Galician, in the presence of a language expert) and cases where annotators were not able to agree were discarded.

3.1 The Competition Dataset

We use the training and development splits from Tayyar Madabushi et al. (2021) with the addition of Galician data, and use the test split released by them as the evaluation split during the initial practice phase of the competition. We create an independent test set consisting of examples with new MWEs, and this set was used to determine the teams' final rankings. The labels for the evaluation and test sets are not released. We note that the competition is still active (in the 'post-evaluation' phase), and open for submissions from anyone[5].

Since one of the goals of this task is to measure the ability of models to perform on previously unseen MWEs (Zero Shot) and on those for which they have very little training data (One Shot), we extract, where available, exactly one idiomatic and one compositional example associated with each MWE in the test data, which is released as associated One Shot training data.

The final dataset consisted of 8,683 entries and the breakdown of the dataset is shown in Table 1. For further details on the training, development and practice evaluation splits, we direct readers to the work by Tayyar Madabushi et al. (2021). It should be noted that this original dataset does not contain data from Galician and so the only training data available in Galician was the One Shot training data. This was to evaluate the ability of models to transfer their learning across languages, especially to one that is low resourced.

	Language			
Split	English	Portuguese	Galician	All
train	3487	1290	63	4840
dev	466	273	0	739
eval	483	279	0	762
test	916	713	713	2342
All	5352	2555	776	8683

Table 1: Breakdown of the full dataset by language and data split.

4 Task Description and Evaluation Metrics

SemEval-2022 Task 2 aims to stimulate research into a difficult area of NLP, that of handling non-compositional, or idiomatic, expressions. Since this is an area of difficulty for existing language

[5]https://competitions.codalab.org/competitions/34710

models, we introduce two Subtasks; the first Subtask relates to idiomaticity detection, whilst the second relates to idiomaticity representation, success in which will require models to correctly encode idiomaticity. It is hoped that these tasks will motivate the development of language models better able to handle idiomaticity. Since we wish to promote multilingual models, we require all participants to submit results across all three languages. Both Subtasks are available in two settings, and participants are given the flexibility to choose which settings they wish to take part in.

4.1 Subtask A: Idiomaticity Detection

The first Subtask is a binary classification task, where sentences must be correctly classified into 'idiomatic' (including 'Meta Usage') or 'non-idiomatic' / literal (including 'Proper Noun'). Each example consists of the target sentence and two context sentences (sourced from either side of the target sentence) along with the relevant MWE. Some examples from this Subtask are shown in Table 2.

This Subtask is available in two settings: Zero Shot and One Shot. In the Zero Shot setting, the MWEs in the training set are disjoint from those in the development and test sets. Success in this setting will require models to generalise to unseen MWEs at inference time. In the One Shot setting, we include in the training set one idiomatic and one non-idiomatic example for each MWE in the development and test sets. This breakdown is shown in Table 3.

We use macro F1 score between the gold labels and predictions as the evaluation metric for this Subtask, due to the imbalanced datasets.

4.2 Subtask B: Idiomaticity Representation

The second Subtask is a novel idiomatic semantic textual similarity (STS) task, introduced by Tayyar Madabushi et al. (2021), where, given two input sentences, models must return an STS score between 0 (least similar) and 1 (most similar), indicating the similarity of the sentences. This requires models to correctly encode the meaning of non-compositional MWEs (idioms) such that the encoding of a sentence containing an idiomatic phrase (e.g. "I initially feared that taking it would make me a **guinea pig**.") and the same sentence with the idiomatic phrase replaced by a (literal) paraphrase (e.g. "I initially feared that taking it would make me a **test subject**.") are semantically similar to each other. Notice also that these two sentences, which mean the same thing, must necessarily be equally similar to any other third sentence. We choose this third sentence to be the sentence with the idiomatic phrase replaced by an *incorrect* literal paraphrase (e.g. "I initially feared that taking it would make me a **pig**."). Such a sentence is the ideal adversarial example, and ensures that we test if models are making use of an incorrect meaning of the MWE in constructing a sentence representation.

Data for this Subtask is generated in the following manner: MWEs in sentences are replaced by the literal paraphrase of one of its associated meanings. For example, the MWE 'guinea pig' in the sentence "I initially feared that taking it would make me a *guinea pig*." is replaced by one of the literal paraphrases 'test subject' or 'pig' (see Table 4). Crucially, these replacements can either be with the correct paraphrase, or one that is incorrect. As such, there are two cases:

- The MWE has been replaced by its correct paraphrase. In this case, the similarity should be 1.
  $sim(E, E_{\rightarrow \text{c}}) = 1$
- The MWE has been replaced by its incorrect paraphrase. In this case, we require the model to give equivalent semantic similarities between this and the sentence where the MWE has been replaced by its correct paraphrase, and this and the original sentence.
  $sim(E, E_{\rightarrow \text{i}}) = sim(E_{\rightarrow \text{c}}, E_{\rightarrow \text{i}})$

Importantly, the task requires models to be *consistent*. Concretely, the STS score for the similarity between a sentence containing an idiomatic MWE and that same sentence with the MWE replaced by the correct paraphrase must be equal to *one* as this would imply that the model has correctly interpreted the meaning of the MWE. In the case where we consider the incorrect paraphrase, we check for consistency by requiring that the STS between the sentence containing the MWE and a sentence where the MWE is replaced by the incorrect paraphrase is equal to the STS between the sentence where the MWE is replaced by the correct paraphrase and one where it is replaced by the incorrect one. Notice, that all this does, is to require the model to, once again, interpret the meaning of the MWE to be the same (or very similar) to the correct literal paraphrase of that MWE. More formally, we require models to output STS scores for each example E such that:

Language	MWE	Sentence	Label
English	old hat	Serve our favorite bourbon whiskeys in an **old hat** and we'd still probably take a sip or two.	1
English	old hat	But not all of the accouterments of power are **old hat** for the president.	0
Portuguese	força bruta	**Força Bruta** vai reunir alguns dos homens mais fortes do mundo.	1
Portuguese	força bruta	Gardner é conhecido por ser impulsivo e usar os poderes com grande impacto, de forma instintiva, com **força bruta**.	0
Galician	porta grande	Á esquerda da **porta grande**, en terra, observamos a tumba de "Don Manuel López Vizcaíno.	1
Galician	porta grande	Os dous dominadores da Copa Galicia 2017 regresaron pola **porta grande** ao certame autonómico na súa quinta xornada.	0

Table 2: Examples for Subtask A. Note that the label 1 is assigned to non-idiomatic usage, which includes proper nouns, as in the Portuguese example.

		Language			
Train Split	MWEs	English	Portuguese	Galician	All
Zero Shot	236	3327	1164	0	4491
One Shot	250	160	126	63	349
Total	486	3487	1290	63	4840

Table 3: Breakdown of the training data into zero shot and one shot. Note that the MWEs in the zero shot and one shot data are disjoint.

$$\forall_{i \in I} \Big(sim(E, E_{\rightarrow \mathrm{c}}) = 1; \\ sim(E, E_{\rightarrow \mathrm{i}}) = sim(E_{\rightarrow \mathrm{c}}, E_{\rightarrow \mathrm{i}}) \Big) \quad (1)$$

In Equation 1 above, $E_{\rightarrow \mathrm{c}}$ represents an example containing the MWE E, wherein that MWE is replaced by its *correct* contextual paraphrase. $E_{\rightarrow \mathrm{i}}$ on the other hand, represents the example wherein the MWE E is replaced by one of its *incorrect* contextual paraphrases. Examples for this Subtask are shown in Table 4.

Since this task relies on models' ability to correctly assign STS scores for sentences with do not contain idiomatic MWEs, we additionally include standard STS data in our test data. This has the added benefit of preventing models from overfitting on the MWE dataset. We include this STS evaluation data from the STS Benchmark dataset (Cer et al., 2017) in English and the ASSIN2 STS dataset (Real et al., 2020) in Portuguese. There is no available STS data for Galician, so none is included. We use the Spearman's rank correlation coefficient between the two sets of STS scores generated by models as the evaluation metric in this Subtask. We do not use Pearson correlation as it has been shown to be a poor indicator of performance on STS tasks (Reimers et al., 2016).

This Subtask is also available in two settings: the Pre Train setting and the Fine Tune setting. In the Pre Train setting, we require that models are *not* trained on idiomatic STS data. However, models can be trained (including "fine-tuned") on any other training objective (such as during the pre-training of language models). The Fine Tune setting, on the other hand, allows all training regimes, including the fine-tuning on any idiomatic STS dataset.

4.3 Baselines

In order to generate baseline results, we used pre-trained transformer-based (Vaswani et al., 2017) language models. We use multilingual BERT (Devlin et al., 2019) to benefit from cross-lingual transfer. For both settings in Subtask A, we simply Fine Tune the pre-trained model on the training data provided. For the Zero Shot setting, we include the context sentences, whereas in the One Shot setting, we exclude the context sentences but add the MWE as a second sentence. This is based on the best-performing approaches found by Tayyar Madabushi et al. (2021).

For Subtask B Pre Train, we introduce single tokens for each MWE in the data. This is motivated by the 'idiom principle' (Sinclair and Sinclair, 1991), which hypothesises that humans process idioms by treating them as a single unit. Since BERT embeddings cannot be directly used for STS, we create a sentence transformer model (Reimers and Gurevych, 2019) using multilingual BERT with these added tokens, and train it on the English and Portuguese STS data. Importantly, the new tokens introduced for MWEs are randomly initialised and no continued pre-training is performed. As such, they serve to 'break compositionality' rather than to create more effective representations of MWEs. This breaking of compositionality has been shown to be effective by Tayyar Madabushi et al. (2021).

For the Fine Tune setting, the same approach is taken, although no training is done on the STS

Sentence (E)	Correct Replacement ($E_{\text{MWE}\rightarrow\text{c}}$)	Wrong Replacement ($E_{\text{MWE}\rightarrow\text{i}}$)	Expected
And finally, the snow falls again, this time in a thick, **wet blanket** that encapsulates everything.	And finally, the snow falls again, this time in a thick, **damp blanket** that encapsulates everything.	And finally, the snow falls again, this time in a thick, **killjoy** that encapsulates everything.	$sim(E, E_{\rightarrow\text{c}}) = 1$ $sim(E, E_{\rightarrow\text{i}}) = sim(E_{\rightarrow\text{c}}, E_{\rightarrow\text{i}})$
I initially feared that taking it would make me a **guinea pig**.	I initially feared that taking it would make me a **test subject**.	I initially feared that taking it would make me a **pig**.	$sim(E, E_{\rightarrow\text{c}}) = 1$ $sim(E, E_{\rightarrow\text{i}}) = sim(E_{\rightarrow\text{c}}, E_{\rightarrow\text{i}})$

Table 4: Examples for Subtask B. For brevity we only include examples in English.

data, and instead we Fine Tune on the training data provided. This lack of training on the STS data is intentional as we intend to establish the effectiveness of the MWE based training data, and are reflected by the comparatively lower scores on the STS subsection of the test data (Table 8).

It should be noted that these baseline methods that make use of multilingual BERT are particularly strong when compared to typical 'baselines'. This is intentional as we aim to promote the development of models that are comparable to the current state-of-the-art.

5 Participating Systems and Results

Twenty five teams in total participated, with the most participants to Subtask A Zero Shot (20). The results for the individual Subtasks are given in Table 5, Table 6, Table 7 and Table 8. Here we discuss the methods used by the best-performing teams as well as some interesting approaches. Full details of methods used by participants is given in Appendix A.

5.1 Subtask A Zero-Shot

Of the twenty teams that submitted to this setting, 12 reported using transformer-based approaches. The best-performing team (clay) used different masking strategies during pretraining, and performed finetuning with data augmentation (including back-translation, Edunov et al., 2018) as well as using soft-label finetuning (a knowledge distillation approach). The team in second (yxb) used a multilingual T5 model (Xue et al., 2021) with various data augmentation techniques including: back-translation; synonym replacement; random insertion, swap, and deletion. They also used an alternative loss function for unbalanced data, called focal loss (Lin et al., 2017). The third team (NER4ID; Tedeschi and Navigli, 2022) used a dual-encoder architecture to encode the MWE and its context, then predicted idiomaticity by looking at the similarity score. This approach has a precedent in previous work that hypothesises the semantic similarity between a MWE and its context to be a good indicator of idiomaticity (Liu and Hwa, 2018). They also implemented named entity recognition as an intermediate step which they found provided great improvements. Interestingly, two teams (UAlberta; Hauer et al., 2022, and Unimelb_AIP) used unsupervised approaches, i.e. not using any of the provided training data. UAlberta were able to beat the baseline using translation information from resources such as Open Multilingual Wordnet (Bond and Foster, 2013) and BabelNet (Navigli and Ponzetto, 2010). They hypothesised that for idiomatic MWEs, the individual words are less likely to share mult-synsets with their translations. They also used a POS tagger for identifying proper nouns.

5.2 Subtask A One Shot

The best-performing team (HIT; Chu et al., 2022) used XLM-R (Conneau et al., 2020), and added '[SEP]' tokens around the relevant MWE in the target sentence, unless it was capitalised, in which case they excluded these tokens. This is an alternative approach to that of Tayyar Madabushi et al. (2021), where the MWEs were added as a second sentence. They also used R-Drop (Wu et al., 2021) as a regularisation method. The second best team (kpfriends; Sik Oh, 2022) used an ensemble of checkpoints with soft-voting. They also started with XLM-RoBERTa (large) trained on CoNLL. Interestingly, this team had the largest difference in performance across the two settings of Subtask A (coming in 16th in the Zero Shot setting). The third best team (UAlberta; Hauer et al., 2022) used a transformer-based classifier with additional features of glosses for the individual words of the relevant MWE. They hypothesised that this would help for determining compositionality, since the meaning of compositional MWEs could be deduced from the glosses of the individual words. An interesting approach was taken by MaChAmp (van der Goot, 2022), who used multi-task learning across multiple SemEval tasks (2, 3, 4, 6, 10, 11, 12), pretrain-

Ranking	Team	Language			All
		English	Portuguese	Galician	
1	clay	0.9016	0.8277	0.9278	0.8895
2	yxb	0.8948	0.8395	0.7524	0.8498
3	NER4ID (Tedeschi and Navigli, 2022)	0.8680	0.7039	0.6550	0.7740
4	HIT (Chu et al., 2022)	0.8242	0.7591	0.6866	0.7715
5	Hitachi (Yamaguchi et al., 2022)	0.7827	0.7607	0.6631	0.7466
6	OCHADAI (Pereira and Kobayashi, 2022)	0.7865	0.7700	0.6518	0.7457
7	yjs	0.8253	0.7424	0.6020	0.7409
8	CardiffNLP-metaphors (Boisson et al., 2022)	0.7637	0.7619	0.6591	0.7378
9	Mirs	0.7663	0.7617	0.6429	0.7338
10	Amobee	0.7597	0.7147	0.6768	0.7250
11	HYU (Joung and Kim, 2022)	0.7642	0.7282	0.6293	0.7227
12	Zhichun Road (Cui et al., 2022)	0.7489	0.6901	0.5104	0.6831
13	海鲛NLP	0.7564	0.6933	0.5108	0.6776
14	UAlberta (Hauer et al., 2022)	0.7099	0.6558	0.5646	0.6647
15	Helsinki-NLP (Itkonen et al., 2022)	0.7523	0.6939	0.4987	0.6625
16	daminglu123 (Lu, 2022)	0.7070	0.6803	0.5065	0.6540
	baseline (Tayyar Madabushi et al., 2021)	0.7070	0.6803	0.5065	0.6540
17	kpfriends (Sik Oh, 2022)	0.7256	0.6739	0.4918	0.6488
18	Unimelb_AIP	0.7614	0.6251	0.5020	0.6436
19	YNU-HPCC (Liu et al., 2022)	0.7063	0.6509	0.4805	0.6369
20	Ryan Wang	0.5972	0.4943	0.4608	0.5331
N/A	JARVix (Jakhotiya et al., 2022)[6]	0.7869	0.7201	0.5588	0.7235

Table 5: Results for Subtask A Zero Shot. The evaluation metric is macro F1 score, and the ranking is based on the 'All' column.

ing a Rebalanced mBERT (RemBERT) (Chung et al., 2020) model across all of the tasks, then retraining a model for each specific task. Since for this task we do not allow the use of additional data, we do not include this team in the ranking, but their score is reported for reference.

5.3 Subtask B Pre Train

No teams reported using non-transformer-based approaches for this setting. The best-performing team (drsphelps; Phelps, 2022) used a modification of the baseline with BERT for Attentive Mimicking (BERTRAM) (Schick and Schütze, 2020) to generate embeddings as replacements for the randomly-initialised one token embeddings used by the baseline. This method takes both form and context into account, thus not assuming total non-compositionality as the one-token method does. It should be noted that every team in this setting improved upon the baseline result.

5.4 Subtask B Fine Tune

No teams reported using non-transformer-based approaches for this setting. The best-performing team (YNU-HPCC; Liu et al., 2022) used a pre-trained Sentence-BERT (Reimers and Gurevych, 2019) model, then finetuned using multiple negatives ranking loss (Henderson et al., 2017) and triplet loss. The second best team (drsphelps; Phelps, 2022) used an identical approach to that in Subtask B Pre Train, using BERTRAM (Schick and Schütze, 2020), with additional finetuning on the training data provided. The third best team (Eat Fish) used a multilingual model pretrained with knowledge distillation, as well as data augmentation.

[6]Not ranked due to only submitting to the 'post-evaluation' phase.

5.5 Overview of Submissions

In Figure 1 we show the models that were mentioned in the submissions.

The majority of participants used transformer-based approaches, although in both settings for Subtask A there were three teams using other approaches. In Subtask B, as mentioned previously, no non-transformer approaches were mentioned, which is expected since this task was designed for the pretrain-finetune paradigm.

In Figure 2 we show the methods mentioned in more than one submission. Data augmentation approaches were popular, the most frequently-mentioned being back-translation (Edunov et al., 2018). Equally as popular were approaches using

Ranking	Team	Language			All
		English	Portuguese	Galician	
1	HIT (Chu et al., 2022)	0.9639	0.8944	0.9369	0.9385
2	kpfriends (Sik Oh, 2022)	0.9606	0.8993	0.9215	0.9346
3	UAlberta (Hauer et al., 2022)	0.9453	0.8918	0.9120	0.9243
4	Zhichun Road (Cui et al., 2022)	0.9344	0.8559	0.8927	0.9033
5	clay	0.9181	0.8423	0.9313	0.9022
6	YNU-HPCC (Liu et al., 2022)	0.9179	0.8633	0.8781	0.8948
7	CardiffNLP-metaphors (Boisson et al., 2022)	0.9464	0.8385	0.8545	0.8934
8	yxb	0.8995	0.8266	0.8781	0.8779
9	NER4ID (Tedeschi and Navigli, 2022)	0.9079	0.8179	0.8695	0.8771
10	HYU (Joung and Kim, 2022)	0.9159	0.8457	0.8287	0.8750
11	yjs	0.9199	0.8365	0.8294	0.8747
	baseline (Tayyar Madabushi et al., 2021)	0.8862	0.8637	0.8162	0.8646
12	Mirs	0.7570	0.7549	0.6712	0.7367
13	daminglu123 (Lu, 2022)	0.7486	0.7085	0.6004	0.7040
14	海鲛NLP	0.7649	0.7156	0.5134	0.6851
15	OCHADAI (Pereira and Kobayashi, 2022)	0.7069	0.6445	0.5235	0.6573
16	Ryan Wang	0.3314	0.4058	0.3779	0.4044
N/A	MaChAmp (van der Goot, 2022)[7]	0.7204	0.6247	0.5532	0.6607
N/A	JARVix (Jakhotiya et al., 2022)[8]	0.8410	0.8162	0.7918	0.8243

Table 6: Results for Subtask A One Shot. The evaluation metric is macro F1 score, and the ranking is based on the 'All' column.

Ranking	Team	Subset		All
		Idiom Only	STS Only	
1	drsphelps (Phelps, 2022)	0.4030	0.8641	0.6402
2	colorful	0.4290	0.8880	0.6262
3	Mirs	0.3750	0.8623	0.6038
4	Zhichun Road (Cui et al., 2022)	0.2826	0.8359	0.5632
5	YNU-HPCC (Liu et al., 2022)	0.2872	0.7125	0.5577
6	ALTA	0.2154	0.8608	0.5379
	baseline (Tayyar Madabushi et al., 2021)	0.2263	0.8311	0.4810

Table 7: Results for Subtask B Pre Train. The evaluation metric is Spearman correlation, and the ranking is based on the 'All' column.

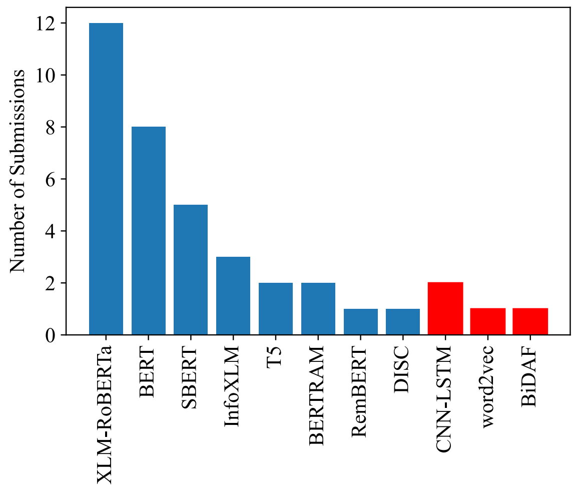


Figure 1: Models mentioned in the submissions. In blue are models that use transformers either wholly or partially, whilst in red are alternative models.

alternative loss functions.

6 Methods

The primary goal of this shared task was to provide a platform for the evaluation of a variety of methods for the identification and represention of MWEs. This section gives an overview of the methods that have been successful in each of the Subtasks. In particular, we attempt to identify the combination of methods across submissions that have significant potential for future development.

6.1 Subtask A

Subtask A, the identification of MWEs, comprised two settings: Zero Shot and One Shot. Crucially, the results from the task show that *methods that are successful in the Zero Shot setting, fail to be*

[7]Not ranked due to using a multi-task learning approach.

[8]Not ranked due to only submitting to the 'post-evaluation' phase.

Ranking	Team	Subset		All
		Idiom Only	STS Only	
1	YNU-HPCC (Liu et al., 2022)	0.4277	0.6637	0.6648
2	drsphelps (Phelps, 2022)	0.4124	0.8188	0.6504
3	Eat Fish	0.3688	0.8660	0.6475
4	Zhichun Road (Cui et al., 2022)	0.3956	0.5615	0.6401
	baseline (Tayyar Madabushi et al., 2021)	0.3990	0.5961	0.5951
5	ALTA	0.2566	0.6156	0.5755

Table 8: Results for Subtask B Fine Tune. The evaluation metric is Spearman correlation, and the ranking is based on the 'All' column.

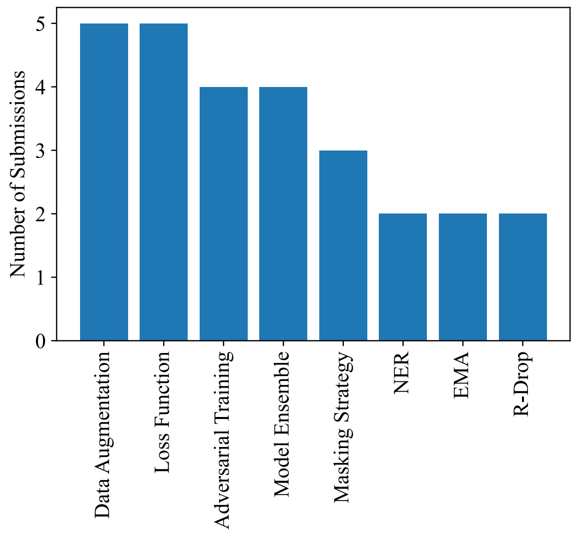


Figure 2: Methods mentioned in more than one submission.

successful in the One Shot setting and vice versa. The two problems seem to require capabilities that are quite distinct. This seems intuitive when translated into the kind of thinking that one might use in identifying idioms: When one hears an idiom for the first time, we are likely to recognise that it sounds 'idiom-like' based on our prior understanding of idioms, whereas when we come across an idiom that we are familiar with, we link our existing knowledge of that idiom with the current instance of it.

This seems to play out in the successful models in this Subtask, as the general trend amongst the methods that were successful in the Zero Shot setting, with one exception, is the generalisation of models using regularisation, data augmentation or dropout. While regularisation did feature in the top performing model in the One Shot setting, it seems to have been less important to generalise models when they had access to as little as one training example associated with each model. The best performing linguistically motivated method – which compares the semantic similarity between the MWE span and that of the surrounding context – ranked third in the Zero Shot setting, although it performed 11 points below the best performing method. This is of particular interest as this method has previously been shown to be extremely powerful in detecting idiomaticity in non-contextual models.

Models successful in the One Shot setting, again with one exception, seem to be those which are more powerful at extracting cues from the minimal training examples and tended to be larger, ensembled or trained to a larger extent using adversarial training. The best performing method which incorporated elements based on linguistic theory also ranked third in this setting and incorporated the gloss of each individual word in the target MWE to aid in models' ability to detect compositionality.

Interestingly, the use of the idiom principle in creating single token representations for MWEs is absent amongst the methods used for this Subtask. While such a comparison would have been interesting, it is hardly surprising that this method is not amongst those used, given that the cost of pre-training with new MWE tokens is rather high.

6.2 Subtask B

Subtask B, the novel task of creating contextual representations of MWEs which are consistent with the paraphrased version of that MWE as measured by Spearman's rank correlation, coefficient also had two settings: the first without associated training examples (Pre Train) and the second with (Fine Tune). Since the sentence embeddings generated by pre-trained language models cannot be directly compared for similarity, such models must be altered so as to be used for this Subtask. Additionally, as pointed out by Tayyar Madabushi et al. (2021), the MWEs contained within sentences can be represented using single tokens even without pre-training, as the 'breaking' of compositionality itself produces more accurate representations of sentences containing MWEs.

As such, models that perform the best on the Pre Train setting focus on the creation of more accurate single token representations of MWEs, while the top performing models on the Fine Tune setting, in general, focus on optimising sentence similarity. This seems to be consistent with the observation by Tayyar Madabushi et al. (2021) that fine-tuning is indeed a reasonable way of learning the representation of MWEs. It should be noted that these trends are less certain since there are fewer participants on this Subtask, some of whom do not share their methods, and the one team that we know used a method of learning new representations of MWEs is ranked first in the Pre Train setting but ranked second in the Fine Tune setting.

7 Conclusions and Future work

We present, in this paper, 'SemEval 2022 Task2: Multilingual Idiomaticity Detection and Sentence Embedding', consisting of two Subtasks: i) Subtask A, to test a language model's ability to detect idiom usage, and ii) Subtask B, to test a model's ability to generate representations of sentences containing idioms. This task, aimed at boosting research into the detection and representation of idiomatic expressions, had submissions from 25 teams consisting of close to 100 participants.

We additionally provide an overview and analysis of the methods used by participants, which we believe will help future research in this field. In particular, we highlight the need for distinct methods when detecting MWEs that have been previously seen and when detecting ones that have not. In representing idiomatic expressions, we show, through the novel idiomatic STS task presented here, that models are rather effective when they have training data available, but, as demonstrated in the Pre Train setting, more methods of encoding MWEs are required when training data is not available.

While the top performing methods across this task have been driven by deep neural models independent of linguistic features, we highlight that this does not imply that the addition of linguistically motivated features does not lead to improvements on the task. Instead, it points to the possibility of integrating these methods into the more powerful neural models in future work where an ablation study might shed more light on the impact of each feature.

Acknowledgements

This work was partially supported by the UK EPSRC grant EP/T02450X/1, by the CDT in Speech and Language Technologies and their Applications (UKRI grant number EP/S023062/1), by a *Ramón y Cajal* grant (RYC2019-028473-I), and by the grant ED431F 2021/01 (Galician Government).

References

Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing*, pages 267–292. CRC Press.

Douglas Biber, Stig Johansson, Geoffrey Leech, Susan Conrad, and Edward Finegan. 1999. *Longman Grammar of Spoken and Written English*. Pearson Education Ltd.

Joanne Boisson, Jose Camacho-Collados, and Luis Espinosa-Anke. 2022. Cardiffnlp-metaphor at semeval-2022 task 2: Targeted fine-tuning of transformer-based language models for idiomaticity detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1352–1362.

Cristina Butnariu, Su Nam Kim, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2010. SemEval-2 task 9: The interpretation of noun compounds using paraphrasing verbs and prepositions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 39–44, Uppsala, Sweden. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Zheng Chu, Ziqing Yang, Yiming Cui, Zhigang Chen, and Ming Liu. 2022. Hit at semeval-2022 task 2: Pre-trained language model for idioms detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2020. Rethinking embedding coupling in pre-trained language models. *arXiv preprint arXiv:2010.12821*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Paul Cook and Suzanne Stevenson. 2006. Classifying particle semantics in English verb-particle constructions. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, pages 45–53, Sydney, Australia. Association for Computational Linguistics.

Silvio Cordeiro, Aline Villavicencio, Marco Idiart, and Carlos Ramisch. 2019. Unsupervised compositionality prediction of nominal compounds. *Computational Linguistics*, 45(1):1–57.

Xuange Cui, Wei Xiong, and Songlin Wang. 2022. Zhichunroad at semeval-2022 task 2: Adversarial training and contrastive learning for multiword representations. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Britt Erman and Beatrice Warren. 2000. The idiom principle and the open choice principle. *Text-Interdisciplinary Journal for the Study of Discourse*, 20(1):29–62.

Samin Fakharian and Paul Cook. 2021. Contextualized embeddings encode monolingual and cross-lingual knowledge of idiomaticity. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 23–32, Online. Association for Computational Linguistics.

Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. 2009. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021a. Assessing the representations of idiomaticity in vector models with a noun compound dataset labeled at type and token levels. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2730–2741, Online. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021b. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Bradley Hauer, Seeratpal Jaura, Talgat Omarov, and Grzegorz Kondrak. 2022. Ualberta at semeval 2022 task 2: Leveraging glosses and translations for multilingual idiomaticity detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Iris Hendrickx, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2013. SemEval-2013 task 4: Free paraphrases of noun compounds. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 138–143, Atlanta, Georgia, USA. Association for Computational Linguistics.

Sami Itkonen, Jörg Tiedemann, and Mathias Creutz. 2022. Helsinki-nlp at semeval-2022 task 2: A feature-based approach to multilingual idiomaticity detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ray Jackendoff. 1997. Twistin' the Night Away. *Language*, 73:534–559.

Yash Jakhotiya, Vaibhav Kumar, Ashwin Pathak, and Raj Shah. 2022. Jarvix at semeval-2022 task 2: It takes one to know one? idiomaticity detection using zero and one shot learning. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Youngju Joung and Taeuk Kim. 2022. Hyu at semeval-2022 task 2: Effective idiomaticity detection with consideration at different levels of contextualization. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Milton King and Paul Cook. 2017. Supervised and unsupervised approaches to measuring usage similarity. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pages 47–52, Valencia, Spain. Association for Computational Linguistics.

Dekang Lin. 1999. Automatic identification of non-compositional phrases. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 317–324, College Park, Maryland, USA. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Changsheng Liu and Rebecca Hwa. 2018. Heuristically informed unsupervised idiom usage recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1723–1731.

Kuanghong Liu, Jin Wang, and Xuejie Zhang. 2022. Ynu-hpcc at semeval-2022 task 2: Representing multilingual idiomaticity based on contrastive learning. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Daming Lu. 2022. daminglu123 at semeval-2022 task 2: Using bert and lstm to do text classification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Diana McCarthy, Bill Keller, and John Carroll. 2003. Detecting a continuum of compositionality in phrasal verbs. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 73–80, Sapporo, Japan. Association for Computational Linguistics.

Navnita Nandakumar, Timothy Baldwin, and Bahar Salehi. 2019. How well do embedding models capture non-compositionality? a view from multiword expressions. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 27–34, Minneapolis, USA. Association for Computational Linguistics.

Roberto Navigli and Simone Paolo Ponzetto. 2010. Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 216–225.

Lis Pereira and Ichiro Kobayashi. 2022. Ochadai at semeval-2022 task 2: Adversarial training for multilingual idiomaticity detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Dylan Phelps. 2022. drsphelps at SemEval-2022 Task 2: Learning idiom representations using BERTRAM. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Carlos Ramisch, Silvio Ricardo Cordeiro, Agata Savary, Veronika Vincze, Verginica Barbu Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, Tunga Güngör, Abdelati Hawwari, Uxoa Iñurrieta, Jolanta Kovalevskaitė, Simon Krek, Timm Lichte, Chaya Liebeskind, Johanna Monti, Carla Parra Escartín, Behrang QasemiZadeh, Renata Ramisch, Nathan Schneider, Ivelina Stoyanova, Ashwini Vaidya, and Abigail Walsh. 2018. Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 222–240, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Carlos Ramisch, Agata Savary, Bruno Guillaume, Jakub Waszczuk, Marie Candito, Ashwini Vaidya, Verginica Barbu Mititelu, Archna Bhatia, Uxoa Iñurrieta, Voula Giouli, Tunga Güngör, Menghan Jiang, Timm Lichte, Chaya Liebeskind, Johanna Monti, Renata Ramisch, Sara Stymne, Abigail Walsh, and Hongzhi Xu. 2020. Edition 1.2 of the PARSEME shared task on semi-supervised identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 107–118, online. Association for Computational Linguistics.

Carlos Ramisch and Aline Villavicencio. 2018. Computational Treatment of Multiword Expressions. In Ruslan Mitkov, editor, *The Oxford Handbook of Computational Linguistics*, 2nd edition. Oxford University Press.

Livy Real, Erick Fonseca, and Hugo Goncalo Oliveira. 2020. The assin 2 shared task: a quick overview. In *International Conference on Computational Processing of the Portuguese Language*, pages 406–412. Springer.

Siva Reddy, Diana McCarthy, and Suresh Manandhar. 2011. An empirical study on compositionality in compound nouns. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 210–218, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Nils Reimers, Philip Beyer, and Iryna Gurevych. 2016. Task-oriented intrinsic evaluation of semantic textual similarity. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 87–96.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2002)*, pages 1–15, Mexico City, Mexico. Springer, Berlin, Heidelberg.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2015. A word embedding approach to predicting the compositionality of multiword expressions. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 977–983, Denver, Colorado. Association for Computational Linguistics.

Agata Savary, Carlos Ramisch, Silvio Cordeiro, Federico Sangati, Veronika Vincze, Behrang QasemiZadeh, Marie Candito, Fabienne Cap, Voula Giouli, Ivelina Stoyanova, and Antoine Doucet. 2017. The PARSEME shared task on automatic identification of verbal multiword expressions. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 31–47, Valencia, Spain. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020. BERTRAM: Improved word embeddings have big impact on contextualized model performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3996–4007, Online. Association for Computational Linguistics.

Nathan Schneider, Dirk Hovy, Anders Johannsen, and Marine Carpuat. 2016. SemEval-2016 task 10: Detecting minimal semantic units and their meanings (DiMSUM). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 546–559, San Diego, California. Association for Computational Linguistics.

Sabine Schulte im Walde, Stefan Müller, and Stefan Roller. 2013. Exploring vector space models to predict the compositionality of German noun-noun compounds. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 255–265, Atlanta, Georgia, USA. Association for Computational Linguistics.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Min Sik Oh. 2022. kpfriends at semeval-2022 task 2: Neamer - named entity augmented multi-word expression recognizer. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

John Sinclair and Les Sinclair. 1991. *Corpus, concordance, collocation*. Oxford University Press, USA.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. Astitchinlanguagemodels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477.

Simone Tedeschi and Roberto Navigli. 2022. Ner4id at semeval-2022 task 2: Named entity recognition for idiomaticity detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Rob van der Goot. 2022. Machamp at semeval-2022 tasks 2, 3, 4, 6, 10, 11, and 12: Multi-task multilingual learning for a pre-selected set of semantic datasets. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

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

Atsuki Yamaguchi, Gaku Morio, Hiroaki Ozaki, and Yasuhiro Sogawa. 2022. Hitachi at semeval-2022 task 2: On the effectiveness of span-based classification approaches for multilingual idiomaticity detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ziheng Zeng and Suma Bhat. 2021. Idiomatic expression identification using semantic compatibility. *Transactions of the Association for Computational Linguistics*, 9:1546–1562.

A Full Breakdown of Methods

All participants were invited to submit a short description of their methods, as well as to submit a paper. In Table 9, Table 10, Table 11, and Table 12 we give all the method descriptions that were submitted.

Ranking	Team	Method
1	clay	"domain pretraining with different masking strategies finetuning with data augmentation such as back-translation finetuning with soft label from former checkpoint"
2	yxb	"use mT5-Base use Easy Data Augmentation techniques include back-translation, synonym replacement, random insertion, random swap, random deletion include label unbalanced loss function：focal loss use model ensemble"
3	NER4ID	"Dual-encoder (Transformer-based) architecture that encodes both the potentially idiomatic expression and its context, and predicts idiomaticity by looking at their similarity score: high similarity -> compositional, low similarity -> idiomatic. Another core contribution of our method is the use of Named Entity Recognition as an intermediate step to pre-identify some non-idiomatic expressions; this provides great improvements."
4	HIT	"1. we use the big pre-trained model, XLM-R-large. Compared with multilingual-BERT and XLM-R-base, XLM-R-large is obviously improved. 2. Separate the exact same phrases as MWE in the target sentence with the sep token. If the phrase in the sentence is capitalized, It is more likely to be named entities that the model can distinguish, so the sep tokens are not added around the capitalized phrases. 3. Using Regularized Dropout(r-drop) as regularization."
5	Hitachi	"Our approach is built on top of multilingual pre-trained language models, which include InfoXLM and XLM-R. We solve the task of multilingual idiomaticity detection as a binary classification task and follow the standard fine-tuning method except not using a special [CLS] representation for classification. Instead, we first take an average over MWE's span representations and subsequently feed the averaged representation into a linear layer for classification."
6	OCHADAI	"our model relies on pre-trained contextual representations from different multilingual state-of-the-art transformer-based language models (i.e., multilingual BERT and XLM-RoBERTa), and on adversarial training, a training method for further enhancing model generalization and robustness."
7	yjs	"For each input sentence in the training set, if the MWE is idiomatic then its corresponding tokens are labeled as "idiomatic" and the remaining tokens are labeled as "literal"; if the MWE is literal then all the tokens in the sequence are labeled as "literal". Method 1: We apply a Bi-Directional Attention Flow (BiDAF) network (Seo et al., 2017), while we use mBERT as the contextualised embedding, and we use pos tag embedding as its query input."
8	CardiffNLP-m	"CardiffNLP-metaphors submitted the results of two methods in total, applied both for Task A Zero Shot and one-shot. The first method uses xlm-roberta-large and the second uses several monolingual bert language models for English, Portuguese and Galician. For the Zero Shot settings, bert-multilingual-base is used to label the Galician sentences, because no Galician examples were included in the training set. The embedding of the three sentences and the embeddings of the isolated target are input of the models. We optimized the models over different training parameters on the development set."
9	Mirs	-
10	Amobee	-
11	HYU	"We devise four features ((i), (ii), (iii), and (iv) in the following) as input for a simple yet effective idiomaticity classifier that is a multi-layer perceptron with one hidden layer. First, to consider the contextualized semantics of a target sentence when influenced by its surrounding context, we concatenate the target sentence with its (i) previous and (ii) next sentences respectively and inject the two chunks into our feature extractor (XLM-R; a bidirectional multilingual language model) independently to generate two distinct ((i) and (ii)) features. While constructing the aforementioned features, we also introduce two techniques to clarify the presence of a MWE in the sequence: the first highlights the location of the MWE with a new, dedicated positional encoding, and the second appends the MWE once again at the end of the sequence. In addition, we focus on the way of better utilizing the information existing solely in the target sentence, regarding a MWE and its context (i.e., phrases in the target sentence except for the MWE) as separate ones. Specifically, we derive (iii) the "context-only" representation of the target sentence by using a variant of the target sentence where the MWE is masked, while we compute (iv) the "MWE-only" representation, which corresponds to the intrinsic meaning of the MWE irrespective of context, by inserting only the MWE into the feature extractor."
12	Zhichun Road	"1. We use InfoXLM-Base as text encoder. (performance: infoxlm > XLM-R > Mbert) 2.We use exponential moving average (EMA) method. 3.We use adversarial attack strategy(performance: Smart > freeLb > PGD = FGM). Finally，our approach ranked 12th."
13	海鲛NLP	-
14	UAlberta	"Our unsupervised translation-based approach leverages translation information in multilingual resources such as OMW and BabelNet. The hypothesis is that the translations of idiomatic MWEs tend to be non-compositional, and therefore the individual words of an MWE are less likely to share mult-synsets with their translations. In addition, since MWEs that are named entities are usually literal, we use a part-of-speech tagger to identify proper nouns."
15	Helsinki-NLP	"The system utilizes linguistically motivated features that typically characterize idiomatic expressions: non-substitutability, non-compositionality and affectiveness. This feature model is based on pre-trained models and classification pipelines that have been integrated into the transformers library provided by HuggingFace. The final classification combines the feature model with either sentence-transformers or a base BERT model. The system also adds a back-translation feature and applies simple post-correction rules based on boolean features."
16	daminglu123	"We used the same model as baseline but added one more LSTM layer at last."
17	kpfriends	"We experimented with various inductive training methods only using Zero Shot data provided. We are still experimenting various schemes, including novel MWE ideas. We will share the findings in our paper."
18	Unimelb_AIP	"We tackled this task in an unsupervised way (i.e. without using any portion of the training data). First, we trained a standard CBOW word2vec model on unlabelled data and used it to predict the top-500 words that would fit into the surrounding context of the target MWE (as performed during the training of the CBOW model). Then, we calculated the maximum cosine similarities between the predicted words and each MWE component word, and regarded the MWE as "literal" ("non-idiomatic") if they are higher than the mean cosine similarity between the component words and their 500 closest words. Finally, we ensembled five CBOW models trained with different window sizes (5, 10, 15, 20, and 30) to incorporate different levels of contextual information. One limitation of this approach is that it often classifies proper-noun and idiomatic usages into the same class ("non-literal"; as their surrounding contexts differ a lot from the literal usage ones), and to mitigate this problem, we always regarded MWEs as "non-idiomatic" if they contained any capital letter."
19	YNU-HPCC	"As for methods of the best submission results, we added a linear layer so as to choose effective information from all of output layer that were extracted by pre-trained model, XLM-RoBERTa, and then fine-tuned it to classify."
20	Ryan Wang	"CNN-bidirectional LSTM classifier with jointly trained word embeddings trained on full passages (target and context) from Zero Shot data"
N/A	JARVix	"we fine-tune a pretrained XLNet on the task dataset (after evaluating multiple large language models and their majority-voting ensemble)."

Table 9: Methods used in Subtask A Zero Shot. Note: CardiffNLP-m is short for CardiffNLP-metaphors.

Ranking	Team	Method
1	HIT	"Mostly the same as the zero-shot. We train the One Shot model initialized from the best Zero Shot checkpoint. We additionally post-processed the predictions based on the distribution of the labels in the One Shot train file."
2	kpfriends	"More than 10 checkpoints were created per "English" and "Spanish / Galician" and inferred separately, later ensembled using soft-voting. To stabilize training of xlm-roberta-large, we started with pre-trained models provided by Huggingface which were xlm-roberta-large trained on CoNLL. We also had some good results with xlm-roberta-base. We will deep dive into methodology and interesting observations / error analysis in our paper."
3	UAlberta	"Our method uses a transformer-based sequence classifier that takes as an input the context sentence and the glosses of each individual word in the target multi-word expression. The intuition is that the addition of the glosses to the input might help the classifier to detect if the meaning of the target multi-word expression can be deduced from the definitions of the individual words, i.e., if it is compositional. Note that this method is applicable to both settings."
4	Zhichun Road	"1. We use InfoXLM-Base as text encoder. (performance: infoxlm > XLM-R > Mbert) 2.We use exponential moving average (EMA) method. 3.We use adversarial attack strategy(performance: freeLB > Smart > PGD = FGM). Finally，our approach ranked 4th."
5	clay	"same as Zero Shot setting, but with more data include Zero Shot data and One Shot data"
6	YNU-HPCC	"As for methods of the best submission results, we simply concated sentence and MWE and input into pre-trained model, XLM-RoBERTa. CLS from last layer was extracted to classify."
7	CardiffNLP-m	"CardiffNLP-metaphors submitted the results of two methods in total, applied both for Task A Zero Shot and one-shot. The first method uses xlm-roberta-large and the second uses several monolingual bert language models for English, Portuguese and Galician. For the Zero Shot settings, bert-multilingual-base is used to label the Galician sentences, because no Galician examples were included in the training set. The embedding of the three sentences and the embeddings of the isolated target are input of the models. We optimized the models over different training parameters on the development set. xlm-roberta-large significantly ouperforms the monolingual experimental settings on the one shot track. "
8	yxb	"use mT5-Base use Easy Data Augmentation techniques include back-translation, synonym replacement, random insertion, random swap, random deletion include label unbalanced loss function：focal loss use model ensemble"
9	NER4ID	"Same as zero-shot"
10	HYU	"In One Shot setting, we used the same method as in a Zero Shot setting."
11	yjs	"Method 2: We used the BiDAF-based DISC architecture by (Zeng and Bhat, 2021). DISC firstly combine GLOVE embeddings and POS embeddings with a BiDAF layer, which is then infused with mBERT by another BiDAF layer. We use both methods in the two settings, Method 1 performs better than Method 2. In the submissions, the different results is caused by different random seeds, with/without previous and next sentences, and with/without MWE."
12	Mirs	-
13	daminglu123	"We used the same model as baseline but added one more LSTM layer at last."
14	海蛟NLP	-
15	OCHADAI	"our model relies on pre-trained contextual representations from different multilingual state-of-the-art transformer-based language models (i.e., multilingual BERT and XLM-RoBERTa), and on adversarial training, a training method for further enhancing model generalization and robustness."
16	Ryan Wang	"CNN-bidirectional LSTM classifier with jointly trained word embeddings trained on full passages (target and context) from zero- and One Shot data"
N/A	MaChAmp	"Multi-task learning across SemEval tasks (2, 3, 4, 6, 10, 11, and 12). First we Pre Train a RemBERT multi-task model across all the tasks. Then we re-train a model for each task specifically. We used the default hyperparameters of MaChAmp v0.3 for all settings, which were finetuned on the GLUE benchmark and UD_English-EWT."
N/A	JARVix	"we use a relation network (Sung, et. al 2018) to find a similarity (or a dissimilarity) score between a query and it's same MWE support set, and assign a label accordingly. For this, we also evaluate a siamese network with a similar inference methodology."

Table 10: Methods for Subtask A One Shot. Note: CardiffNLP-m is short for CardiffNLP-metaphors.

Ranking	Team	Method
1	drsphelps	"Our model is a modification of the baseline system with the randomly initialised word embeddings for the one token MWEs replaced with embeddings created using Schick and Schutze's BERT for Attentive Mimicking (BERTRAM). BERTRAM models are trained for Portuguese and Galician alongside the provided English model, and examples use to create the MWE emebddings are taken from the common crawl corpora for English, Portuguese, and Galician. Further pretraining (up to 45 epochs) is done on the sentence transformers."
2	colorful	-
3	Mirs	-
4	Zhichun Road	"1.We add CrossAttention-Module at the top of the Sentence-Bert. (Including train and evaluate). 2.We add an extra Contrastive Loss. Finally, our approach ranked 4th."
5	YNU-HPCC	"As for methods of the best submission results, we extracted first-last-average vector and used an optimized method called CoSENT to train model. In comparison to SBERT, it could solve the problem of difference in process of training and prediction and get a better results."
6	ALTA	-

Table 11: Methods for Subtask B Pre Train.

Ranking	Team	Method
1	YNU-HPCC	"As for methods of the best submission results, both multiple-negatives-ranking-loss and triplet-loss function combined with pre-trained model, distiluse-base-multilingual-cased-v1, were used to fine-tune. "
2	drsphelps	"Using the models trained for the Pre Train setting, fine-tuning is performed using the provided training data, just as in the baseline system. The best overall performance is found after fine tuning for one epoch, however training for up to 50 epochs can drastically increase Spearman Rank scores for the idiom only data, while causing much less performance drop on the general STS data."
3	Eat Fish	"Multilingual model which was pretrained by using knowledge distillation Data augmentation Extract multiword from exist multiword package Two state training trick"
4	Zhichun Road	"1.We add CrossAttention-Module at the top of the Sentence-Bert. (Including train and evaluate). 2.We add an extra Contrastive Loss. Finally, our approach ranked 4th."
5	ALTA	-

Table 12: Methods for Subtask B Fine Tune.

Helsinki-NLP at SemEval-2022 Task 2: A Feature-Based Approach to Multilingual Idiomaticity Detection

Sami Itkonen and **Jörg Tiedemann** and **Mathias Creutz**
Department of Digital Humanities
Faculty of Arts
University of Helsinki
Finland
{firstname.lastname}@helsinki.fi

Abstract

This paper describes the University of Helsinki submission to the SemEval 2022 task on multilingual idiomaticity detection. Our system utilizes several models made available by HuggingFace, along with the baseline BERT model for the task. We focus on feature engineering based on properties that characterize idiomatic expressions. The additional features lead to improvements over the baseline and the final submission achieves 15th place out of 20 submissions. The paper provides an error analysis of our model including visualisations of the contributions of individual features.

1 Introduction

We participated in the SemEval 2022 Task 2 (Tayyar Madabushi et al., 2022) Subtask A, zero-shot[1] setting: classification of a sentence containing a potentially idiomatic two-word multiword expression (MWE) as idiomatic or literal. The task provided four data sets[2] for English, Portuguese and Galician. Each MWE was represented by multiple example sentences, accompanied by the context (previous and next sentences). Each MWE could be always idiomatic, always literal or anything in between. Table 1 shows examples for both idiomatic (0) and literal (1) cases. Expanded examples (with context) are shown in Table 6 in the Appendix.

[1]The MWEs in the test data do not appear in the training data.

[2]Training, development, evaluation and test sets, with Galician only appearing in the final test set.

The motivation for our approach is testing linguistically motivated features that reflect important properties of idioms, such as non-compositionality, non-substitutability, non-literal-translatability and affectiveness (see chapter 2) and to see whether pre-trained models can be helpful for capturing these features. Our system uses a combination of models: BERT fine-tuning (Tayyar Madabushi et al., 2021), sentence embeddings (Reimers and Gurevych, 2019) and a feature model based on the above idiomatic properties.

2 Background and Related Work

The detection and analysis of idiomaticity has a rich history in the literature. An important property of idioms is non-compositionality (that is, the meaning of the expression does not correspond to the combination of the meaning of its components). (Peng et al., 2014; Constant et al., 2017; Gantar et al., 2018) Related to it are non-substitutability (components cannot be substituted with their synonyms) (Farahmand and Henderson, 2016; Senaldi et al., 2016; Constant et al., 2017) and non-literal-translatability (Constant et al., 2017).

Idioms tend to be semantic outliers (Feldman and Peng, 2013; Peng et al., 2014; Salton et al., 2016) in the sense that they violate the lexical cohesion of the surrounding discourse. They are also known to be more affective (either positive or negative) (Peng et al., 2014) than literal expressions.

In addition to being relatively fixed lexically (non-substitutability), idioms often exhibit lack of

Label	Target
0	He was not a *blue blood* jurist issuing judicial decisions that nobody understood affecting people and corporations that nobody knew.
1	The *blue blood* of the fossil-like creature is the only natural source of limulus amoebocyte lysate, a clotting agent that is used to test batches of injectable drugs for bacterial contamination that could cause fever, organ damage and even death.

Table 1: Idiomatic (0) and literal (1) examples from the zero_shot setting of training set for English MWE *blue blood*, which can be interpreted as either idiomatic or literal.

syntactic and/or morphological variability (Peng et al., 2014; Constant et al., 2017). In general, quantifying any variability has traditionally required obtaining frequencies of the variants from a full corpus, as done by Inurrieta et al. (2020). However, as we only have a small number of examples for each idiom, these properties are not modeled in our approach.

Compositionality and substitutability are often tested with techniques like backtranslation and mask filling tasks. **Backtranslation** involves translating text to another (i.e. pivot) language and translating it back (Sennrich et al., 2016; Edunov et al., 2018), and it has often been used for paraphrasing and data augmentation. Backtranslations have also been used for idioms in related work, see, e.g., Moirón and Tiedemann (2006); Bahar Salehi and Baldwin (2018).

Mask filling (Zhu et al., 2019; Donahue et al., 2020) is closely related to the cloze task (Taylor, 1953), where the objective is to predict a word missing from an expression. Mask filling has lately been made easier as modern languages models such as BERT (Devlin et al., 2019) and its derivatives are themselves so-called Masked Language Models (MLM). Mask filling can be useful for testing substitutability in context (Karidi et al., 2021; Zhu and Bhat, 2021).

3 System Description

Our submission[3] considers three models: the baseline BERT model provided by the task authors (Tayyar Madabushi et al., 2021), sentence embeddings with sentence-transformers (Reimers and Gurevych, 2019) and a feature model based on idiomaticity features. All our components rely on existing models and tools that have been integrated into the transformers library provided by HuggingFace (Wolf et al., 2020).

The final classification combines information from two components (either fine-tuned BERT + feature model or sentence embeddings + feature model). The result will be taken from the model which has the higher label probability[4]. See Figure 1 for an overview.

We compare different variants of the system with the performance of individual features and various baselines.

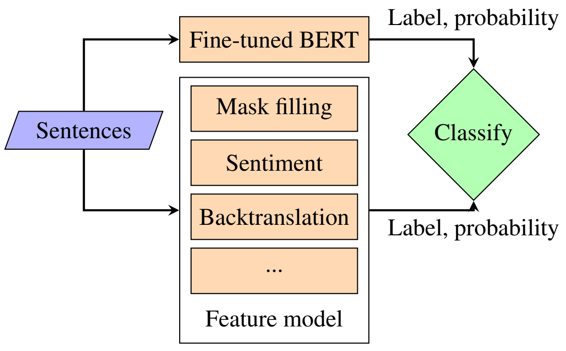


Figure 1: Basic classification procedure for the fine-tuned BERT + feature model combination. The feature model combines information from a number of HuggingFace models. Each model independently produces a label and associated probability. The label is by default taken from the model that has a higher probability. See Chapter 3.4 for the detailed classification procedure.

3.1 Fine-tuned BERT

The baseline model provided by the task organisers (Tayyar Madabushi et al., 2021) is based on BERT (Devlin et al., 2019). We build three variants: a) multilingual model (*bert-base-multilingual-cased*) for all languages (equivalent to the provided baseline), b) English model (*bert-base-cased*) for English data and multilingual model for non-English (trained with all data, including English) and c) same as case b, but multilingual model trained only with non-English data. The BERT model was fine-tuned with the training data, with the development set used for validation.

3.2 Sentence Embeddings (sbert)

Sentence embeddings can be used as an alternative baseline. We apply the *distiluse-base-multilingual-cased-v1*[5] model provided by HuggingFace and use the *sentence-transformers* python module[6]. The training procedure adopts the approach used by Tayyar Madabushi et al. (2021) by appending the MWE to the target sentence before training, as they found it to improve performance[7]. Logistic regression is used to train a classifier on top of the sentence embedding that we obtain from sbert.

[3]Implementation and details are available at https://github.com/dustedmtl/semeval2022.

[4]While both logistic regression and BERT models produce probabilities, the values aren't necessarily consummerate as BERT seems a lot more confident about the results.

[5]https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v1

[6]https://www.sbert.net

[7]This is not, however, equivalent to their methodology where the MWE is treated as a single token according to the "idiomatic principle" (i.e. stored as a single token in the mental lexicon) (Hashempour and Villavicencio, 2020).

Target	Label	Top terms	Top score	Hassub	Short	Found Idx	Found Score
There are several theories behind the origin of the term "*Double Dutch*."	0	., -, ..., s, *man*	0.008	False	3	10	-1.000
Além de ter sido um fracasso de bilheteria e crítica, o filme acabou marcado pelos seus *efeitos especiais*, principalmente ao antropomorfizar os gatos, que, bem, ficam um pouco bisonhos.	0	*erros*, personagens, **efeitos**, problemas, animais	0.540	True	0	3	0.092

Table 2: Training set substitution examples. In the first row, most of the suggestions are too short and no valid lexical substitute is found. The second example finds a component of *efeito especial* in plural form. The *Top terms* column shows the entry corresponding to *Top score* in *italics* and the one for *FoundIdx/Score* (if found) in **bold**. The above-zero scores represent the output from the mask-filling pipeline.

Note that we do not use the context sentences in this approach in any way.

3.3 Idiomaticity Features

Idiomaticity features are extracted using a number of HuggingFace pipelines and pre-trained models (see details below) for lexical substitution, sentiment analysis and backtranslation. Additionally, semantic outliers and surface-form-based boolean features are calculated. The training and classification with the feature model is done with logistic regression again, with boolean values converted to integers (True = 1 / literal, False = 0 / idiomatic).

3.3.1 Lexical Substitution

Because of the limited lexical variability and non-compositional nature of idiomatic expressions, it should be more difficult to find lexical substitutes for them, or their parts, than for literal expressions.

Our lexical substitution model utilizes the huggingface *fill-mask*[8] pipeline with the *xlm-roberta-base*[9] model. The pipeline will output a ranked list of top substitutions along with their scores[10]. Three different masks are used: one for masking the whole MWE (e.g. the expression *panda car* is replaced with <*mask*>)[11]), another for masking the first term (<*mask*> *car*) and a third one for masking the second term (*panda* <*mask*>)[12].

We obtain the top five candidates (individual words) from the pipeline. We are interested in two things: 1) how difficult it is to get a substitute in general, and 2) how difficult it is to get the *correct* substitute. The former reflects non-substitutability and the latter non-compositionality. A valid general substitute must only contain word characters and be at least three characters long. No other checks are made (such as whether the word class is correct or that the candidate is a synonym). A valid lexical substitute will additionally need to (case-insensitively) match either component of the MWE as it appears in the Target sentence. Inflected forms of the components are found by progressively stemming the component(s) with a regular expression- Additional tweaks are required for Portuguese because of orthographic variation (see Table 7 in the Appendix for examples).

The features that are generated are described below. Substitutions from masking individual terms are only used for the *Top score 1/2* and *FS/SS* features; all other features are derived from replacing the whole expression. Table 2 shows two examples for the features, with more examples in Table 8 in the Appendix.

Hassub Boolean feature: True when a valid lexical substitute is found, False otherwise.

Top score, Top score 1, Top score 2 The score of the top candidate, from replacing the whole expression, first term and second term, respectively. These features are a proxy for general (non-)substitutability.

Short, FS, SS The number of candidates that are too short (less than three characters) from masking the whole expression, first term and second term. The reasoning is that a lack of good suggestions is another proxy for non-substitutability.

FoundScore The score of the first valid lexical substitute [0-1], -1 otherwise. This one re-

[8]https://huggingface.co/tasks/fill-mask

[9]https://huggingface.co/xlm-roberta-base

[10]The mask-filling pipeline documentation doesn't explicitly state what the scores represent, but it's likely to be probability.

[11]The mask token is taken from the underlying model, which in this case is <*mask*>.

[12]The pipeline (by default) does not support using multiple mask tokens, so replacing the MWE with <*mask*> <*mask*> is not possible.

flects non-compositionality: replacement of the MWE with one of its components.

FoundIdx The index [1-5] of the first valid lexical substitute, 10 otherwise.

Note that the *FoundScore* and *FoundIdx* features essentially mix categorical and numeric values, which may reduce their usefulness. Additionally, *Top terms*, *Top terms 1* and *Top terms 2* are recorded from the mask filling process.

3.3.2 Sentiment Analysis

The affection feature is based on sentiment classification using the *cardiffnlp/twitter-xlm-roberta-base-sentiment*[13] model for predicting positive, negative or neutral sentiment. The neutral probability [0-1] is used as the value for the feature **Sentiment**.

3.3.3 Backtranslation

The target sentence is translated to another language (Portuguese for English, English for Portuguese and Galician) and then back-translated to the original language with the OPUS-MT (Tiedemann and Thottingal, 2020) models *opus-mt-en-roa* and *opus-mt-roa-en*[14]. The rationale is that idiomatic expressions exhibit non-compositionality and as such are less likely to be backtranslated correctly. The logic for locating the expression is the same that was used for lexical substitution: the exact form is required, with allowances for variations in Portuguese orthography. The value for the feature **Trans** is True if it is found and False otherwise. Table 7 in the Appendix shows a number of backtranslated examples.

3.3.4 Semantic Outliers

To measure semantic coherence, sentence embeddings are retrieved from sentence-transformers for the sentences/expressions. The value of the feature is the cosine similarity between the two.

Prevdiff Cosine similarity between the Previous and Target sentences.

Nextdiff Cosine similarity between the Next and Target sentences.

MWEdiff Cosine similarity between the MWE and the Target sentence.

3.3.5 Surface-form features

Based on data exploration, two additional surface features are used:

Quotes True if the MWE is enclosed in quotes, in which case it is more likely to be idiomatic.

Caps True if the MWE is capitalized (Camel Case). This is more likely to be a Proper Noun.

Table 9 in the Appendix shows examples for these features.

3.4 Final Classification

With the exception of simple baselines and majority voting classifier, the final classification is done by combining two components. For each prediction, the result will be taken from the model which has the higher probability. A number of ablation tests were run for the feature model with the development set to select the best set of features. In the end, all features except *Top score 1*, *FS* and *MWEdiff* were retained (where *features* means the bolded items in Chapters 3.3.1 through 3.3.5).

We also added a final post-correction step based on the results observed during development: the boolean features may (potentially) override the label. There are two modes for this: the first one will force the label unconditionally, the second one will force it only if the models disagree (*agree*).

The potential idiomatic features are *Quotes* and *!Trans* (*Trans* == False, that is, a mistranslation). Potential literal features are *Hassub* and *Caps*. Literal features take precedence, so if an expression is both quoted and capitalized, it is considered literal.

4 Results

4.1 Experimental Setup

Four data sets were released by the task administrators: training and developments sets, for which gold labels were provided; an evaluation set without gold labels (for which classification results could be obtained from the competition website) and a blind test set. The training set had more idiomatic (56%) and the development set more literal (54%) sentences.

The label is overwhelmingly likely to be 1 (literal) when the surface feature *Caps* == True (see

[13] `https://huggingface.co/cardiffnlp/twitter-xlm-roberta-base-sentiment`

[14] `https://huggingface.co/Helsinki-NLP/opus-mt-en-roa` and `https://huggingface.co/Helsinki-NLP/opus-mt-roa-en`. The models were chosen out of convenience, as only two models are required for translating between the languages to either direction.

Configuration	F1	EN	PT
Hassub	0.551	0.535	0.547
Trans	*0.597*	0.549	*0.615*
Sentiment	0.542	0.571	0.486
Majority class	0.545	*0.609*	0.564
Sentence transformers	0.614	0.635	0.536
+ features	*0.713*	*0.735*	*0.629*
BERT baseline	0.694	0.705	0.612
+ ml1: PT from full model	0.721	0.760	0.612
+ ml2: PT from separate model	0.725	0.760	0.590
+ features	0.715	0.716	0.656
+ ml1 + features	0.733	0.751	0.666
+ ml1 + features, agree	0.731	0.754	0.656
+ ml1 + features + trans	**0.751**	0.762	**0.694**
+ ml1 + features + trans, agree	0.750	**0.767**	0.683
+ ml2 + features + trans, agree	0.742	**0.767**	0.642
Majority voting + trans	*0.724*	0.743	*0.645*
Majority voting + trans, agree	0.723	*0.746*	0.633

Table 3: Results for the development set. Sections in order: baselines; combinations with sentence embeddings; BERT fine tuning models; majority voting classifiers. For BERT models, ml1 uses English model for English and and multilingual model for Portuguese (trained on all data), while ml2 is only trained with Portuguese data. Best/second best results are **bolded**/underlined, while the best result for each section is in *italics*.

Configuration	F1	EN	PT
Sentence transformers	0.558	0.579	0.500
+ features	0.646	0.655	**0.615**
BERT baseline	0.702	0.760	0.566
+ ml1	0.723	0.791	0.578
+ features	0.714	0.779	0.577
+ features + trans	0.671	0.695	0.591
+ features, agree	0.723	0.791	0.578
+ ml1 + features	0.720	0.794	0.577
+ ml1 + features, agree	**0.725**	**0.800**	0.578

Table 4: Results for the evaluation set. The feature model provides less improvement over the baseline. For Portuguese, the sbert+feature model combination outperforms all BERT-based variants. Best/second best results are **bolded**/underlined.

Language	F1
English	0.752
Portuguese	0.694
Galician	0.499
Total	0.663

Table 5: Official results for the test set.

Figure 3 in the Appendix). We also found the features *Hassub* and *Quotes* to be useful, so they are used in all cases involving the feature model.

4.2 Development and Evaluation Sets

Results for the development set are shown in Table 3 for various baselines, sentence-transformers-based models and BERT-based models. Baselines for the boolean *Hassub* and *Trans* are taken directly from the feature: True=1, False=0, while for *Sentiment* above-mean scores are considered literal. *Majority class* assigns the majority label (literal) for all sentences.

The sentence embeddings + feature model yields better results than the base BERT model, but in general fine-tuning BERT is much better than using sentence embeddings as a fixed feature. For the BERT-based models[15], using an English-only model for English improves results, as does using the *!Trans* boolean feature. Using the boolean features only when the models disagree (*agree*) does not seem to have much impact. As Figure 4 in the Appendix shows, the BERT-based models are more likely to label a sentence as literal. Finally, the majority voting classifier (using the majority label from all three classifiers) fares worse than BERT+feature models.

The results for the evaluation set (in Table 4 are largely similar to those for the development set, except for two things: 1) the *!Trans* feature is detrimental to English and somewhat helpful for Portuguese and 2) boolean features should be used only when the underlying models disagree. In the end, using the feature model with BERT only slightly improves the result (0.725 > 0.723). Additionally, sentence embeddings + feature model approach outperforms BERT-based models for Portuguese.

4.3 Test Set

The test predictions were generated with the setup that produced the best overall results for the evaluation set: different BERT models for English and non-English combined with the feature model with boolean features *Hassub*, *Quotes* and *Caps* (only when the models disagree about the label). The official test results in Table 5 show that the results for Galician are not great - roughly on the level of random chance[16]. The official baseline isn't much better[17], likely due Galician being a low-resource language and lacking training data for the pre-trained models that were used. For English

[15]The baseline, multilingual 1 and 2 (ml1 and ml2) configurations refer to variants a-c in section 3.1.

[16]Without knowing the true labels, we assume a 50/50 split.

[17]`https://sites.google.com/view/semeval2022task2-idiomaticity/baselines`

and Portuguese, the results are similar to the best results for the development set.

Regarding specific features, the results lend support to the idea that idioms are more affective, thus sentiment analysis can be useful for detecting idiomaticity (see Figure 2 and Figure 6 in the Appendix). The exception here seems to be Galician, which is probably because the sentiment model is based on tweets. However, it is easier to get a lexical replacement for Galician (see Figure 5a in the Appendix). It may be possible that the Galician test sentences use simpler language - relatively speaking.

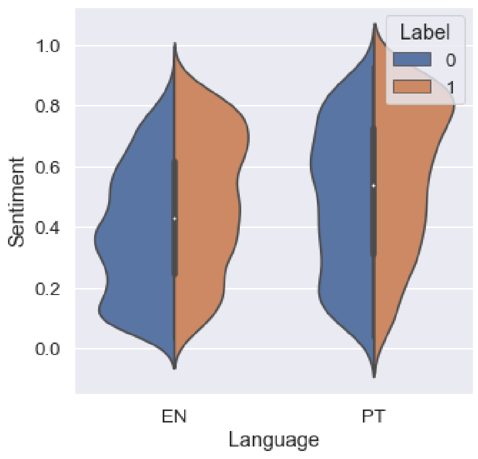


Figure 2: Violin plot for sentiment per language for the training set. The skewed sentiment distribution shows that the label is more likely to be literal (on average) for both English and Portuguese, if the sentiment score is higher (neutral sentiment). However, this feature alone is not sufficient for good performance.

Using the boolean features on top of the classifier models can be a bit of a hit-and-miss: what works with one dataset may be detrimental with another. Specifically, the *!Trans* feature worked well on the development set, but not on the evaluation set, and the *Hassub* feature worked on both of these sets, but not on the test set. In other words, the boolean features may make the model less robust.

Ablation studies performed after the official end of the competition confirm that using the *Hassub* feature for the test set was not a good strategy. Furthermore, a feature-only model (without sentence embeddings or BERT) outperformed the combined model, with the best results achieved by using the combined model for English and feature-only model for Portuguese and Galician. Nevertheless, even these results do not come close to the best models of the competition.

For detecting semantic outliers, the approach used in this paper (similarity based on sentence-transformers embeddings) appears to be too simple. More refined methods, such as those measuring lexical cohesion (Sporleder and Li, 2009) would be required.

5 Conclusions

Our system combines a feature model based on a number of idiomaticity features with a BERT transformer classifier. The feature model achieves competitive results compared to the reportedly strong baseline (Tayyar Madabushi et al., 2022), although it does not fare nearly as well as the best systems that competed in the subtask. Unsurprisingly, most of the features work best for English, whether or not the underlying BERT model is multilingual or not.

The work shows that a classification system utilizing idiomatic properties such as non-compositionality, non-substitutability and affectiveness can be implemented with readily available transformer APIs.

Another idea for future work is to improve the back-translation test by combining a "good" forward translation model (i.e. one that tends to properly treat idiomatic expressions) with a "bad" back-translation model (i.e. one that tends to produce literal translations). The latter could also be done by forcing component-wise translations in the back-translation step to reveal non-compositionality of the expression.

Acknowledgements

The work in this paper was supported by the project Behind the Words funded by the Academy of Finland. Computational resources were provided by CSC, the Finnish IT Center for Science. We also thank the anonymous reviewers for their substantive comments.

References

Paul Cook Bahar Salehi and Timothy Baldwin. 2018. Exploiting multilingual lexical resources to predict mwe compositionality. In *Multiword expressions at length and in depth: Extended papers from the MWE 2017 workshop*, page 343–373, Berlin, Heidelberg.

Mathieu Constant, Gülşen Eryiğit, Johanna Monti, Lonneke van der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. Survey: Multiword expression processing: A Survey. *Computational Linguistics*, 43(4):837–892.

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

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Meghdad Farahmand and James Henderson. 2016. Modeling the non-substitutability of multiword expressions with distributional semantics and a log-linear model. In *Proceedings of the 12th Workshop on Multiword Expressions*, pages 61–66, Berlin, Germany. Association for Computational Linguistics.

Anna Feldman and Jing Peng. 2013. Automatic detection of idiomatic clauses. In *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, CICLing'13, page 435–446, Berlin, Heidelberg. Springer-Verlag.

Polona Gantar, Lut Colman, Carla Parra Escartín, and Héctor Martínez Alonso. 2018. Multiword Expressions: Between Lexicography and NLP. *International Journal of Lexicography*, 32(2):138–162.

Reyhaneh Hashempour and Aline Villavicencio. 2020. Leveraging contextual embeddings and idiom principle for detecting idiomaticity in potentially idiomatic expressions. In *Proceedings of the Workshop on the Cognitive Aspects of the Lexicon*, pages 72–80, Online. Association for Computational Linguistics.

Uxoa Inurrieta, Itziar Aduriz, Arantza Díaz de Ilarraza, Gorka Labaka, and Kepa Sarasola. 2020. Learning about phraseology from corpora: A linguistically motivated approach for Multiword Expression identification. *PLoS ONE*, 15(8 August):1–18.

Taelin Karidi, Yichu Zhou, Nathan Schneider, Omri Abend, and Vivek Srikumar. 2021. Putting words in BERT's mouth: Navigating contextualized vector spaces with pseudowords. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10300–10313, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Begoña Villada Moirón and Jörg Tiedemann. 2006. Identifying idiomatic expressions using automatic word-alignment. In *Proceedings of the Workshop on Multi-word-expressions in a multilingual context*.

Jing Peng, Anna Feldman, and Ekaterina Vylomova. 2014. Classifying idiomatic and literal expressions using topic models and intensity of emotions. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2019–2027, Doha, Qatar. Association for Computational Linguistics.

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

Giancarlo D. Salton, Robert J. Ross, and John D. Kelleher. 2016. Idiom token classification using sentential distributed semantics. *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, 1:194–204.

Marco Silvio Giuseppe Senaldi, Gianluca E. Lebani, and Alessandro Lenci. 2016. Lexical variability and compositionality: Investigating idiomaticity with distributional semantic models. In *Proceedings of the 12th Workshop on Multiword Expressions*, pages 21–31, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Caroline Sporleder and Linlin Li. 2009. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 754–762, Athens, Greece. Association for Computational Linguistics.

Wilson L. Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal. European Association for Machine Translation.

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

Wanrong Zhu, Zhiting Hu, and Eric Xing. 2019. Text Infilling.

Wanzheng Zhu and Suma Bhat. 2021. Euphemistic Phrase Detection by Masked Language Model. *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 163–168.

A Appendix

A number of tables and figures are presented here.

Table 6 shows samples from the training data. Tables 7, 8 and 9 list feature examples for substitution, backtranslation and Quotes/Caps surface features.

Figure 3 plots the boolean features *Hassub*, *Quotes*, *Caps* and *Trans* against the literal and idiomatic labels for the training set. Figure 4 demonstrates the differences in the labeling done by fine-tuned BERT and the feature model.

Figure 5 shows counts for boolean features for each language and data set. Illustrations for sentiment scores for all languages and datasets are shown in Figure 6.

Label	**Previous**	**Target**	**Next**
0	Heading outside (even just for a couple of minutes) or doing mundane things like brushing your teeth and making the bed can help your mind accept the fact that yes, alas, you are awake now.	Whether you're a *night owl* or early bird, though, try to make sure you're not diving right onto your phone.	Your morning will start calmer if you don't dive right into work emails and scrolling.
1	LCG asks that Monday customers put garbage and recycling carts at the curb for collection Tuesday morning.	In addition, the Lafayette Transit System office will be closed Monday, and there will be no Daytime, *Night Owl* or Paratransit bus service Monday.	Bus and paratransit services will resume regular schedules Tuesday.
0	I practiced before him in court and stood beside him on Canal Street during Endymion.	He was not a *blue blood* jurist issuing judicial decisions that nobody understood affecting people and corporations that nobody knew.	His blood was red with a little Irish green thrown in.
1	The American horseshoe crab has outlived the dinosaurs and survived four mass extinction events, but its population has been devastated in recent years, partly due to harvesting for biomedical production.	The *blue blood* of the fossil-like creature is the only natural source of limulus amoebocyte lysate, a clotting agent that is used to test batches of injectable drugs for bacterial contamination that could cause fever, organ damage and even death.	The crabs are fished from the oceans, taken to a lab to have about 30% of their blood harvested, then released back into the wild.

Table 6: Idiomatic (0) and literal (1) examples from the training set for English MWEs *night owl* and *blue blood* in the zero_shot setting. For *night owl*, the second example is considered literal as the MWE refers to a company name (Proper Noun).

MWE	Target	Label	BT	Trans
double dutch	Since settlers from other areas of the world could not understand the songs, they labeled the activity “*Double Dutch*.”	1	As settlers from other parts of the world could not understand the songs, they labeled the activity "*Double Dutch*".	True
double dutch	At 6,400gns, Auldhouseburn sold another by the same sire, and again in lamb to *Double Dutch*, to Northern Irish buyer, J. Cubbitt of Ballymena.	1	At 6,400gns, Auldhouseburn sold another by the same sire, and again in lamb to Duplo Dutch, to the Northern Irish buyer, J. Cubbitt of Ballymena.	False
círculo virtuoso	Com a segurança da imunização em massa e os números traduzindo sua eficácia, fica mais fácil para o americano médio sentir-se confiante em marcar sua próxima viagem, gerando um *circulo virtuoso* para o setor nos próximos meses.	0	Com a segurança da imunização em massa e os números traduzindo sua eficácia, torna-se mais fácil para o americano médio sentir-se confiante em marcar sua próxima viagem, gerando um *círculo virtuoso* para o setor nos próximos meses.	True
círculo virtuoso	Apesar de dizer que o Brasil está no caminho de um "*círculo virtuoso* na economia", o executivo do banco enxerga riscos internos e externos no horizonte da renda variável e, por isso, evita projeções de curto prazo.	0	Apesar de dizer que o Brasil está no caminho de um "*círculo virtuoso* na economia", o executivo do banco vê riscos internos e externos no horizonte da renda variável e, portanto, evita projeções a curto prazo.	True
amor-próprio	No novo livro, sobre *amor-próprio* e também validação social, Paula Cordeiro relata como sobreviver à era digital.	1	No novo livro, sobre o *amor próprio* e também a validação social, Paula Cordeiro relata como sobreviver à era digital.	True

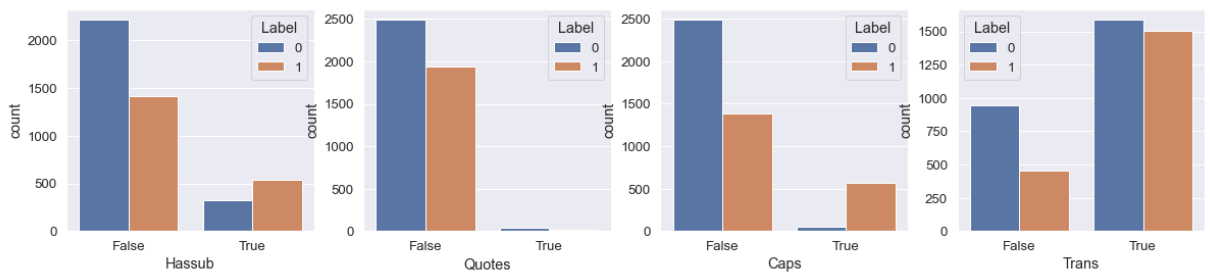
Table 7: Backtranslation examples for the training set; *Target* is the original sentence, *BT* is the backtranslated one. The matching process occasionally requires some tweaks for Portuguese. In the third row, the Target contains the expression *círculo virtuoso* without an accent, while the last row shows the translation of *amor-próprio* separated with a space instead of a dash.

Target	Label	Top terms	Top score	Hassub	Short	Found Idx	Found Score
There are several theories behind the origin of the term "*Double Dutch*."	0	., -, ..., s, *man*	0.008	False	3	10	-1.000
Double Dutch also derives from the same era, Dutch seeming a strange and convoluted language hence Double Dutch meaning indescernible, mad and generally all round not on foreign speak.	0	It, *English*, This, **Dutch**, German	0.385	True	1	4	0.062
No vídeo divulgado nas redes sociais, é possível perceber que um som faz o casal olhar para o prédio da frente e ver o *efeito especial* da fumaça.	0	*som*, **efeito**, tamanho, aumento, ar	0.214	True	1	2	0.120
Os *efeitos especiais* são necessários em cenas de batalha, porém, a DC costuma abusar da técnica.	0	***efeitos***, equipamentos, personagens, dados, filmes	0.158	True	0	1	0.158

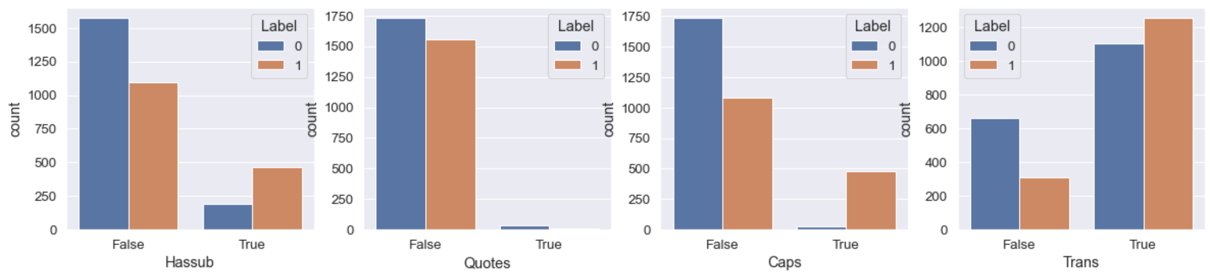
Table 8: Abbreviated substitution examples for the training set. The first two examples are for the English MWE *double dutch*, the last two for the Portuguese MWE *efeito especial*. In the first row, a substitution is not found and most of the suggested substitutions are too short, leading to a *Short* value of 3. In the second row, the fourth suggestion matches the MWE. For Portuguese, the expression *efeito especial* is found in singular form in the first example and in plural form in the second; the substitute suggestions must match the expression. The *Top terms* column shows the entry corresponding to *Top score* in *italics* and the one for *FoundIdx* and *FoundScore* (if found) in **bold**. The scores represent the output from the mask-filling pipeline.

MWE	Target	Label	Quotes	Caps
double dutch	*Double Dutch* also derives from the same era, Dutch seeming a strange and convoluted language hence Double Dutch meaning indescernible, mad and generally all round not on foreign speak.	0	False	True
double dutch	Since 1977 we have had a plethora of Foreign Ministers, to whom the subject of foreign affairs was *double Dutch*.	0	False	False
double dutch	At 6,400gns, Auldhouseburn sold another by the same sire, and again in lamb to *Double Dutch*, to Northern Irish buyer, J. Cubbitt of Ballymena.	1	False	True
night owl	The researchers said experience shows *"night owl"* patients with depression are less likely to recover and are more likely to commit suicide.	0	True	False

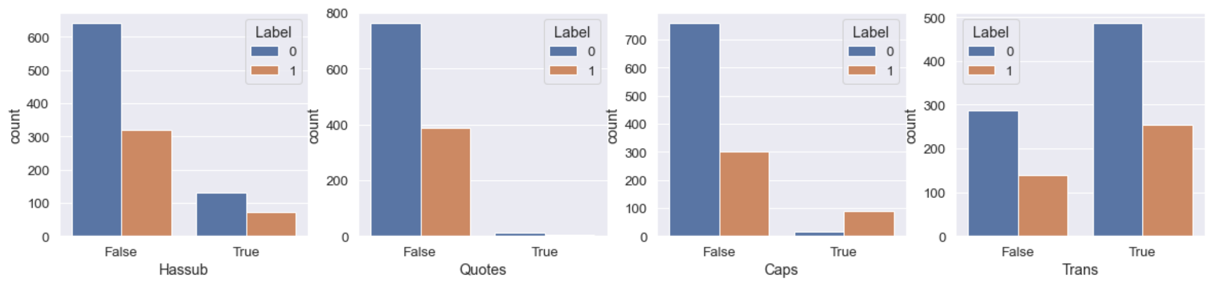
Table 9: Quotes/Caps examples for the training set. In the second row, *Caps* == False as both components of *double dutch* are not capitalized.



(a) All

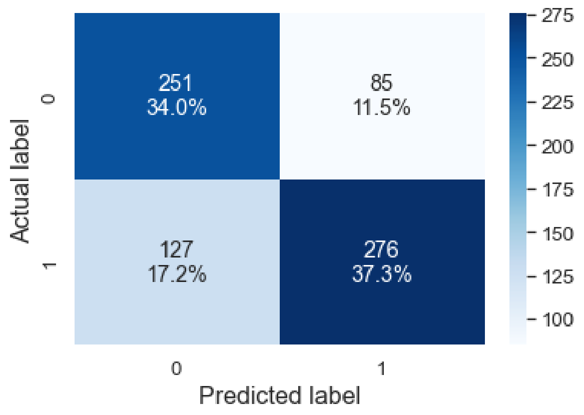


(b) English

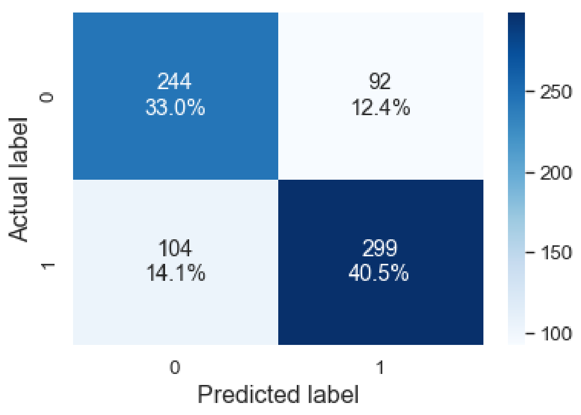


(c) Portuguese

Figure 3: Boolean features vs label for the training set. The label is overwhelmingly likely to be literal if the MWE is Capitalized (*Caps* == True), while idiomatic label is more likely if the MWE is mistranslated (*Trans* == False). It is generally difficult to get a valid lexical substitute (*Hassub* == True).

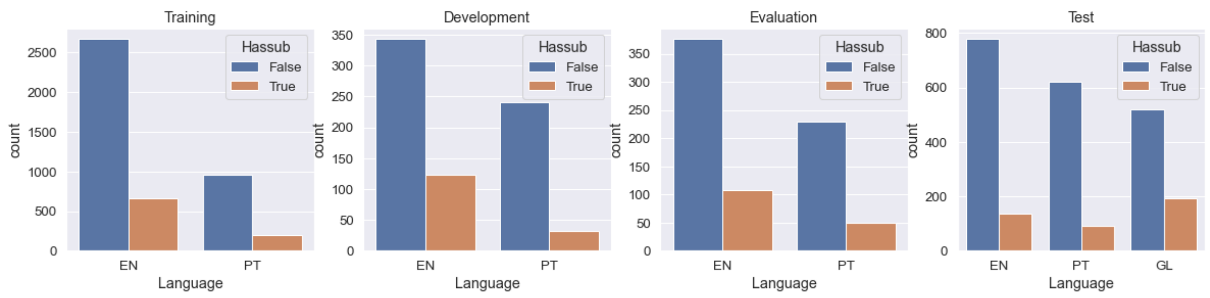


(a) Confusion matrix for the sbert+feature model.

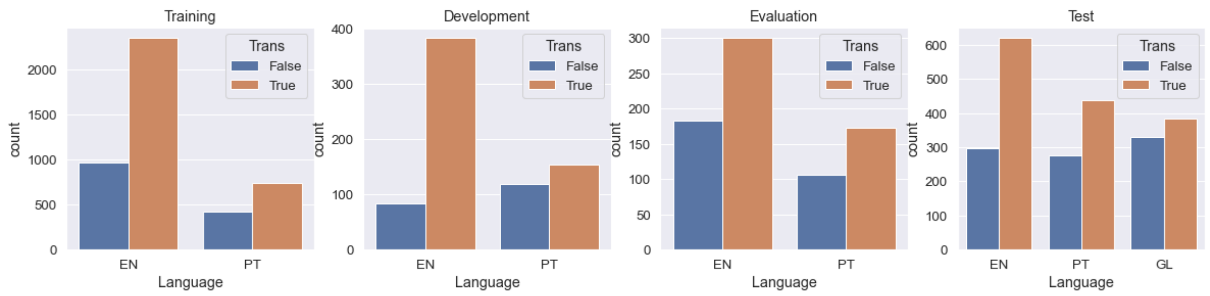


(b) Confusion matrix for the BERT+feature model.

Figure 4: Confusion matrices for the development set. The fine-tuned BERT model is more likely to classify the sentence as literal than the feature model.

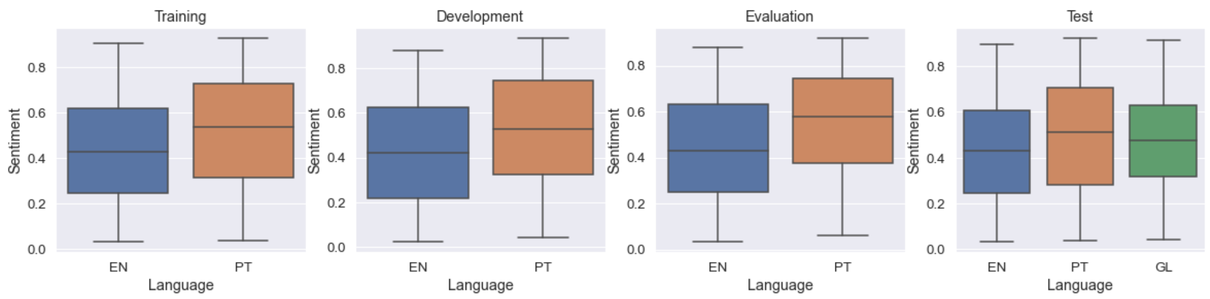


(a) Hassub

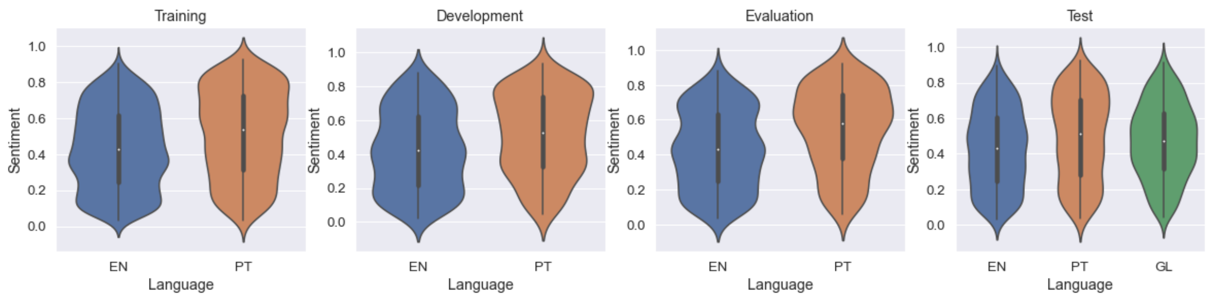


(b) Trans

Figure 5: Counts per feature, set and language. It is relatively easier to get a valid lexical substitute for Galician. Getting a correct backtranslation is harder for Portuguese than English, and harder still for Galician.



(a) Box plot



(b) Violin plot

Figure 6: Sentiment scores per set and language. The distributions are skewed for English and Portuguese, while the sentiment scores seem uninformative for Galician. Portuguese scores are generally higher - it is more difficult for the sentiment classifier to classify sentences as affective (either positive or negative).

Hitachi at SemEval-2022 Task 2: On the Effectiveness of Span-based Classification Approaches for Multilingual Idiomaticity Detection

Atsuki Yamaguchi, Gaku Morio, Hiroaki Ozaki and **Yasuhiro Sogawa**
Research and Development Group, Hitachi, Ltd.
Kokubunji, Tokyo, Japan
`{atsuki.yamaguchi.xn, gaku.morio.vn, hiroaki.ozaki.yu, yasuhiro.sogawa.tp}@hitachi.com`

Abstract

In this paper, we describe our system for SemEval-2022 Task 2: *Multilingual Idiomaticity Detection and Sentence Embedding*. The task aims at detecting idiomaticity in an input sequence (Subtask A) and modeling representation of sentences that contain potential idiomatic multiword expressions (MWEs) (Subtask B) in three languages. We focus on the zero-shot setting of Subtask A and propose two span-based idiomaticity classification methods: MWE span-based classification and *idiomatic* MWE span prediction-based classification. We use several cross-lingual pre-trained language models (InfoXLM, XLM-R, and others) as our backbone network. Our best-performing system, fine-tuned with the span-based idiomaticity classification, ranked fifth in the zero-shot setting of Subtask A and exhibited a macro F_1 score of 0.7466.

1 Introduction

SemEval-2022 Task 2 (Tayyar Madabushi et al., 2022) involves detecting idiomaticity in a given sentence (Subtask A) and learning effective representations of sentences that may contain idiomatic multiword expressions (MWEs) (Subtask B) in three languages: English, Portuguese, and Galician. Processing idiomaticity in a sequence correctly is an essential task in natural language processing (NLP), as idiomatic expressions are a key component of natural languages. Its high performance will contribute to various downstream tasks, such as sentiment analysis, information retrieval, and machine translation (Hashempour and Villavicencio, 2020; Tayyar Madabushi et al., 2021).

In this work, we propose two different approaches for multilingual idiomaticity detection (Subtask A) that take advantage of MWE span-based features. We use several cross-lingual pre-trained language models (InfoXLM (Chi et al., 2021a), XLM-R (Conneau et al., 2020), and others) and exploit their MWE span representations for classification, instead of using a special classification token (`[CLS]`), which typically corresponds to the first input token. Our concept is that these models should be able to focus more on the idiomaticity of an MWE in a given sequence by using its span representation rather than using the `[CLS]` token for classification, potentially resulting in a better detection performance.

Our main findings in the shared task are in three-fold.

1. The span-based idiomaticity classification method is highly effective compared to the standard `[CLS]`-based sequence classification approach adopted in various BERT (Devlin et al., 2019)-like models (Liu et al., 2019; Lan et al., 2020; Clark et al., 2020).

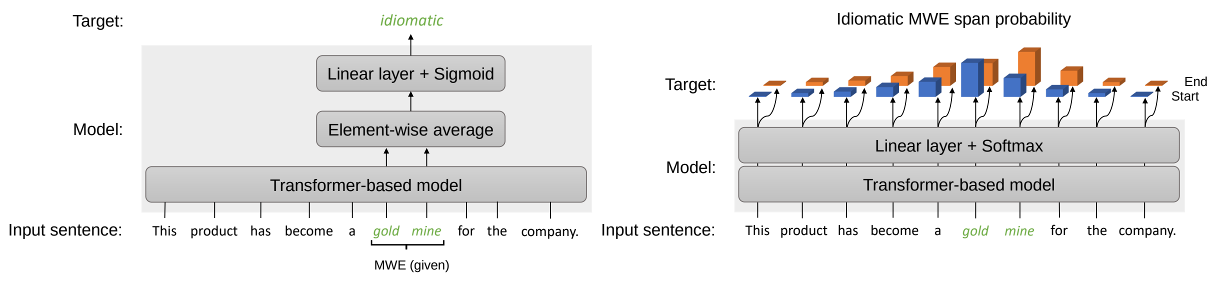
2. Detecting idiomaticity in Galician with no training data available is challenging even with state-of-the-art cross-lingual pre-trained language models.

3. Utilizing adjacent contexts with a target sentence is not always the best option for idiomaticity detection, even though it improves the baseline performance.

Consequently, our best-performing system, using the span-based classification, ranked fifth among 20 systems in the zero-shot setting of Subtask A and showed a macro F_1 score of 0.7466 on the test set.

2 Background

Idiomaticity Detection While the task of idiomaticity detection with respect to MWEs is not new, it is still considered challenging because state-of-the-art language representation models heavily depend on the principle of compositionality (Pelletier, 1994), which idioms do not follow, due to their tokenization methods (Kudo, 2018; Sennrich



(a) Span-based idiomaticity classification

(b) Span prediction-based idiomaticity classification

Figure 1: Overview of two proposed approaches for detecting idiomaticity.

et al., 2016). To overcome this problem, some studies (Hashempour and Villavicencio, 2020; Garcia et al., 2021) have regarded an MWE as a single token motivated by the assumption that people recognize an idiom as a single token (Sinclair et al., 1991). Alternatively, others have tried to utilize the adjacent contexts of MWEs as inputs and have demonstrated the effectiveness of this against tasks targeting verb-noun constructions (Sporleder and Li, 2009; Salton et al., 2016; King and Cook, 2018) and noun compounds (Tayyar Madabushi et al., 2021). This paper also utilizes adjacent contexts for classification but proposes new idiomaticity detection approaches in which the span information of an MWE plays an important role.

Cross-lingual Pre-trained Language Models Cross-lingual pre-trained language models have shown promising results in multilingual NLP tasks since the emergence of multilingual BERT (mBERT) (Devlin et al., 2019). In general, they are pre-trained with either multilingual masked language modeling (Devlin et al., 2019; Conneau and Lample, 2019; Conneau et al., 2020; Chung et al., 2021) or translation language modeling (Conneau and Lample, 2019). The difference between the two is that the former uses monolingual sentences while the latter utilizes concatenated parallel sentences for inputs. The state-of-the-art InfoXLM (Chi et al., 2021a) further utilized contrastive learning, where a model needs to distinguish a correct translated sample from negative ones. Our approach uses several cross-lingual pre-trained language models, including InfoXLM, XLM-R, XLM-Align (Chi et al., 2021b), and RemBERT (Chung et al., 2021), to utilize multilingual idiomaticity data efficiently.

3 Task Description

We briefly describe a multilingual idiomaticity detection task (Subtask A). Given a sentence composed of n words $\mathcal{S}_{\text{target}} = [w_1, \ldots, w_n]$ that contains an m-word MWE $\mathcal{W} = [w_1^{\text{MWE}}, \ldots, w_m^{\text{MWE}}]$, $\mathcal{S}$'s preceding sentence $\mathcal{S}_{\text{prev}}$, and succeeding sentence $\mathcal{S}_{\text{next}}$, the task is to classify if $\mathcal{W}$ is *idiomatic* (0) or not (1). The task dataset is based on Tayyar Madabushi et al. (2021) and contains *Galician* in addition to English and Portuguese. Each sample consists of $\mathcal{S} = [\mathcal{S}_{\text{prev}}; \mathcal{S}_{\text{target}}; \mathcal{S}_{\text{next}}]$, $\mathcal{W}$, a language type $lang \in \{\text{“EN”}, \text{“PT”}, \text{“GL”}\}$, and an idiomaticity label $y_{\text{MWE}} \in \{0, 1\}$. In the zero-shot setting, participants do not have any training samples for Galician and are only allowed to use the officially provided training and development sets for training. They must also use the same approach for all samples except language and can submit up to five systems for evaluation.

4 System Overview

Our system relies on cross-lingual pre-trained language models (InfoXLM and XLM-R and others) and classifies samples using either span-based classification or span prediction-based classification. We fine-tune several pre-trained language models and obtain final predictions by using an ensemble method. Figure 1 visualizes our approach for multilingual idiomaticity detection.[1]

4.1 Span-based Classification

There have been several attempts to utilize span hidden representations from a Transformer (Vaswani et al., 2017)-based pre-trained language model in various NLP tasks that can be formulated as span classification, including named entity recognition

[1]Appendix A describes our approaches in detail using equations.

(Yamada et al., 2020; Eberts and Ulges, 2020), relation classification (Baldini Soares et al., 2019) and propaganda technique classification (Da San Martino et al., 2020; Dimov et al., 2020; Jurkiewicz et al., 2020). These studies have all demonstrated the effectiveness of the span representations.

Here, we utilize this approach to solve the task of idiomaticity detection. We first pick up the span hidden representations of an MWE from the Transformer-based model, take their average, and feed the resulting vector into a linear layer for final classification (Figure 1 (a)).

Our concept with this approach is that the model should be able to focus more on the usage of an MWE in terms of idiomaticity in context rather than using a special `[CLS]` token for classification. Although we do not regard an MWE as a single token to encode, it is true that our approach is inspired by the idiom principle (Sinclair et al., 1991) in the sense that our model classifies a single averaged MWE span hidden representation for final classification.

4.2 Span Prediction-based Classification

For the second approach, we propose span prediction-based idiomaticity classification, inspired by BERT (Devlin et al., 2019)'s fine-tuning approach against the SQuAD v2.0 dataset (Rajpurkar et al., 2016, 2018), which contains some unanswerable questions. In BERT's approach, the answer text span in a given text for answerable questions is predicted, and the position of a `[CLS]` token for questions that do not have an answer is output. In our case, the task is to predict the MWE span in a given sequence for idiomatic MWEs and to output the position of a `[CLS]` token for non-idiomatic MWEs. This approach is illustrated in Figure 1 (b).

Our concept with this approach lies in the generalizability over unseen data. Predicting an idiomatic MWE span requires a model to differentiate non-idiomatic MWEs from idiomatic ones. This should force the model to learn semantic knowledge on MWEs in terms of idiomaticity and subsequently help the model to deliver a better performance on the test data.

5 Experimental Setup

Models We mainly utilized InfoXLM and XLM-R for our system submission, but we also tested several other cross-lingual pre-trained language models. Table 1 shows the list of models tested in this paper.[2] We selected these models because they are easy-to-use thanks to their availability on the HuggingFace Hub.[3]

Model	**Identifier**
InfoXLM (Chi et al., 2021a)	microsoft/infoxlm-large
XLM-R (Conneau et al., 2020)	xlm-roberta-large
XLM-Align (Chi et al., 2021b)	microsoft/xlm-align-base
RemBERT (Chung et al., 2021)	google/rembert
mBERT (Devlin et al., 2019)	bert-base-multilingual-cased

Table 1: List of cross-lingual pre-trained language models tested in this paper. Each identifier corresponds to the model name in the `transformers` library.

Data and Preprocessing We utilized the official training and development sets[4] for training, and no additional data was used, as stipulated by the competition rules. We tokenized samples using pre-trained tokenizers provided by the `transformers` library (Wolf et al., 2020) and set the sequence length to 256. When using an MWE as an additional input feature, we added it to the second sentence following Tayyar Madabushi et al. (2021).

Evaluation Metrics The evaluation metric for Subtask A is a macro F_1 score with respect to idiomaticity labels.

Fine-tuning We implemented our approaches using PyTorch (Paszke et al., 2019) and the `transformers` library. We fine-tuned all models for ten epochs each using one NVIDIA Tesla V100 (SXM2 - 32GB) with a batch size of 16 and automatic mixed precision applied. We used an Adam optimizer (Kingma and Ba, 2014) and saved a checkpoint of each model every ten steps. To minimize the effect of random seeds, we trained all models for ten times each with different random seeds. We then selected the best-performing models on the basis of the macro F_1 scores on the development set.[5]

Ensemble We fused the outputs of the fine-tuned pre-trained language models to further boost perfor-

[2]We provide a brief explanation of the five cross-lingual pre-trained language models in Appendix B and the performance comparison in Appendix E.

[3]`https://huggingface.co/models`

[4]`https://github.com/H-TayyarMadabushi/SemEval_2022_Task2-idiomaticity`

[5]For more details on hyperparameters, please refer to Appendix C.

System	Ensemble Method	Models Used
System 1	No Ensemble	InfoXLM for EN, XLM-R for PT & GL
System 2	Stacking	InfoXLM $\times$ 4, XLM-R $\times$ 1
System 3	Majority Vote	InfoXLM $\times$ 5, XLM-R $\times$ 1
System 4	Stacking	InfoXLM $\times$ 5, XLM-R $\times$ 1
System 5	Majority Vote	InfoXLM $\times$ 5, XLM-R $\times$ 1

Table 2: Configurations of our submitted systems. For ensemble methods, we used predicted labels from pre-trained language models as an input feature. For systems with stacked generalization, we trained a logistic regression model as a meta estimator.

mance on unseen data. For submission, we use either stacked generalization (Wolpert, 1992), where we train a machine learning model using predictions from pre-trained language models and corresponding idiomaticity labels and then make a final decision with it, or naïve majority voting on predicted labels. We implemented stacked generalization using scikit-learn (Pedregosa et al., 2011). Prediction labels on the development set were used as training data for a meta estimator. To train the estimator, we first divided the training data into 90% for training and 10% for hold-out. We then trained the estimator using three-fold cross-validation (CV). We tested both a ridge classifier and a logistic regression and chose the best-performing model based on the average CV score over the three validation folds. We subsequently picked up the best estimator from the resulting three models using the hold-out set. Finally, the best estimator predicted labels for the test set using the predictions of the pre-trained language models.

Submitted Systems We submitted the five models listed in Table 2 to the evaluation phase.[6] Note that all models were fine-tuned with the span-based classification approach following our preliminary experiments on the development and evaluation sets.

6 Results

Table 3 shows the official test set results for the zero-shot setting of Subtask A. Our best-performing model (System 2), using four InfoXLM models and one XLM-R model with stacked generalization, achieved a macro F_1 score of 0.7466 and was ranked fifth among 20 teams.

Ensemble We utilized ensemble methods in four out of five submissions, of which two use stacked generalization and the others adopt a naïve majority voting approach. Table 4 lists the results of our five submissions on the test set. The results indicate the effectiveness of the ensemble methods, which outperform the model with no ensemble methods by 0.0112 for the best-performing model using stacked generalization. Even for the naïve majority voting approach, the performance improved or did not fall below the result without ensembling.

Classification Approaches We verified the efficacy of three idiomaticity classification approaches—span-based classification, span prediction-based classification, and conventional `[CLS]`-based classification—using the same pre-trained model (InfoXLM). We can see in Table 5 that the span-based classification approach exhibited by far the best average macro F_1 score of 0.7303 on the test set, compared to average

Rank	Team	Macro F_1
1	clay	0.8895
2	yxb	0.8498
3	NER4ID	0.7740
4	HIT	0.7715
5	**Hitachi** (Ours)	0.7466
	Baseline	0.6540

Table 3: Top five macro F_1 scores on test set in zero-shot setting of Subtask A. Baseline uses mBERT following Tayyar Madabushi et al. (2021).

Approach		Macro F_1
No Ensemble	System 1	0.7354
Majority Vote	System 3	0.7354
	System 5	0.7448
Stacking	System 2	**0.7466**
	System 4	0.7452

Table 4: Macro F_1 test scores for our five submitted systems. All models were trained with the span-based classification approach. **Bold** indicates the best result.

[6]For the detailed configurations of each model, please see Appendix D.

Approach	Macro F$_1$ Development	Macro F$_1$ Test
Span-based	**0.7898** (.0138)	**0.7303** (.0211)
Span prediction-based	0.7514 (.0086)	0.6245 (.0255)
`[CLS]`-based	0.7166 (.0675)	0.6333 (.0371)

Table 5: Average macro F$_1$ development and test scores of three classification approaches with standard deviations over ten runs in parentheses. We fine-tuned InfoXLM and used the same hyperparameter settings and input features for all models.

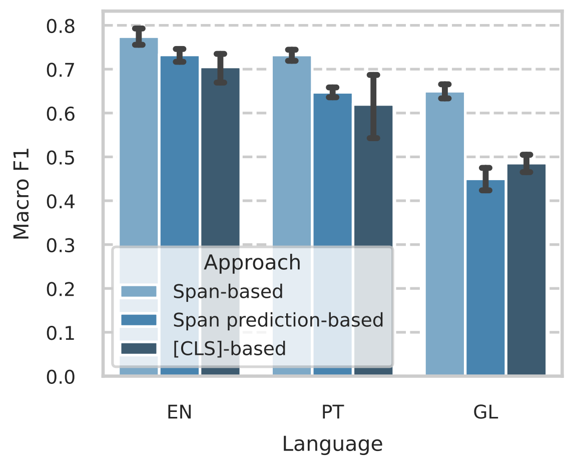


Figure 2: Average macro F$_1$ scores of three idiomaticity classification approaches on the test set grouped by language. Error bars denote 95% confidence interval.

macro F$_1$ scores of 0.6245 and 0.6333 for the span prediction-based and the `[CLS]`-based approaches, respectively. This huge difference stems partly from the Galician classification performance, since we have no associated training or development sets for Galician.

Figure 2 shows the average macro F$_1$ scores on the test set grouped by language. The span-based classification approach produced the highest performance across the three languages, and the performance variations among languages were relatively small, with the maximum difference of 0.1245 between English and Galician. In contrast, the span prediction-based and `[CLS]`-based approaches did not perform well on Galician samples, exhibiting average macro F$_1$ scores of 0.4499 and 0.4858, respectively. We assume that because idioms are generally language- and culture-specific[7] (Aldahesh, 2013; Al-kadi, 2015), it is difficult for models fine-tuned on English and Portuguese data to detect idiomaticity in unseen Galician samples without letting them know where they should be mainly looking, as in the span-based approach.

[7]Although Portuguese and Galician have strong historical ties, they are categorized as two different languages (Ramallo and Rei-Doval, 2015; Garcia, 2021).

Feature	Macro F$_1$ Development	Macro F$_1$ Test
Plain	0.7859 (.0131)	0.7131 (.0196)
Plain + MWE	0.7835 (.0078)	**0.7315** (.0179)
Plain + Context	0.7898 (.0138)	0.7303 (.0211)
Plain + MWE + Context	**0.7918** (.0141)	0.7280 (.0212)

Table 6: Average macro F$_1$ development and test scores with standard deviations over ten runs in parentheses. "Plain" denotes a target sentence, while "Context" represents the previous and next sentences. We fine-tuned InfoXLM using the span-based classification approach and used the same hyperparameter settings for all models.

Input Features Tayyar Madabushi et al. (2021) reported that encoding a target sentence along with its adjacent contexts showed the best classification performance in the zero-shot setting among the four possible input feature combinations: (i) a target sentence, (ii) a target sentence with its MWE as a second sentence, (iii) a target sentence with its adjacent contexts, and (iv) a target sentence, its MWE and adjacent contexts. Here, we also investigated these combinations using the span-based classification approach (Table 6). The results indicate that considering a target sentence and its adjacent contexts is not always the best option. In our experiments, utilizing a target sentence and its target MWE as inputs (Plain + MWE) achieved the best average macro F$_1$ score of 0.7315, followed by Plain + Context with a macro F$_1$ score of 0.7303. While using only a target sentence showed comparable performance to the other approaches on the development set, it ended up producing the worst result on the test set. These results suggest that using an additional feature along with a target sentence is likely to boost detection performance, but it is not clear which combination of input features yields the best performance given the standard deviations.

7 Conclusion

In this paper, we have proposed two approaches for detecting idiomaticity in a given sequence: span-based classification and span prediction-based classification. While the performance of the latter was almost on par with that of the well-known standard sequence classification approach using a `[CLS]` hidden representation, the former outperformed it and showed the best macro F_1 score of 0.7466, which ranked fifth in the zero-shot setting of Subtask A. We also found that it is essential to guide a model on which tokens to look at when no training data is available for a particular language. In future work, we will investigate a more effective idiomaticity detection approach against unseen language data.

Acknowledgements

We would like to thank anonymous reviewers for their valuable feedback and Dr. Masaaki Shimizu for maintenance and management of large computational resources.

References

Abdu Mohammad Talib Al-kadi. 2015. Towards idiomatic competence of yemeni efl undergraduates. *Journal of Language Teaching and Research*, 6(3):513.

Ali Yunis Aldahesh. 2013. On idiomaticity in english and arabic: A cross-linguistic study. *Journal of Languages and Culture*, 4(2):23–29.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Andrea Cascallar-Fuentes, Alejandro Ramos-Soto, and Alberto Bugarín Diz. 2018. Adapting SimpleNLG to Galician language. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 67–72, Tilburg University, The Netherlands. Association for Computational Linguistics.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021a. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Zewen Chi, Li Dong, Bo Zheng, Shaohan Huang, Xian-Ling Mao, Heyan Huang, and Furu Wei. 2021b. Improving pretrained cross-lingual language models via self-labeled word alignment. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3418–3430, Online. Association for Computational Linguistics.

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2021. Rethinking embedding coupling in pre-trained language models. In *International Conference on Learning Representations*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. *Cross-Lingual Language Model Pretraining*. Curran Associates Inc., Red Hook, NY, USA.

Giovanni Da San Martino, Alberto Barrón-Cedeño, Henning Wachsmuth, Rostislav Petrov, and Preslav Nakov. 2020. SemEval-2020 task 11: Detection of propaganda techniques in news articles. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1377–1414, Barcelona (online). International Committee for Computational Linguistics.

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

Ilya Dimov, Vladislav Korzun, and Ivan Smurov. 2020. NoPropaganda at SemEval-2020 task 11: A borrowed approach to sequence tagging and text classification. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1488–1494, Barcelona (online). International Committee for Computational Linguistics.

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. *ArXiv*, abs/1909.07755.

Marcos Garcia. 2021. Exploring the representation of word meanings in context: A case study on homonymy and synonymy. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3625–3640, Online. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Assessing the representations of idiomaticity in vector models with a noun compound dataset labeled at type and token levels. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2730–2741, Online. Association for Computational Linguistics.

Reyhaneh Hashempour and Aline Villavicencio. 2020. Leveraging contextual embeddings and idiom principle for detecting idiomaticity in potentially idiomatic expressions. In *Proceedings of the Workshop on the Cognitive Aspects of the Lexicon*, pages 72–80, Online. Association for Computational Linguistics.

Dawid Jurkiewicz, Łukasz Borchmann, Izabela Kosmala, and Filip Graliński. 2020. ApplicaAI at SemEval-2020 task 11: On RoBERTa-CRF, span CLS and whether self-training helps them. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1415–1424, Barcelona (online). International Committee for Computational Linguistics.

Milton King and Paul Cook. 2018. Leveraging distributed representations and lexico-syntactic fixedness for token-level prediction of the idiomaticity of English verb-noun combinations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 345–350, Melbourne, Australia. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Francis Jeffry Pelletier. 1994. The principle of semantic compositionality. *Topoi*, 13(1):11–24.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

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

Fernando Ramallo and Gabriel Rei-Doval. 2015. The standardization of galician. *Sociolinguistica*, 29(1):61–82.

Giancarlo Salton, Robert Ross, and John Kelleher. 2016. Idiom token classification using sentential distributed semantics. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 194–204, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

J. Sinclair, L. Sinclair, and R. Carter. 1991. *Corpus, Concordance, Collocation*. Describing English language. Oxford University Press.

Caroline Sporleder and Linlin Li. 2009. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 754–762, Athens, Greece. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

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

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5(2):241–259.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Appendices

A Approaches

Here, we describe our two approaches in detail using mathematical notations.

A.1 Span-based Classification

Given $\mathcal{S}$, $\mathcal{W}$, and y_{MWE}, we first tokenize $\mathcal{S}$ and $\mathcal{W}_{\text{MWE}}$ using a pre-trained tokenizer and obtain their token-level representations: $\mathcal{S}' = [t_1, \ldots, t_{n' \in \mathbb{N}}]$ and $\mathcal{W}' = [t_1^{\text{MWE}}, \ldots, t_{m' \in \mathbb{N}}^{\text{MWE}}]$. We then feed $\mathcal{S}'$ into a Transformer-based pre-trained language model and obtain the output hidden representation $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_n']$. We pick up only the hidden representation of $\mathcal{W}'$ and compute their average as $\mathbf{h}_{\text{MWE}} = \frac{1}{|\mathcal{W}'|}\left(\mathbf{h}_1^{\text{MWE}} + \cdots + \mathbf{h}_{m'}^{\text{MWE}}\right)$. Finally, we put $\mathbf{h}_{\text{MWE}}$ into an output linear layer and obtain the prediction. The training objective in this task is the binary cross-entropy loss.

A.2 Span Prediction-based Classification

Given $\mathcal{S}'$, a start position of $\mathcal{W}'$ and an end position of $\mathcal{W}'$, we first feed $\mathcal{S}$ into a Transformer-based model and then put the output representation $\mathbf{H}$ to a linear classifier for classification, yielding $\mathbf{O} \in \mathbb{R}^{n' \times 2} = [\mathbf{o}_{\text{start}}; \mathbf{o}_{\text{end}}]$. We finally apply the softmax function to $\mathbf{o}_{\text{start}}$ and $\mathbf{o}_{\text{end}}$ in order to obtain the idiomatic MWE span probabilities. For prediction, we first calculate the maximum scoring span and obtain its score as $s = o_i^{\text{start}} + o_j^{\text{end}}$, where j must be greater than i, and o_i^{start} and o_j^{end} are the i-th and j-th values of $\mathbf{o}_{\text{start}}$ and $\mathbf{o}_{\text{end}}$, respectively. We also calculate the non-idiomatic score as $\overline{s} = o_1^{\text{start}} + o_1^{\text{end}}$, where index 1 refers to the index of the `[CLS]` token. If $s \geq \overline{s}$, $\mathcal{W}$ is regarded as *idiomatic*. Otherwise, $\mathcal{W}$ is predicted as a non-idiomatic MWE. This task is trained with an average of the log-likelihoods of the correct start $\mathcal{L}_{\text{start}}$ and end $\mathcal{L}_{\text{end}}$ positions: $\mathcal{L}_{\text{span}} = \frac{1}{2}\left(\mathcal{L}_{\text{start}} + \mathcal{L}_{\text{end}}\right)$.

B Cross-lingual Pre-trained Language Models

We briefly explain the five cross-lingual pre-trained language models tested in this paper.

- mBERT (Devlin et al., 2019): Pre-trained with multilingual masked language modeling using Wikipedia. Its architecture follows that of BERT-BASE.

- XLM-R (Conneau et al., 2020): Pre-trained with multilingual masked language modeling using CommomCrawl, which is much larger than Wikipedia. The architecture generally follows that of BERT-LARGE.

- InfoXLM (Chi et al., 2021a): Pre-trained with multilingual masked language modeling, translation language modeling, and the newly proposed cross-lingual contrastive learning, using CommonCrawl. The architecture follows that of XLM-R.

- XLM-Align (Chi et al., 2021b): Pre-trained with multilingual masked language modeling, translation language modeling and denoising word alignment, using CommonCrawl and

Wikipedia. The architecture generally follows that of BERT-BASE.

- RemBERT (Chung et al., 2021): Pre-trained with multilingual masked language modeling using both CommonCrawl and Wikipedia. The architecture is completely different from that of XLM-R, though it has the same number of parameters (559M). It consists of 32 hidden layers, 18 attention heads and $Dim_{\text{hidden}} = 1152$.

Note that all models can accommodate the three target languages (English, Portuguese, and Galician).

C Hyperparameter Settings

Table 8 shows the hyperparameter settings. We explored various hyperparameter combinations with respect to a pre-trained language model, a peak learning rate, and an input feature and selected the models with a macro F_1 score of 0.795 or above on the development set. Note that we also tested XLM-Align, RemBERT, and mBERT in our preliminary experiments, but these did not perform well on the development set (see Appendix E); therefore, we did not use them in our submissions.

D Model Configurations

Table 9 lists the models used for our submissions, while Table 10 shows the configurations of our five submitted systems. For System 1, we used the two different best-performing models for English and Portuguese.[8] For Galician, because we did not have any training samples provided in the zero-shot setting, we used the same model as Portuguese, as both Galician and Portuguese have grammatical and lexical similarities due to their shared historical background (Ramallo and Rei-Doval, 2015; Cascallar-Fuentes et al., 2018; Garcia, 2021).

E Performance Comparison of Five Cross-lingual Pre-trained Language Models

Table 7 compares average macro F_1 scores of five cross-lingual pre-trained language models on the development and test sets. Interestingly, RemBERT produced the best result on the test set with an average macro F_1 score of 0.7452, though it ranked third for the development set. This is presumably because the Galician and English classification performances of RemBERT are better than any other cross-lingual pre-trained language models that we tested (Figure 3).

[8] We selected the best-performing models based on macro F_1 scores on the evaluation set.

Model	**Macro F_1**	
	Development	Test
InfoXLM	0.7898 (.0139)	0.7304 (.0211)
XLM-R	**0.7959** (.0110)	0.7116 (.0125)
XLM-Align	0.7600 (.0096)	0.7015 (.0119)
RemBERT	0.7833 (.0090)	**0.7452** (.0192)
mBERT	0.7440 (.0125)	0.7014 (.0125)

Table 7: Average macro F_1 development and test scores of five cross-lingual pre-trained language models with standard deviations over ten runs in parentheses. We use the same hyperparameter settings for all models.

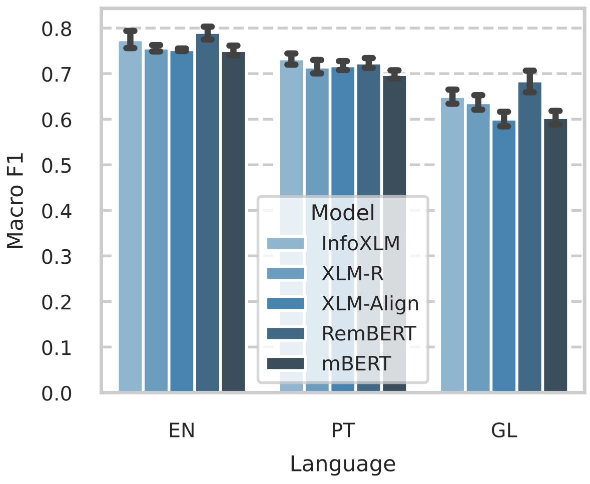


Figure 3: Average macro F_1 scores of five cross-lingual pre-trained language models on the test set grouped by language. Error bars denote 95% confidence interval.

Hyperparameter	**Candidate**
Batch size	16
Epochs	10
Model	(InfoXLM, XLM-R)
Peak learning rate	(0.5e-5, 1e-5, 1.5e-5, 2e-5, 2.5e-5, 3e-5)
Input feature	(Plain + MWE, Plain + Context, Plain + MWE + Context)
Warmup steps	5% of steps
Weight decay	0.01
Adam ϵ	1e-8
Adam β_1	0.9
Adam β_2	0.999
Sequence length	256
Attention Dropout	0.1
Dropout	0.1

Table 8: Hyperparameters in our experiments. We explored various hyperparameter combinations with respect to pre-trained language model, peak learning rate, and input feature. If not specifically mentioned in the paper, we used hyperparameters denoted with an underline.

	Model Type	**Hyperparameter**			**Macro F_1**					
					Development			Evaluation		
		LR	Input Feature	Seed	EN	PT	All	EN	PT	All
$\mathcal{M}_1$	InfoXLM	2e-5	Plain + Context	25	.800	**.814**	.814	**.862**	.678	.801
$\mathcal{M}_2$	InfoXLM	1e-5	Plain + MWE	25	.787	.796	.799	.858	.667	.797
$\mathcal{M}_3$	XLM-R	1e-5	Plain + Context	42	.814	.746	.797	.850	**.743**	**.817**
$\mathcal{M}_4$	InfoXLM	1e-5	Plain + Context	42	.799	.770	.797	.844	.679	.792
$\mathcal{M}_5$	InfoXLM	1e-5	Plain + MWE + Context	22	.788	.784	.796	.854	.696	.803
$\mathcal{M}_6$	InfoXLM	1.5e-5	Plain + Context	42	**.837**	.742	.810	.800	.665	.762
$\mathcal{M}_7$	InfoXLM	2e-5	Plain + MWE + Context	29	.808	.806	**.818**	.854	.708	.812

Table 9: List of models used in our submissions and their macro F_1 scores on the development and evaluation sets. **Bold** indicates the best result in each category, while underline indicates the second-best result.

System	**Approach**	**Models Used**	**Macro F_1**			
			EN	PT	GL	All
System 1	No Ensemble	$\mathcal{M}_1$ for EN, $\mathcal{M}_3$ for PT & GL	**.820**	.733	.614	.735
System 2	Stacking	$\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5$	.783	**.761**	.663	**.747**
System 3	Majority Vote	$\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5, \mathcal{M}_6$	.785	.739	.647	.735
System 4	Stacking	$\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5, \mathcal{M}_7$	.785	.757	.660	.745
System 5	Majority Vote	$\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5, \mathcal{M}_7$	.769	.753	**.685**	.745

Table 10: Configurations of our submitted systems and their macro F_1 scores on the test set. **Bold** indicates the best result in each category, while underline indicates the second-best result.

UAlberta at SemEval 2022 Task 2: Leveraging Glosses and Translations for Multilingual Idiomaticity Detection

Bradley Hauer, Seeratpal Jaura, Talgat Omarov, Grzegorz Kondrak
Alberta Machine Intelligence Institute, Department of Computing Science
University of Alberta, Edmonton, Canada
{bmhauer,seeratpa,omarov,gkondrak}@ualberta.ca

Abstract

We describe the University of Alberta systems for the SemEval-2022 Task 2 on multilingual idiomaticity detection. Working under the assumption that idiomatic expressions are non-compositional, our first method integrates information on the meanings of the individual words of an expression into a binary classifier. Further hypothesizing that literal and idiomatic expressions translate differently, our second method translates an expression in context, and uses a lexical knowledge base to determine if the translation is literal. Our approaches are grounded in linguistic phenomena, and leverage existing sources of lexical knowledge. Our results offer support for both approaches, particularly the former.

1 Introduction

In this paper, we describe the University of Alberta systems for the task of classifying multi-word expressions (MWEs) in context as either *idiomatic* or *literal* (Tayyar Madabushi et al., 2022). Each instance in the data includes a MWE (e.g., *closed book*), its language, and its context, composed of the three surrounding sentences. We participate in both the zero-shot and one-shot settings.

While the exact definitions of the two key terms are not stated explicitly in the task description[1], it is suggested that *idiomatic* is synonymous with *non-compositional*. The Pocket Oxford Dictionary defines *idiomatic* as "not immediately comprehensible from the words used," and *literal* as "taking words in their basic sense." Therefore, we adopt the following MWE *compositionality criterion*

$$\text{literal} \equiv \text{compositional} \equiv \neg\,\text{idiomatic}$$

where the three terms are considered to be Boolean variables. In addition, the shared task considers all proper noun MWEs (e.g., *Eager Beaver*) as literal.

[1]https://sites.google.com/view/semeval2022task2-idiomaticity

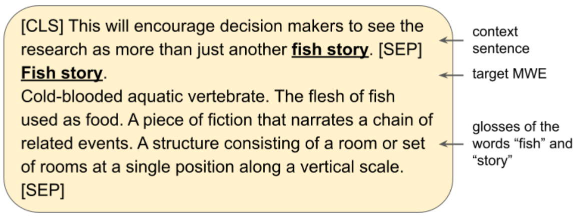

[CLS] This will encourage decision makers to see the research as more than just another **fish story**. [SEP]
Fish story.
Cold-blooded aquatic vertebrate. The flesh of fish used as food. A piece of fiction that narrates a chain of related events. A structure consisting of a room or set of rooms at a single position along a vertical scale.
[SEP]
context sentence
target MWE
glosses of the words "fish" and "story"

Figure 1: An example of defBERT input.

Our goal is to explore the idea that glosses and translations of word senses can help decide whether the meaning of a given MWE occurrence is compositional. Based on the above-stated compositionality criterion, this in turn could facilitate idiomaticity detection. In particular, we hypothesize that at least one of the words in any idiomatic expression is used in a non-standard sense. Following the intuition that a traditional word sense disambiguation (WSD) system can only identify senses that are included in a given sense inventory, we propose two methods that indirectly detect non-standard senses by leveraging either glosses or translations of senses from such an inventory.

Our gloss-based method follows from the intuition that the meaning of a given MWE occurrence is related to any of the existing sense glosses of its component words *only if the expression is compositional*. Therefore, the addition of the glosses to the context of the expression should help the classifier in deciding whether the MWE is used in a literal or idiomatic sense. We implement this method by adding the glosses of each sense of each individual word, retrieved from a lexical knowledge base, to the input to a neural classifier which fine-tunes multilingual BERT (mBERT; Devlin et al., 2019) for the idiomaticity detection task. We refer to this method as defBERT (Figure 1).

Our translation-based method follows from the observation that compositional expressions are typically translated word-for-word ("literally"), which implies that each content word and its translation should have the same meaning. Therefore, each such multilingual word pair should share a multi-synset in a multi-wordnet (Hauer and Kondrak, 2020b). The procedure is as follows: (1) translate the MWE in context; (2) word-align the source and target sentences; (3) lemmatize and POS-tag the source MWE; and (4) for each lemma in the MWE, search for a multi-synset that contains both the lemma and its translation. This method is unsupervised, and we refer to it as MT.

Our results provide evidence that leveraging lexical resources is beneficial for idiomaticity detection. In particular, our gloss-based method, when combined with a type-based UNATT heuristic, is among the top-scoring submissions in the one-shot setting. The heuristic is based on the observation that some MWEs are inherently idiomatic or literal, regardless of their context, which is confirmed by our analysis of the development set annotations.

2 Related Work

Early attempts to represent idiomatic MWEs involve treating idiomatic phrases as individual tokens and learning corresponding static embeddings (Mikolov et al., 2013). However, Cordeiro et al. (2016) show that the effectiveness of this method is limited by data sparsity for longer idiomatic expressions. Furthermore, Shwartz and Dagan (2019) and Garcia et al. (2021) conclude that idiomaticity is not yet accurately represented even by contextual embedding models. Tayyar Madabushi et al. (2021) create a new manually labeled dataset containing idiomatic and literal MWEs, and propose a method based on a pre-trained neural language model.

Regarding using lexical translations for idiomaticity detection, Moirón and Tiedemann (2006) measure semantic entropy in bitext alignment statistics, while Salehi et al. (2014) predict compositionality by presenting an unsupervised method that uses Wiktionary translation, synonyms, and definition information. We extend these ideas by applying machine translation, and consulting a multilingual lexical knowledge base.

Our prior work has already demonstrated the utility of lexical translations for various semantic tasks, including prior SemEval tasks on predicting cross-lingual entailment (Hauer et al., 2020) and contextual synonymy detection (Hauer et al., 2021), as well as word sense disambiguation (Luan et al., 2020), and homonymy detection (Hauer and Kondrak, 2020a; Habibi et al., 2021).

3 Methods

In this section, we describe our methods for idiomaticity detection.

3.1 Baseline mBERT

We re-implemented the mBERT classifier baseline (Devlin et al., 2019) following the methodology of Tayyar Madabushi et al. (2021). The model takes the context sentence and the relevant MWE as an input, and outputs a binary label indicating the idiomaticity of the target MWE. The input sequence is constructed by concatenating the MWE to the end of the context sentence after the special [SEP] token.

It is important to note the differences between our re-implementation and the official baseline provided by the task organizers. In the official baseline, the organizers add the target MWE as an additional feature in the one-shot setting but not in the zero-shot setting. Furthermore, the organizers include the sentences preceding and succeeding the target sentence only in the zero-shot setting. In our re-implementation, we add the target MWE and exclude the preceding and succeeding sentences in both zero-shot and one-shot settings.

3.2 Gloss-based Method

Our first method, defBERT, extends the baseline model by adding the glosses of all possible senses of each individual word in the target MWE to the classifier's input. The intuition is that the addition of the glosses to the input should help the classifier decide if the meaning of the target MWE can be deduced from the definitions of the individual words, i.e., if it is compositional. In the example in Figure 1, the disparity between the context in which *fish story* appears, and the glosses of the various senses of the words *fish* and *story* indicates that the MWE is idiomatic in this context.

The intuition for this method is that non-native speakers can identify idiomatic expressions, provided they understand the standard meanings of the words which comprise them. Suppose that the vocabulary of a non-native speaker covers most of the essential words necessary to understand a language, but not idiomatic expressions. Even if the speaker cannot deduce the meaning of an idiomatic expression in context, they can guess that the expression was used in an idiomatic sense because individual words of this expression do not make sense in the given context.

3.3 Translation-based Method

Our MT method is based on translating the target MWE in context, and leverages multilingual semantic resources. The intuition behind this method is that idioms are generally specific to a particular language, and, being non-compositional, their meanings cannot be conveyed simply by translating the individual words.

Under this hypothesis, to classify an MWE as literal or idiomatic, we need only determine whether the words in the MWE are translated literally. We do this by first identifying the translation of each word via alignment. We then consult a multilingual wordnet, or *multi-wordnet*, a lexical knowledge-base which organizes words in two or more languages into multilingual synonym sets, or *multi-synsets*. Each multi-synset corresponds to a unique concept, and contains the words which express that concept. Given a word in context, and a translation of that word in that context, we consider the word to be literally translated if it shares at least one multi-synset with its translation.

For example, consider an instance in which the MWE *wedding anniversary* is translated into Italian as *anniversario di matrimonio*. Our method checks if either of the translation pairs (*wedding*, *matrimonio*) and (*anniversary*, *anniversario*) share a multi-synset in a multi-wordnet. We test two versions of this method: in MT(all), this condition must be satisfied for all content words in the MWE; in MT(one), detecting a literal translation for one word is sufficient to classify the MWE as literal. In addition, multiple languages of translation may be considered.

3.4 Additional Heuristics

The annotation methodology for this shared task includes proper nouns in the literal class. We therefore use a part-of-speech tagger to detect proper nouns; if any word in the MWE is tagged as a proper noun, MT automatically classifies it as literal without further consideration.

In the one-shot setting, we also use a type-based heuristic which we refer to as UNATT. The intuition behind this heuristic is that certain MWEs are inherently idiomatic or literal, regardless of the context that they appear in. If the training data has no example of an MWE in a particular class, the heuristic exploits this fact as evidence that the MWE should always be classified as the opposite, attested class. For example, this heuristic always classifies *life vest* as idiomatic and *economic aid* as literal, as these are the only classes in which these MWEs appear in the training data. In practice, since UNATT returns no classification if the training set contains instances that belong to either class, this heuristic must be used in combination with another method.

3.5 Combination

Our defBERT and MT methods take different views of the data, with the former using a neural language model and gloss information, and the latter using translation and a lexical knowledge base. We therefore consider combining the two methods. In this approach, we independently apply defBERT and MT to a given instance. If the two methods agree, we return the agreed-upon classification; if they disagree, we return a default class, which is a tunable parameter. As with the other methods, we can combine this method with the UNATT heuristic in the one-shot setting.

4 Experiments

We now describe our experiments, including the tools and resources, the experimental setup, the results, and a discussion of our findings.

4.1 Lexical Resources

As lexical resources for sense translations and glosses, we use two different multi-wordnets: BabelNet (BN; Navigli and Ponzetto, 2010, 2012), and Open Multilingual WordNet (OMW; Bond and Foster, 2013). The defBERT method and the alignment tool access BN 4.0 via the provided Java API[2]. For the MT method, we access the BN 5.0 via the HTTP API. We access OMW via the NLTK interface (Bird et al., 2009). For the MT method, we consider the translation of a word to be literal if it shares a multi-synset with the word in either BN or OMW. For lemmatization and POS tagging, we use TreeTagger[3] (Schmid, 2013).

Both BN and OMW contain English glosses for most concepts, but the availability of glosses in other languages varies. In particular, OMW contains no Portuguese or Galician glosses. With BabelNet, we experimented with two techniques: using English glosses for all languages, and using glosses from the language of the instance, i.e. the

[2]`https://babelnet.org/guide`

[3]We use the pre-trained models for English, Portuguese, and Galician from `https://cis.uni-muenchen.de/~schmid/tools/TreeTagger.`

source language, when available. We refer to these variants as defBERT-BN-en and defBERT-BN-src, respectively. Since defBERT uses a multilingual pre-trained language model, it can seamlessly handle input from multiple languages. Furthermore, because of the relatively poor coverage of Galician in the lexical resources (only 54% of glosses are available in this language), we attempt to leverage its close relationship to Portuguese by processing Galician as if it was Portuguese.

4.2 Translation and Word Alignment

We translate the context sentence of each MWE with Google Translate API[4]. We translated English instances into Italian, and Portuguese/Galician instances into English, because of the good coverage of these languages in our resources. We also conducted development experiments with translation into less related languages, as well as with combining translation information from multiple languages, but we observed no consistent improvements.

We align each input sentence with its translation using BabAlign (Luan et al., 2020), which consults BabelNet to refine the alignments generated by a base aligner, FastAlign (Dyer et al., 2013). To further improve the alignment quality, we augment the set of sentence-translation pairs with additional parallel data from the OpenSubtitles parallel corpus (Lison and Tiedemann, 2016). We note that the English-Galician bitext is less than 1% of the size of the other two bitexts.

4.3 mBERT and defBERT

We fine-tune the mBERT-based models using the binary classification objective on the labeled training dataset. In the zero-shot setting, the MWEs in the training data are disjoint from those in the development and test splits, while in the one-shot setting, all MWEs in the development and test splits have at least one example in the training data. In the zero-shot setting, we trained the models only on the zero-shot training set, while in the one-shot setting, we trained the models on both training sets. In particular, we fine-tuned the models for 20 epochs with a maximum sequence length of 256, a learning rate of 2e-5, and a per device batch size of 16, using the HuggingFace Transformers library.[5]

[4]`https://cloud.google.com/translate`
[5]`https://huggingface.co`

4.4 Development experiments

Table 1 contains the results of the following models: the official mBERT-based baseline (row 0) as reported by the shared task organizers, our reimplementation of the official baseline (row 1), three variants of defBERT method which is based on mBERT (rows 2-4), defBERT combined with the UNATT heuristic (row 5), and the MT method combined with defBERT (rows 6-7)[6]. For rows 1-5 we average the macro F1 score obtained over five runs with random initializations.

Our experiments with defBERT explored the impact of adding glosses to the mBERT model, including the source and language of the glosses. With English glosses retrieved from BabelNet, defBERT improves the total score over the mBERT model in the zero-shot setting, especially on Portuguese. The results also suggest that the English glosses may be preferable to glosses in the source language, a finding which could simplify work on lower-resourced languages, where glosses may not be available.

Combining the predictions of the mBERT-based models with the UNATT heuristic improves the one-shot F1 scores in all cases (row 5 vs. row 4).

The MT methods achieve the best results when combined with defBERT on the development set in the zero-shot setting: MT(one) for English (row 6), and MT(all) for Portuguese (row 7). This demonstrates the utility of using lexical translation information for idiomaticity detection when annotated training data is not available.

4.5 Error Analysis

We found that the defBERT method performs slightly better, by about 1% F1, on literal instances as compared to idiomatic instances in the one-shot setting. In other words, the method is less likely to make an error when given a literal instance. We speculate that this is explained by the model's consistent classification of proper nouns as literal expressions. Indeed, a proper noun is identified incorrectly in only one instance. The fraction of idiomatic vs. literal instances is 39% in English and 56% in Portuguese.

For the MT method, a large number of of errors were caused by a literal translation of an idiomatic expression by Google Translate, even though the

[6]After the test output submission deadline, we discovered errors in our implementation of the MT methods. We report our original results for consistency with the official results.

		Development results				Test results							
		Zero-Shot		One-Shot		Zero-Shot				One-Shot			
		EN	PT	EN	PT	EN	PT	GL	ALL	EN	PT	GL	ALL
0	Baseline	66.2	63.9	87.0	86.7	70.7	68.0	50.7	65.4	88.6	86.4	81.6	86.5
1	mBERT	74.6	62.5	85.7	85.9	**75.1**	63.3	**61.1**	68.2	90.0	83.6	86.6	87.7
2	defBERT-BN-src	75.5	64.8	85.4	86.7	72.0	66.4	57.8	67.2	**95.7**	88.5	88.9	92.2
3	defBERT-BN-en	75.3	66.4	87.6	86.6	73.4	**68.4**	59.7	**69.5**	95.0	**89.3**	87.9	91.8
4	defBERT-OMW-en	74.8	64.5	87.1	84.5	71.0	65.6	56.5	66.5	92.4	86.7	88.5	90.1
5	UNATT + defBERT	-	-	**92.0**	**87.7**	-	-	-	-	94.5	89.2	**91.2**	**92.4**
6	MT(one) + defBERT	**77.3**	64.9	84.5	78.0	68.2	54.6	56.3	62.7	85.9	70.6	78.2	80.6
7	MT(all) + defBERT	66.4	**69.2**	73.7	78.0	65.4	62.5	54.3	62.1	80.3	73.8	73.9	77.3

Table 1: The macro F1 scores on the development and test datasets. Our official submissions are in rows 4-7. Where not otherwise specified, defBERT is in the OMW-en configuration.

corresponding expression is not meaningful in the target language. For example, "she was different, like a closed book" is translated into Italian as "era diversa, come un libro chiuso" even though the Italian translation does not carry the meaning of a person being secretive. In a few cases, the translation would simply copy the source language expression, yielding output which is not fully translated. In addition, some correct lexical translations are not in our lexical resources. Finally, a number of incorrect idiomatic predictions could be traced to word alignment errors, especially in cases of many-to-one alignments (e.g., *bow tie* correctly translated as *papillon*).

Manual analysis performed on the development set corroborates our hypothesis that most multi-word expressions are inherently idiomatic (e.g., *home run*) or literal (e.g., *insurance company*). Only about one-third of the expressions are ambiguous in the sense that they can be classified as either class depending on the context (e.g. *closed book*). Our judgements are generally corroborated by the gold labels, with the exception of proper nouns, which are consistently marked as literal. The UNATT heuristic (Section 3.4), which is based on this observation, obtains a remarkable 98.3% precision and 55.8% recall on the set of 739 instances in the development set.

4.6 Test set results

The results on the test set are shown in Table 1. Our best results are produced by defBERT-BN-en in the zero-shot setting, and the combination of defBERT with the UNATT heuristic in the one-shot setting. The latter also obtains the best result on Galician, which demonstrates its applicability to low-resource languages, as this method only requires English glosses.

The results of combining defBERT with MT are well below the baseline, which may be due to a different balance of classes in the test set, omissions in lexical resources, and/or errors in our initial implementation. Another possible reason is that modern idiomatic expressions are often translated word-for-word ("calqued"), especially from English into other European languages. Examples from the development set include *flower child, banana republic*, and *sex bomb*.

5 Conclusion

Our top result ranks third overall in the one-shot setting. The corresponding method is applicable to a wide variety of languages. It takes advantage of the ability of neural language models to seamlessly incorporate textual information such as glosses, even if it is expressed in a different language. These results strongly support our hypothesis that the gloss information of individual words can improve idiomaticity detection. Moreover, our development results support the hypothesis that non-compositional expressions can be identified through their translations. These findings conform with our prior work on leveraging translation for various semantic tasks (Section 2). We hope that this work will motivate further investigation into the role of multilinguality in semantics.

Acknowledgments

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Alberta Machine Intelligence Institute (Amii).

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*, 1st edition. O'Reilly Media, Inc.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual Wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1352–1362, Sofia, Bulgaria.

Silvio Cordeiro, Carlos Ramisch, Marco Idiart, and Aline Villavicencio. 2016. Predicting the Compositionality of Nominal Compounds: Giving Word Embeddings a Hard Time. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1986–1997.

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

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564.

Amir Ahmad Habibi, Bradley Hauer, and Grzegorz Kondrak. 2021. Homonymy and polysemy detection with multilingual information. In *Proceedings of the 11th Global Wordnet Conference*, pages 26–35.

Bradley Hauer, Hongchang Bao, Arnob Mallik, and Grzegorz Kondrak. 2021. UAlberta at SemEval-2021 task 2: Determining sense synonymy via translations. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 763–770, Online.

Bradley Hauer, Amir Ahmad Habibi, Yixing Luan, Arnob Mallik, and Grzegorz Kondrak. 2020. UAlberta at SemEval-2020 task 2: Using translations to predict cross-lingual entailment. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 263–269, Barcelona (online).

Bradley Hauer and Grzegorz Kondrak. 2020a. One homonym per translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7895–7902.

Bradley Hauer and Grzegorz Kondrak. 2020b. Synonymy = translational equivalence. *arXiv preprint arXiv:2004.13886*.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016*, pages 923–929. European Language Resources Association.

Yixing Luan, Bradley Hauer, Lili Mou, and Grzegorz Kondrak. 2020. Improving word sense disambiguation with translations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4055–4065.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Begona Villada Moirón and Jörg Tiedemann. 2006. Identifying idiomatic expressions using automatic word-alignment. In *Proceedings of the Workshop on Multi-word-expressions in a multilingual context*.

Roberto Navigli and Simone Paolo Ponzetto. 2010. BabelNet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216–225.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2014. Detecting non-compositional MWE components using Wiktionary. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1792–1797, Doha, Qatar.

Helmut Schmid. 2013. Probabilistic part-of-speech tagging using decision trees. In *New methods in language processing*, page 154.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477.

HYU at SemEval-2022 Task 2: Effective Idiomaticity Detection with Consideration at Different Levels of Contextualization

Youngju Joung[*]
Dept. of Industrial Security
Dept. of Applied Statistics
Chung Ang University
ojoo.yj@gmail.com

Taeuk Kim[**]
Dept. of Computer Science
Dept. of Artificial Intelligence
Hanyang University
kimtaeuk@hanyang.ac.kr

Abstract

We propose a unified framework that enables us to consider various aspects of contextualization at different levels to better identify the idiomaticity of multi-word expressions. Through extensive experiments, we demonstrate that our approach based on the inter- and inner-sentence context of a target MWE is effective in improving the performance of related models. We also share our experience in detail on the task of SemEval-2022 Task 2 such that future work on the same task can be benefited from this.

1 Introduction

Multi-word expressions (MWEs) are a group of linguistic components containing two or more words with outstanding collocation (Baldwin and Kim, 2010; Constant et al., 2017). MWEs are valuable in that they contribute to enriching the expressiveness of a language, allowing diverse interpretations of their meaning according to the context in which they are located. That is, the semantics of an MWE can be originated from either (i) the direct composition of the literal definitions of its constituents (i.e., compositional meaning) or (ii) its conventional usage in the language (i.e., idiomatic meaning). For instance, given an expression called *wet blanket*, its compositional meaning is ‘a piece of cloth soaked in liquid’, whereas its idiomatic meaning is ‘a person who spoils the mood’ (see Table 1).

While MWEs function as an effective means of improving the abundance of a language, they are also one of the main obstacles that complicate natural language processing (NLP), from the perspective that an NLP model should be able to precisely identify their mode. In addition, the current trend where the goal of most NLP models is chiefly focused on capturing compositionality raises the question of how properly to deal with idiomatic aspects of linguistic expressions (Garcia et al., 2021a,b; Zeng and Bhat, 2021).

Category	**Meaning**	**Example**
Compositional (Non-idiomatic)	A piece of cloth soaked in liquid.	And finally, the snow falls again, this time in a thick, *wet blanket* that encapsulates everything.
Idiomatic	A person who spoils the mood.	When Marie brings him down to earth, it’s not clear if she’s being a passive-aggressive *wet blanket* or if she might have a point.

Table 1: Comparison between the compositional and idiomatic meanings of the expression *wet blanket*.

An intuitive solution for mitigating the aforementioned problem is an introduction of a sophisticated method designed to estimate the idiomaticity of a given expression, which enables the separate processing of the expression according to its category. In a similar vein, we propose a series of techniques for better detecting the idiomaticity of a target MWE by actively exploiting its surrounding context in addition to considering the relationship between metaphors and the notion of idiomaticity.

Participating in SemEval-2022 Task 2, we focus on classifying two-noun compounds into idiomatic and non-idiomatic. The task provides two configurations. In the zero-shot setting, a model’s performance is evaluated on the set of sequences that include MWEs never appeared in the training phase. Meanwhile, in the one-shot setting, our model is exposed to a pair of instances per each MWE during training, one of which shows the idiomatic use of the MWE while the other is an example for the non-idiomatic case.[1] We present a unified framework that can be used in both kinds, paying slightly more attention to the one-shot setting. In extensive experiments, we show that most of our considerations lead to improvement in performance. We also present discourse on the task specification to promote a fair comparison between related models.

[*]Work done while Youngju was an intern at HYU.
[**]Corresponding author.

[1]For more details on the task’s specification, refer to the task description paper (Tayyar Madabushi et al., 2022).

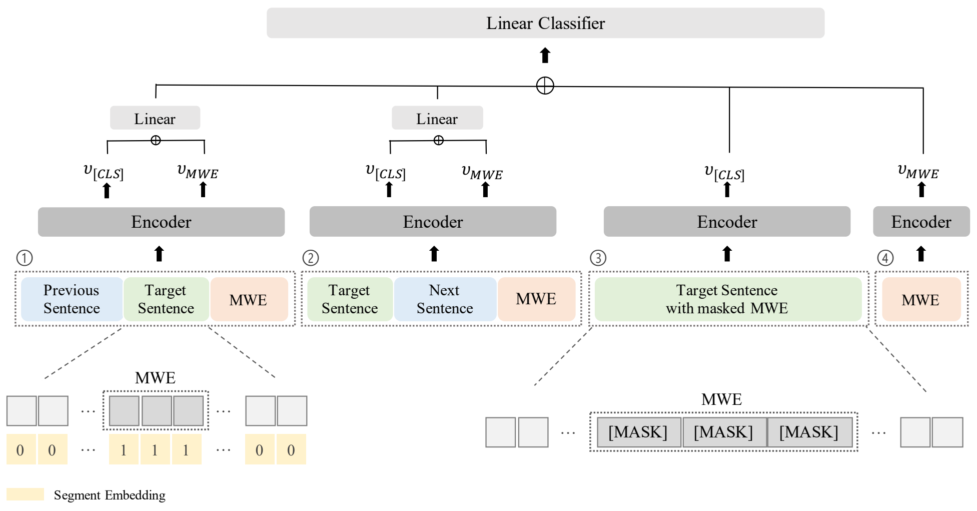


Figure 1: Proposed framework. Two features on the left (① and ②) are based on the surrounding context (Section 3.1), while the remaining two (③ and ④) are originated from considering the inner-sentence context (Section 3.2).

2 Related Work

Idiomaticity detection has been widely studied in the literature (Reddy et al., 2011; Liu and Hwa, 2019; Garcia et al., 2021a,b; Zeng and Bhat, 2021). Above all, Tayyar Madabushi et al. (2021) present a dataset that is the foundation of SemEval 2022 Task 2. This dataset consists of sentences that contain potential idiomatic MWEs with two surrounding sentences and annotations about the fine-grained set of meanings. The authors also evaluate a model's ability to detect idiomatic usage depending on whether context and MWE are included. They report that reflecting the context in the way of simply concatenating surrounding sentences is not generally helpful, and that adding the corresponding MWE at the end of the input sequence improves performance. In the following sections, we re-examine their findings and present our own revision.

On the other hand, we investigate the viability of applying techniques for metaphor detection (Gao et al., 2018; Mao et al., 2019; Lin et al., 2021) into idiomaticity classification, inspired by the resemblance of the two tasks. A metaphor is a form of figurative expression used to implicitly compare two things seemingly unrelated on the surface at the attribute level (Baldwin and Kim, 2010). Not all metaphors have the property of idiomaticity, but some idioms rely on metaphorical composition.

In practice, Choi et al. (2021) introduce two metaphor identification theories (Metaphor Identification Procedure (MIP; Group (2007), Steen (2010)) and Selectional Preference Violation (SPV; Wilks (1975)) into their model to better capture metaphors, which we expect also might be helpful for the procedure of identifying idiomatic expressions. The basic ideas of MIP and SPV are that a metaphor can be identified when we discover the difference between its literal and contextual meaning, and that it can also be detected when its semantics is distinguishable from that of its context. To realize the concepts, for MIP, Choi et al. (2021) employ a target word's contextualized and isolated representations, while for SPV, they utilize the contextualized representations of the target word and the sentence including the word. We adopt some of their ideas and customize them for our purpose, i.e., modeling features for idiomaticity detection.

3 Proposed Method

As a participant of SemEval-2022 Task 2, we propose a framework powered on four features devised to facilitate the detection of idiomatic expressions. These features are computed by the same foundation model,[2] but distinguished from each other by what is inserted into the model as input to compute the features. A simple linear classifier is introduced on top of the concatenation of the four features to finally gauge the idiomaticity of an MWE in an input sequence. Figure 1 presents the overall picture of our method.

3.1 Features Based on Surrounding Context

We first focus on the fact reported by Tayyar Madabushi et al. (2021) that the *surrounding context*

[2]In this work, a 'foundation model' refers to a Transformer *encoder* pre-trained on large corpora, e.g., BERT and XLM-R.

(a.k.a. inter-sentence context), which we define as sentences located right before and after a target sentence, is uninformative in predicting idiomaticity. We hypothesize that this disappointing outcome is partly due to the way such context was exploited.

To be specific, given a sentence containing an MWE and its previous and following sentences, Tayyar Madabushi et al. (2021) propose simply putting all the three together in order. Despite its simplicity, this approach has an explicit drawback that a model should automatically learn how to distinguish the target sentence from its surrounding context. Moreover, when combined with context without caution, the input sequence becomes much (approximately $3\times$) longer than its original form, which might cause a negative effect on performance by merely intensifying the complexity of the problem rather than providing additional information.

To alleviate the aforementioned problems, we suggest a new approach of combining a target sentence with its context. Concretely, we first concatenate the target sentence with its ① previous and ② next sentences *respectively* (i.e., previous-target & target-next), and then inject each chunk into our encoder to derive $v_{\text{[CLS]}}$ and v_{MWE}. By doing so, we expect that the target sentence can be relatively more emphasized than its context, as the target sentence naturally appears twice while its context is exposed only once. Plus, by dividing the whole sequence into two parts, it is anticipated that the encoder struggles less to extract useful information from the input. Note that $v_{\text{[CLS]}}$ is the representation for the entire chunk, which is obtained by taking the [CLS] embedding from the last layer of the encoder, and that v_{MWE} is the average of the representations of the subwords that constitute the target MWE. Lastly, the final context-sensitive feature is computed by conducting a linear transformation of the concatenation of $v_{\text{[CLS]}}$ and v_{MWE}.

On the other hand, we propose two extra techniques in order to provide a clue on the location of MWEs. While constructing token-level representations for our encoder, we employ trainable segment embeddings that draw the line between tokens for the target MWE and others. Moreover, the target MWE is repeated at the end of each chunk, following Tayyar Madabushi et al. (2021).

3.2 Features Based on Inner-Sentence Context

Second, we consider adding features dedicated to more effectively leveraging the information embedded in the target sentence, regarding the MWE and its neighboring words as separate objects. We import some ideas from prior work for metaphor detection (Mao et al., 2019; Choi et al., 2021), exploiting the conceptual relationship between metaphors and idiomatic expressions.

Initially, we assume that similar to Metaphor Identification Procedure (MIP), whose core idea is that a metaphoric word's semantics become distinct from its lexical meaning when it is contextualized, we consider an MWE as idiomatic when its static and contextualized embeddings are heterogeneous. While the contextualized representation of the target MWE is already available from the features provided in Section 3.1, we have not yet introduced the MWE's static representations. To implement this, we again make use of the same encoder, however, only the MWE itself (removed from its context) is presented as input to the model. We call the output of this procedure the ④ *MWE-exclusive* representation, which becomes an ingredient for realizing the 'idiomatic' version of MIP. Note also that according to Garcia et al. (2021b), static models have been considered as competitive or even better to/than contextualized models for idiomaticity detection. Therefore, we aim to reinforce our framework by employing both the options.

Meanwhile, Choi et al. (2021) use Selectional Preference Violation (SPV) for metaphor detection, assuming that the semantics of a metaphoric word should be distinctive from that of its context.[3] We basically adopt their idea, but revise its implementation, arguing that their implementation might be somewhat defective. In detail, Choi et al. (2021) compute the embeddings of a target and its context exactly as we do when computing $v_{\text{[CLS]}}$ and v_{MWE} in Section 3.1. However, it is highly probable that $v_{\text{[CLS]}}$ and v_{MWE} contain similar information as they are intertwined with each other by the attention mechanism, which is undesirable when estimating separate semantics of the target and context. We thus introduce the ③ *context-exclusive* representation of the target sentence by providing our encoder with a variant of the sentence where the target MWE is masked. When combined with the features from the previous section, we expect that the inner-sentence context independent from the target MWE at all can be useful for applying the concept of SPV into idiomaticity detection.

[3]This time, we limit the scope of the context as the sentence emcompassing a target expression (i.e., inner-sentence context), following Choi et al. (2021).

Model / Lang.	English	Portuguese	Galician	Overall
Zero-shot setting				
Baseline (BERT)	70.70	68.03	50.65	65.40
Baseline (XLM-R)	72.29	65.68	46.16	63.21
Ours (submitted)	**76.42**	**72.82**	**62.92**	**72.27**
One-shot setting				
Baseline (BERT)	88.62	86.37	81.62	86.46
Baseline (XLM-R)	88.45	85.03	84.02	86.56
Ours (submitted)	91.59	84.57	82.87	87.50
Ours (post-eval)	**92.29**	**88.05**	**87.10**	**89.96**

Table 2: Main results on the test set. Numbers are from the best configuration (random seed) of each model.

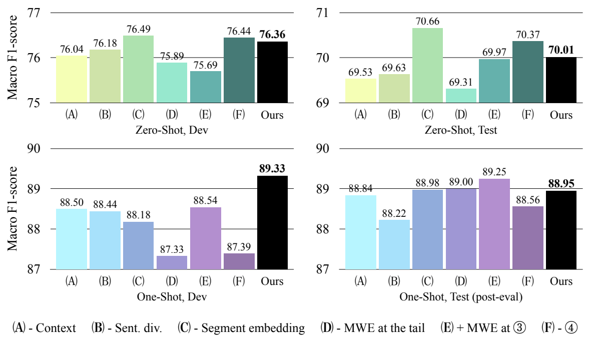


Figure 2: Ablation study.

4 Experiments

4.1 Experimental Setup

For all experiments, we present five instances per each model with the corresponding random seeds (42, 360, 2578, 5925, 9463). Each instance is trained for 10 epochs, and its best checkpoint which shows the top performance on the development set in terms of the macro F1-score is chosen for the inference of the test set. We use a max sequence length of 300, a learning rate of 3e-5 for the AdamW (Loshchilov and Hutter, 2019) optimizer, and a batch size of 16 for the training set and 8 for the validation and test sets. The vectors of each representation ($v_{\text{[CLS]}}$ and v_{MWE}) have 768 dimensions respectively and the learning rate is scheduled to linearly decrease after the second epoch. We leverage XLM-R(-base) (Conneau et al., 2020) as our foundation model.

4.2 Main Results

We compare the results of our method (submitted) against those of the baseline offered by the task organizers (Tayyar Madabushi et al., 2022). Although the original baseline is powered on Multilingual BERT(-base), for a comparison, we also report the performance of the baseline equipped with XLM-R(-base). Evaluation is conducted on the test set, and each model's performance is reported according to the language on which it is tested (English, Portuguese, and Galician) and the setting it is trained (zero- and one-shot).

From Table 2, we can see that both in the zero- and one-shot settings, our model largely outperforms baselines. Notice that in the zero-shot setting, our model outperforms the baseline powered on the same foundation model (XLM-R) by more than 16% in Galician. Considering that Galician was not included in the training data, this result confirms that our model is more generalizable than the baselines from the perspective of input language.

4.3 Ablation Study

We perform an ablation study to confirm whether the elements of our framework are significant. Note that all the results used for comparison are the average of the scores of different instances with five random seeds. Overall, when tested on the validation set, the final version of our approach succeed in outperforming most of the variations, especially in the one-shot setting where our decisions for selecting the final model were made. We present more detailed analysis in the following.

First, we compare our method with the variation (A) which uses only target sentences without surrounding context and the variation (B) which reflects the context by concatenating three sentences. Tayyar Madabushi et al. (2021), where the authors employ the variation (B), previously reported that it is not helpful for idiomaticity detection to consider the surrounding context of a target MWE. However, as shown in Figure 2, we find that taking the context into account following our approach (i.e., separating the context into two chunks) is in fact advantageous in all experimental settings. Furthermore, we observe that the deviation of the scores of our method is much smaller and more stable than that of not considering context. This implies that if there exists a data instance not having much information available from its target sentence, the surrounding context of the target sentence can complements the lack of information.

Contrary to our expectation, it is shown that our method is not always better than the three variations (C), (D) and (E). The variation (C) removes segment embeddings, the variation (D) stops the repetition of MWEs at the tails of ① and ②, and in the variation (E) the target MWE is recovered (not masked) in the computation of '*context-exclusive*' representation (③). We leave a detailed examination regarding these as a follow-up study.

Lastly, it turns out that the variation (F) which removes the '*MWE-exclusive*' representation (④), is more favored in the zero-shot setting. Unlike the one-shot setting, where a pair of positive and negative examples for a particular MWEs can be provided, the zero-shot setting requires the evaluation of MWEs not presented in the training set, which is a more harsher condition for idiomaticity detection models. Therefore, we conjecture that static representations for the MWEs unseen during training become a little bit noisy in the zero-shot setting, failing to function following our intention.

5 Discussion

5.1 Issue on Validation Set in One-shot Setting

In the one-shot setting, we expect that a pair of data instances (one for idiomatic and the other for non-idiomatic) per every MWE in the test set should be provided to the model we train. Likewise, if one wants to confirm that the validation set is rigorous enough to be a substitute for the test set in the procedure of selecting hyperparameters, every MWE in the validation set should have two corresponding instances in the training set. During the competition for SemEval 2022 Task 2, we have discovered that the necessary condition holds for the validation set in the practice phase, while **it does not hold in the evaluation phase**. In other words, the training set provided in the practice phase incorporated data instances that correspond to MWEs in the validation set. However, as the training set has been substituted with a new version, a problem has arisen where MWEs in the newly released training set do not match with those in the validation set. We conjecture that this discrepancy prevents one's optimal actions in choosing the best models.

To prove our hypothesis, we test a variant whose performance on the validation set is not optimal, but has the potential of working well when evaluated on the test set. Specifically, we replicate our experiments, but do not choose the best instance based on validation performance. Instead, we simply choose the model instance trained until 9 epochs and compare it to baselines. As shown in Table 2, we find that the instance chosen based on the validation set (i.e., ours (submitted)) is worse than the randomly selected one (i.e., ours (post-eval)), implying that the inappropriateness of the validation set in the evaluation phase might hinder correct comparisons between submitted models.

Form of MWEs	Validation	Test
Zero-shot setting		
Original form	76.34	**71.72**
Inflectional form	**76.36**	70.01
One-shot setting		
Original form	88.14	**89.80**
Inflectional form	**89.33**	88.95

Table 3: Performance gap with form changes in MWEs.

5.2 Impact of Form Changes in MWEs

When MWEs are repeated at the end of input sequences in ① and ② and embedded solely in ④ in our implementation, we copy them from target sentences so that we can preserve their inflectional form appearing in the sentences, rather than adopting their original form. To confirm the effectiveness of this approach, we conduct an experiment where we replace the MWEs with their original form.

From Table 3, we observe that unlike the case on the validation set where applying inflectional form is always helpful, it turns out that when evaluated on the test set, employing inflectional form is not beneficial for performance improvement, contrary to our expectation. The idea of having utilized the inflectional form of MWEs is from our conjecture that compositional and static representations of MWEs should be computed from the same form for a fair comparison between them. However, it seems that it is more effective to provide a model with a MWE's original form in addition to its inflectional form such that the model can extract more information from the both sides. We leave thorough analysis on this phenomenon as future work.

6 Conclusion

In this work, we investigate the method of implementing better idiomaticity detection models by considering different levels of contextualization. We propose four features grounded on the surrounding and inner-sentence context of a target MWE, showing that these features are effective in improving performance. Moreover, we present a discussion on the issue related to the validation set in the one-shot setting and the impact of the form of MWEs. As future work, we plan to develop a method of designing the interaction between related features in a more sophisticated fashion, instead of simply concatenating them.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2020-0-01373, Artificial Intelligence Graduate School Program (Hanyang University)).

References

Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. *Handbook of natural language processing*, 2:267–292.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor detection via contextualized late interaction using metaphorical identification theories. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1763–1773, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Mathieu Constant, Gülşen Eryiğit, Johanna Monti, Lonneke Van Der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. Multiword expression processing: A survey. *Computational Linguistics*, 43(4):837–892.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 607–613, Brussels, Belgium. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021a. Assessing the representations of idiomaticity in vector models with a noun compound dataset labeled at type and token levels. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2730–2741, Online. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021b. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Pragglejaz Group. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and symbol*, 22(1):1–39.

Zhenxi Lin, Qianli Ma, Jiangyue Yan, and Jieyu Chen. 2021. CATE: A contrastive pre-trained model for metaphor detection with semi-supervised learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3888–3898, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Changsheng Liu and Rebecca Hwa. 2019. A generalized idiom usage recognition model based on semantic compatibility. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6738–6745.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3898, Florence, Italy. Association for Computational Linguistics.

Siva Reddy, Diana McCarthy, and Suresh Manandhar. 2011. An empirical study on compositionality in compound nouns. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 210–218, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial intelligence*, 6(1):53–74.

Ziheng Zeng and Suma Bhat. 2021. Idiomatic expression identification using semantic compatibility. *Transactions of the Association for Computational Linguistics*, 9:1546–1562.

drsphelps at SemEval-2022 Task 2: Learning idiom representations using BERTRAM

Dylan Phelps
Healthy Lifespan Institute
Department of Computer Science, The University of Sheffield
Sheffield, United Kingdom
`drsphelps1@sheffield.ac.uk`

Abstract

This paper describes our system for SemEval-2022 Task 2 Multilingual Idiomaticity Detection and Sentence Embedding sub-task B. We modify a standard BERT sentence transformer by adding embeddings for each idioms, which are created using BERTRAM and a small number of contexts. We show that this technique increases the quality of idiom representations and leads to better performance on the task. We also perform analysis on our final results and show that the quality of the produced idiom embeddings is highly sensitive to the quality of the input contexts.

1 Introduction

Idiomatic expressions present a challenge to Large Language Models (LLMs) as their meaning cannot necessarily be derived from the composition of their component tokens, a trait that LLMs often exploit to create representations of multi-word expressions. The lack of compositionality leads to poor representations for idiomatic expressions and in turn poor performance in downstream tasks whose data includes them.

SemEval-2022 task 2b (Tayyar Madabushi et al., 2022) encourages the creation of better representations of idiomatic expressions across multiple languages by presenting a **Semantic Text Similarity (STS)** task in which correct STS scores are required whether or not either sentence contains an idiomatic expression. The sub-task requires the creation of a self-consistent model in which a sentence including an idiomatic expression and one containing its literal meaning ('*swan song*' and '*final performance*') are exactly similar to each other and equally similar to any other sentence.

To achieve this goal, we investigate whether due to the similarity between idioms and rare-words Schick and Schütze's BERT for Attentive Mimicking (Schick and Schütze, 2020) (BERTRAM) model, which was designed for use with rare-words, can be used to explicitly learn high-quality embeddings for idiomatic expressions. We also investigate how many examples of each idiom are required to create embeddings that perform well on the task, as well as how the quality of contexts fed to the BERTRAM model effects the representations and performance on the task.

Evaluating our model on the task shows that externally trained idiom embeddings significantly increase the performance on STS data containing idioms while maintaining high performance on general STS data. This improved performance gained an overall spearman rank score of 0.6402 and first place (of six entries) on the pre-train setting, and an overall spearman rank score of 0.6504 and second place (of five entries) on the fine-tune setting.[1]

2 Background

Adopting the idiom principle (Sinclair, 1991) to produce a single token representation for MWEs has been used widely within static embedding distributional semantic models (Mikolov et al., 2013; Cordeiro et al., 2019). Within contextualised representation models, Hashempour and Villavicencio, 2020 show that the contextualised representations produced by context2vec (Melamud et al., 2016) and BERT (Devlin et al., 2019) models can be used to differentiate between idiomatic and literal uses of MWEs. However, the MWEs are only represented by one token in the input, before being broken into many tokens using BERTs word piece tokenizer. Tayyar Madabushi et al., 2021 add a token to the BERT embedding matrix and shows that this method improves representations through increased performance on their proposed STS task. The embeddings they add to BERT are randomly initialised, however, and only trained during the fine-tun step on limited data.

[1]The code for creating the embeddings and the modified baseline system code can be found on GitHub: https://github.com/drsphelps/semeval-task-2.

Usage	Example in Sentence
Idiomatic	Blockchains, fundamentally, are banking because what they're doing is allowing the transaction of value across networks … they're doing it in an orthogonally different way," he said Wednesday in what may be his **swan song** in public office.
Literal	Blockchains, fundamentally, are banking because what they're doing is allowing the transaction of value across networks … they're doing it in an orthogonally different way," he said Wednesday in what may be his **bird song** in public office.
Semantically Similar	Blockchains, fundamentally, are banking because what they're doing is allowing the transaction of value across networks … they're doing it in an orthogonally different way," he said Wednesday in what may be his **final performance** in public office.

Table 1: Example sentences for the Idiomatic STS data. Idiomatic and Semantically similar should be given an STS score of 1, and be given the same score when compared to the literal use.

2.1 BERTRAM

BERT for Attentive Mimicking (BERTRAM) (Schick and Schütze, 2020), originally developed to improve representations of rare words, builds upon attentive mimicking (Schick and Schütze, 2019) to create embeddings, within existing embedding spaces, for tokens that incorporate both form and context information from a small number of example contexts. During training the model attempt to recreate embeddings for common words with the existing embedding in the model treated as the 'gold embedding', a process known as mimicking. Form embeddings are then learnt using trained n-gram character embeddings, before being passed with a context into a BERT model. The output of the BERT model forms the embedding for that specific context. To incorporate knowledge from many contexts an attention layer is applied over the outputs for each context to get the final embedding. There exist other models to produce effective embeddings from a small number of contexts (Zhao et al., 2018; Pinter et al., 2017), however, BERTRAM is the only model that is non-bag-of-words and incorporates both form and context information when creating the embedding.

Rare words are unsurprisingly defined by how uncommon they are within datasets. This leads to problems when using LLMs on tasks involving rare words as the word pieces they are broken down into have not been influenced enough during pre-training to accurately represent them. Similarly, idiomatic phrases represent a small proportion of the usage of their constituent words, the idioms in the development set for this task represent an average of 4.9% of the usage of their constituent words. Therefore, the embeddings for constituent words are not significantly effected by the usage of idioms in the training data, leading to the model failing to understand the idiomatic expressions. Further similarities between idioms and rare-words include the variance in compositionality, for example, *unicycle* can be partially understood from its word pieces, whereas *kumquat* cannot.

3 Methodology

3.1 Embedding Creation

Due to the similarities between rare words and idioms, we use BERTRAM to create representations for idiomatic expressions. A separate BERTRAM model is used for each nof the tasks languages. For English, we use the pre-trained model provided with the original paper. For Portuguese and Galician we train BERTRAM models with BERTimbau Base (Souza et al., 2020) and Bertinho-Base (Vilares et al., 2021) respectively used as the base transformers. The Portuguese and Galician BERTRAM models that we train are trained using almost the same training regime outlined for the English model in the original paper, 3 epochs of context only training, 10 epochs of form only training and 3 epochs of combined training. Due to time and compute restrictions, we do not use One-Token Approximation to expand the number of gold standard representations that can be used for attentive mimicking. The Portuguese and Galician splits of the cc100 dataset (Conneau et al., 2020; Wenzek et al., 2020) are used to train the models, with the entire split being used for Galician, and a 10GB subset used for Portuguese.

Contexts for each of the idioms found in the task data can then be created using these models. Examples are retrieved from the relevant split in the cc100 dataset using a grep command [2] that retrieves the entire line that the instance of the idiom is found on. We investigate how changing the number of contexts used to create each embeddings

[2] *grep -i " $val" -m250 en.txt > $val.data*, where $val is the idiom of interest

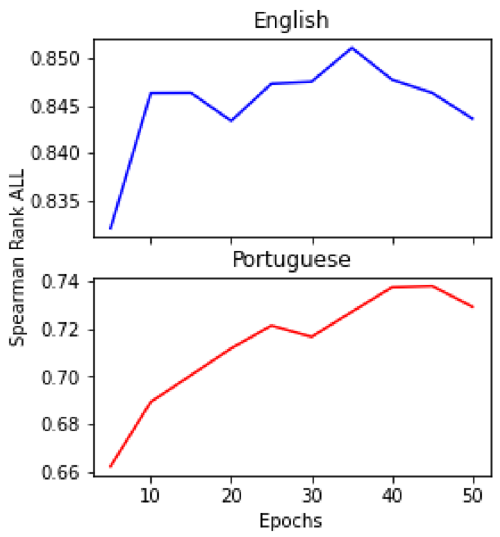


Figure 1: Overall Spearman Rank performance on the development set for the English and Portuguese models at different epochs during pretraining

changes our performance on the task by creating embeddings for each idiom with between 1-250 examples in intervals.

3.2 Model Architecture

For predicting the similarity scores, a separate model is used for each of the languages BERT-Base (Devlin et al., 2019) for English, BERTimbau for Portuguese, and Bertinho-Base for Galician. The created BERTRAM embeddings for each of the idioms found within the task are added into the embedding matrix of the relevant model. These models are used within a Sentence BERT (Reimers and Gurevych, 2019) setup, implemented using the SentenceTransformers library, which consists of a siamese network structure that uses mean squared error over the cosine similarities of the input sentences as it's loss function. This allows us to use the contextualised embedding outputs of our BERT networks to find cosine similarity between a given pair of sentences.

3.3 Data

This sub-task uses data in English, Portuguese and Galician. Data is also split into general STS data which does not necessarily contain idioms and idiom STS data which specifically contains idioms and phrases which are semantically similar or literally similar. An example of idiom STS data taken from the task description can be seen in Table 1.

English and Portuguese are the primary languages and general STS data, from STSBenchmark

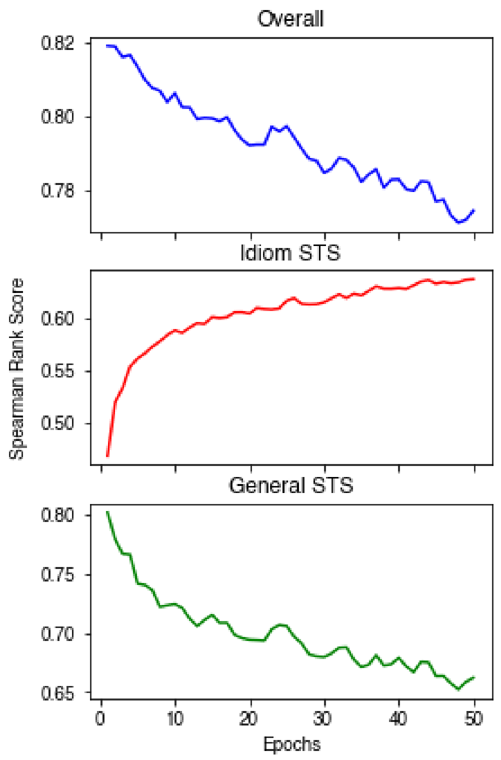


Figure 2: Overall and Idiom STS Only Spearman Rank on the development set whilst training on the Idiom STS data

(Cer et al., 2017) and ASSIN2 (Real et al., 2020) for English and Portuguese respectively, and idiom STS data for both languages are included in the train, dev, eval and test sets. A very small amount (50 examples) of Galician data, comprised of idiom STS data, is also included in the test set.

The task is split into two settings, pre-train and fine-tune. The pre-train setting does not allow for the use of STS score annotated data which includes idioms, whereas any data can be used in the fine-tune setting.

The evaluation metric used in this task is the correlation between the predicted similarities and the gold standard ones, calculated using Spearman's Rank Correlation Coefficient. The Spearman's Rank is calculated for the general STS data and the idiom STS data separately, however, the Spearman's Rank for the entire dataset is used in the final evaluation.

3.4 Pre-train Setting

For the pre-train setting, we use the general STS data in English and Portuguese to train the respective models. Due to a lack of available STS data for Galician, it is trained on the Portuguese data, as

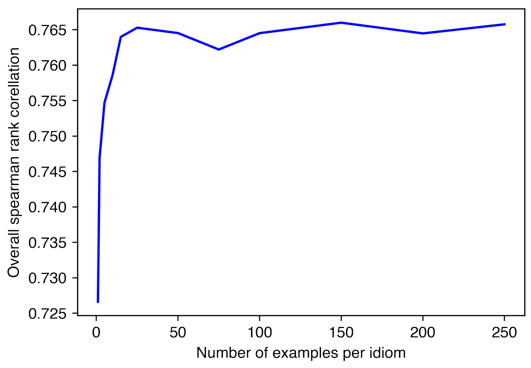


Figure 3: Overall Spearman Rank corellation score on the development set with different numbers of examples used to create the idiom embeddings.

there is a high level of similarity between the two languages.

Evaluating the models on the dev split, we investigate the optimal number of epochs for the English and Portuguese models. The results (shown in figure 1) show that 45 epochs are optimal for Portuguese and 35 for English. Due to a lack of dev split data for Galician we use the result from the Portuguese model as they are trained on the same data.

3.5 Fine-tune Setting

For the fine-tune setting we start with the models from the pre-train setting, and further train them on the Idiom STS data provided as part of the task.

Again we investigate the optimal number of epochs of training on this data (results shown in figure 2). We find that the overall spearman rank is highest after just a single epoch of training, with further training considerably reducing the performance on the general STS data, and thus on the overall STS score. However, further training, up to 50 epochs, continues to increase the performance of the model on Idiom STS data. Therefore, depending on the application and required trade-off, the model can be tuned to either perform better on general STS data or idiom STS data.

3.6 Number of Examples

We also tune the number of examples given for each idiom on the development data. Using BERTRAM we train embeddings for each of the idioms using a range of different numbers of examples from 1-250. The performance of each set of embeddings is evaluated by training the whole system for 10 epochs followed by evaluation on the dev set. Figure 3 shows the results of this experiment. The performance increases quickly from 1-15 examples before flattening out. The absolute highest performance is achieved at 150 examples, and so this is the value we use going forward.

4 Results

The final results for our system on the test data can be seen in Table 2. These scores show significant improvement over the baseline system and led to our system being placed first for the pre-train setting, and second for the fine-tune setting.

Fine-tuning has a much lower effect on the performance of the system when evaluated on the test set than compared with the dev and evaluation sets, with only a small, but significant, rise in overall correlation. Performance rises by only 0.0198 and 0.022 for English and Portuguese respectively, and unlike on dev data we do not see a uniform increase on the SR Idiom score.

4.1 Galician Performance

The performance we achieve on the Galician idiom data is much lower than what is seen on the English and Portuguese data. As we didn't have access to any development data for Galician further investigation will be needed to identify the causes of this discrepancy. Due to the smaller amount of Galician data in the cc100 corpus, some idioms did not have the full 150 examples that were used to create the embeddings for the English and Portuguese idioms. Additionally, there was no Galician STS data to train the final model on, and even though Portuguese and Galician are very similar, the small difference may lead to differences in the performance.

4.2 Error Analysis and Data Issues

To perform analysis on the quality of the created representations we calculate the Spearman's Rank Correlation for each of the idioms in the development set individually. Any idioms with less than 5 occurrences in the development data are removed, as significant correlation scores cannot be achieved with such a low sample size.

When evaluating the performance of the idioms individually, we can see that some of the idiomatic expressions perform much worse than average. For example the spearman rank for score for 'fish story' is just 0.190 when the embedding is trained on 10

Setting	Language(s)	SR ALL	SR Idiom	SR STS
Pre-Train	EN	0.7445	0.4422	0.8709
Pre-Train	PT	0.7087	0.4806	0.8010
Pre-Train	GL	0.2924	0.2924	-
Pre-Train	**All**	**0.6402**	**0.4030**	**0.8641**
Pre-Train	*EN*	*0.5958*	*0.2488*	*0.8300*
Pre-Train	*PT*	*0.5584*	*0.2761*	*0.7745*
Pre-Train	*GL*	*0.1976*	*0.1976*	-
Pre-Train	*All*	*0.4810*	*0.2263*	*0.8311*
Fine-Tune	EN	0.7643	0.4861	0.8344
Fine-Tune	PT	0.7307	0.4643	0.7908
Fine-Tune	GL	0.2859	0.2859	-
Fine-Tune	**All**	**0.6504**	**0.4124**	**0.8188**
Fine-Tune	*EN*	*0.6684*	*0.4109*	*0.6210*
Fine-Tune	*PT*	*0.6026*	*0.4090*	*0.5523*
Fine-Tune	*GL*	*0.3842*	*0.3842*	-
Fine-Tune	*All*	*0.5951*	*0.3990*	*0.5961*

Table 2: Final Spearman Rank (SR) scores of the system on the test set, split into idiom Semantic Text Similarity (STS), general STS, and all datasets. Aggregated results for all languages in bold. Results for the baseline system, also broken down into languages, are in italics.

random examples.

Analysis of these errors shows that the lower performance can, at least in part, be attributed to different phrase senses in the automatically collected examples. Taking our above example '*fish story*', 3 different phrase senses can be observed in the original randomly selected examples: a tall tale, a literal story about fish, and as a proper noun in the title of the film 'A Fish Story'. This leads to a divergence in the contexts in the examples, and the contexts for the idiomatic uses, leading to worse embeddings for the idiomatic phrases.

We can explore this further by producing a manually collected gold standard example set, for the English language subset of the MWEs. Taking the original 250 examples for each idiom, we select 10 gold standard examples. To avoid overfitting our embeddings to this task, we only manually remove examples where the MWE is being used as a proper noun (e.g. the film 'A Fish Story'), or the idiom is being misused, leaving in correct literal and idiomatic uses of the phrase. After removing the proper noun and misused cases, 10 random examples are selected to form our 'gold standard' example set.

We then compare the spearman scores achieved when the embeddings are trained with the gold standard examples, to scores when the representations are produced using 10 random examples when both models are evaluated on the English split of development set. The results for selected MWEs with the randomly selected (auto) and manually chosen (manual) contexts can be seen in table 3.

The manually selected examples lead to an increase in performance on the Idiom STS data split from 0.406 to 0.450. A small increase from 0.841 to 0.848 overall on the English split can also be observed, however this performance is limited by the general STS score which is unaffected by our manual selection. Particularly large improvements in spearman rank coefficient can be seen on MWEs with multiple meanings (panda car, banana republic, fish story, etc.). Surprisingly, we actually see the performance on some MWEs fall, however this can likely be attributed to the random selection of examples, and variance in the contexts used for each idiom, especially on the MWEs which did not have many usages removed as they are only used in the idiomatic form (eager beaver, chain reaction, etc.).

5 Conclusion

We build our system by augmenting BERT models for each language with single token embeddings learnt using BERTRAM. BERTRAM is used due to its high performance on rare words, which share many properties with idioms such as non-compositionality and being rare examples of com-

MWE	Auto	Manual	Change
panda car	0.399	0.851	0.452
banana republic	0.391	0.753	0.362
...	...	...	...
fish story	0.190	0.304	0.114
...	...	...	...
chain reaction	0.356	0.240	-0.116
eager beaver	0.491	0.352	-0.159

Table 3: Improvement in correlation, measured using Spearman's Rank Coefficient, when trained on manually chosen examples vs. automatically collected ones.

ponent pieces. Our results, and subsequent ranking at first place (of six entries) in the pre-train setting and second place (of five entries) in the fine-tune setting, show that BERTRAM can learn high-quality word embeddings for idioms and that this leads to better performance on downstream tasks. Our error analysis shows that BERTRAM is sensitive to the quality of examples it is shown, and that performance can be improved even further by manually selecting a gold set of contexts for each idiom. Future work could look at the differences in performance between the Portuguese and Galician models with the goal of increasing performance on Galician, and perform more analysis to explore the discrepancy in performance between individual idioms further.

Acknowledgements

This work was supported the Healthy Lifespan Institute (HELSI) at The University of Sheffield and is funded by the Engineering and Physical Sciences Research Council [grant number EP/T517835/1].

References

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Silvio Cordeiro, Aline Villavicencio, Marco Idiart, and Carlos Ramisch. 2019. Unsupervised compositionality prediction of nominal compounds. *Computational Linguistics*, 45(1):1–57.

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

Reyhaneh Hashempour and Aline Villavicencio. 2020. Leveraging contextual embeddings and idiom principle for detecting idiomaticity in potentially idiomatic expressions. In *Proceedings of the Workshop on the Cognitive Aspects of the Lexicon*, pages 72–80, Online. Association for Computational Linguistics.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

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

Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 2017. Mimicking word embeddings using subword RNNs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 102–112, Copenhagen, Denmark. Association for Computational Linguistics.

Livy Real, Erick Rocha Fonseca, and Hugo Gonçalo Oliveira. 2020. The assin 2 shared task: A quick overview. In *International Conference on Computational Processing of the Portuguese Language*, pages 406–412. Springer.

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

Timo Schick and Hinrich Schütze. 2019. Attentive mimicking: Better word embeddings by attending to informative contexts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 489–494, Minneapolis, Minnesota. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020. BERTRAM: Improved word embeddings have big impact on contextualized model performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3996–4007, Online. Association for Computational Linguistics.

John Sinclair. 1991. *Corpus, Concordance, Collocation*. Oxford University Press.

Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2020. BERTimbau: pretrained BERT models for Brazilian Portuguese. In *9th Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear)*.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

David Vilares, Marcos García, and Carlos Gómez-Rodríguez. 2021. Bertinho: Galician BERT representations. *CoRR*, abs/2103.13799.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Jinman Zhao, Sidharth Mudgal, and Yingyu Liang. 2018. Generalizing word embeddings using bag of subwords. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 601–606, Brussels, Belgium. Association for Computational Linguistics.

JARVix at SemEval-2022 Task 2: It Takes One to Know One? Idiomaticity Detection using Zero and One-Shot Learning

Yash Jakhotiya[*†]
Georgia Institute of Technology
yashjakhotiya@gatech.edu

Vaibhav Kumar[*]
Georgia Institute of Technology
vaibhavk@gatech.edu

Ashwin Pathak[*]
Georgia Institute of Technology
apathak60@gatech.edu

Raj Shah[*]
Georgia Institute of Technology
rajsanjayshah@gatech.edu

Abstract

Large Language Models have been successful in a wide variety of Natural Language Processing tasks by capturing the compositionality of the text representations. In spite of their great success, these vector representations fail to capture meaning of idiomatic multi-word expressions (MWEs). In this paper, we focus on the detection of idiomatic expressions by using binary classification, based on Subtask A of SemEval-2022 Task 2 (Tayyar Madabushi et al., 2022). Thereafter, we perform the classification in two different settings: zero-shot and one-shot, to determine if a given sentence contains an idiom or not. N shot classification for this task is defined by N number of common idioms between the training and testing sets. In this paper, we train multiple Large Language Models in both the settings and achieve an F1 score (macro) of 0.73 for the zero-shot setting and an F1 score (macro) of 0.85 for the one-shot setting. An implementation of our work can be found at https://github.com/ashwinpathak20/Idiomaticity_Detection_Using_Few_Shot_Learning.

1 Introduction

Transformer-based Large Language Models (LLMs)(Kant et al., 2018) like BERT, DistilBERT, RoBERTa and their variants show state of art performance on a large number of NLP tasks, yet, they fail to capture multi-word expressions such as idioms. This is because contextualized pre-trained models learn compositional representations of text at sub-word and word level to reduce the vocabulary size.

Therefore, we evaluate how well do LLMs identify idiomatcity by formulating the problem as a classification task.

In this paper, we propose an approach for Subtask A of SemEval-2022 Task 2 (Tayyar Madabushi et al., 2022). We treat the development data as held-out development data, and report our performance on the test data. To evaluate how well LLMs identify idiomaticity, we use two different settings to determine the generalizability of the LLMs: zero-shot and one-shot setting. The zero-shot setting is defined such that the MWEs in the train set are mutually exclusive of the MWEs found in the test set. For the one-shot setting, there is only one Idiomatic and/or one Literal training example for one MWE in the development set. This is different from traditional definitions of zero-shot and one-shot classification.

The rest of the paper describes the related works in section II and the dataset used in Section III. Section IV gives the methodology used in zero-shot and one-shot learning. Section V describes the performed experiments and Section VI discusses the results. Section VII concludes the paper with a discussion on future research prospects and directions.

2 Related Work

Idiomaticity identification for MWEs has been widely studied for single token representation using statistical and semantic methods (Lin, 1999; Baldwin and Villavicencio, 2002).

Recent works use contextual representations without any token representation for idiomaticity identification for MWEs (Hashempour and Villavicencio, 2020). (Tayyar Madabushi et al., 2021) introduces new tokens for MWEs into a contextual pre-trained language model. However, they do not explore the relationship of potential MWEs in a sentence.

To this end, we present a contextual and compositional network incorporating latent semantic

[*] Equal contribution
[†] Corresponding author

significance of MWEs in a sentence. Using word embeddings for semantic similarity have been explored before (Katz and Giesbrecht, 2006). However, the challenge for the semantic usage identifcation of MWEs lies in the ambiguity in meanings of MWEs. Additionally, low frequency occurrences of MWEs inhibit the models to effectively learn the contextual representations as well.

Siamese Networks have been widely used for similarity detection and difference tracking. We propose to carry forward this idea for identification of idioms in MWEs by comparing the literal usage of MWEs from their idiomatic usage. This enables our approach to learn a contextual and compositional structure within a sentence.

3 Method

SemEval 2022 task 2 Subtask A (Tayyar Madabushi et al., 2021, 2022) is a task to evaluate the extent to which models can identify idiomaticity in text through a coarse-grained classification into an "Idiomatic" or "Non-idiomatic" class. To better evaluate a model's ability to generalise and learn in a sample efficient fashion, the scores are reported in the zero-shot and one-shot setups.

Data

The dataset used in this report is the one provided by (Tayyar Madabushi et al., 2021). Each of the train and development splits of this dataset consists of samples containing a target sentence, it's language information, a multiword expression (MWE), two contextual sentences that occur before and after the target sentence, and a label associated with the target. The label represents whether the multiword expression was used in an idiomatic sense or not.

The train split is further divided into zero-shot and one-shot data, containing 4491 and 140 samples each, consisting of 236 and 100 distinct MWEs respectively. Similarly, the development data contains 739 samples made from 50 different MWEs. One-shot MWEs have no overlap with zero-shot ones. However, development data MWEs are a proper subset, as can be expected in a one-shot classification scenario.

Zero-shot learning

For the zero-shot learning task, we use the train data to build a classifier using large language models like BERT-multilingual-uncased, DistilBERT-multilingual-uncased, XLM-RoBERTa-large and XLM-net. This task is "zero-shot" in nature as the idioms used in the train set and the development set are distinct. Therefore, we capture the discrepancy in the contextual meaning for idiomaticity, that is, we aim that our classifier distinguishes on the basis of lack of semantic correctness of literal meaning in the presence of an idiom in a sentence.
To make sure that idioms are not used explicitly while pre-training in large language models, we run a natural language inference task on BART-Large-MNLI and RoBERTa-Large-MNLI with the hypothesis as "idiom". The macro F1 score for both approaches is 0.51 and 0.50 respectively, which proves that there is no semantically learnt concept of "idiomaticity" by the model. No training data was used for this step.

We therefore use multilingual LLMs to build classifiers for this setting. We need multilingual classifiers as the data consists of idioms in three languages: English, Portuguese and Galician. We further analyse the majority voting approach on the predictions of trained classifiers (inference based ensembling).

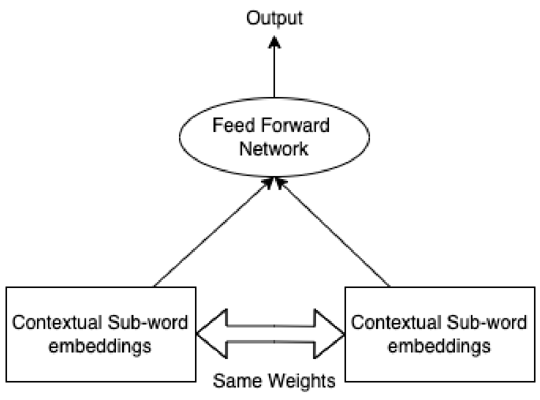

Output
Feed Forward Network
Contextual Sub-word embeddings
Same Weights
Contextual Sub-word embeddings

Figure 1: One-shot learning framework

One-shot learning

In the one-shot setting, we use the only positive and/or the only negative training example, as available for each MWE in the development set. Note that the actual examples in the training data are different from those in the development set in both settings.

As shown in Fig 1, our model relies on finding similarity or relation scores between two input sentences. We first train this model on the pretext task of predicting whether two sentences with the same MWE belong to the same class. To achieve this goal, we employ contextual word embeddings to encode two sentences into feature vectors via an embedding function f_θ. The feature vectors are then combined with an operator $\mathcal{O}(.,.)$ to output $\mathcal{O}(f_\theta(x_i), f_\theta(x_j))$ on two inputs x_i and x_j. This is finally passed to a similarity/relation function g_ϕ to give score $s_{i,j}$ as,

$$s_{i,j} = g_\phi(\mathcal{O}(f_\theta(x_i), f_\theta(x_j)))$$

We test this framework with two underlying models - a Siamese Neural Network (Koch et al., 2015) and a Relation Network (Sung et al., 2018). With the Siamese Network, the operator $\mathcal{O}(.,.)$ is the element-wise difference between the two input feature vectors. The function g_ϕ is a fully connected layer followed by sigmoid activation. The loss in this case can be defined as, $L(s_{i,j}) = \Sigma_{i,j} 1_{y_i=y_j} \log(s_{i,j}) + (1 - 1_{y_i=y_j}) \log(1 - s_{i,j})$, where y_i and y_j are the labels associated with x_i and x_j. Similarly, for the Relation Network, $\mathcal{O}(.,.)$ becomes the concatenation operator, g_ϕ becomes three fully connected layers with non linear activations followed by a sigmoid activation function. The loss in this case is the MSE loss, $L(s_{i,j}) = \frac{1}{n}\Sigma_{i,j}(s_{i,j} - 1(y_i == y_j))^2$. In both of the models, x_i, x_j pairs are samples with matching MWEs.

We propose a novel inference methodology for our binary classification problem, where we also consider a dissimilarity score $1 - s_{i,j}$, with x_i, x_j belonging to support and query sets respectively. Support set is defined to be all samples with the same MWE as the query. We find the maximum of similarity and dissimilarity scores for all examples in the support set, and assign the same label or the opposite depending on whether the maximum was the similarity or the dissimilarity score. This helps us with (Tayyar Madabushi et al., 2021) dataset where one-shot training data doesn't have samples for both the classes (idiomatic and non-idiomatic) for all MWEs.

4 Experiments

Zero-shot learning

We run our experiments on pre-trained models for zero-shot classification. We use Multilingual BERT, Multilingual DistilBERT, BERT-Portuguese, XL-Net and XLM-RoBERTa for exhaustive comparison and evaluation. We ensemble XL-NET, XLM-RoBERTa, and Multilingual DistilBERT in a majority vote based setting. As per the SemEval task, our baseline is Multilingual BERT for classification.

One-shot learning

For the contextual embeddings, we run our experiments on pre-trained compositional multilingual base models BERT, DistilBERT and XLM-RoBERTa for exhaustive comparison and evaluation. We run Siamese networks with cross entropy loss and Relation Networks with an MSE loss.

Our hyperparameter search pointed towards a dropout rate of 0.5, a learning rate of 2e-5 and we found AdamW to be the best performing optimizer.

5 Results

LN	Model	Dev F1
EN	BERT	0.65
EN	DistilBERT	0.70
EN	XLM-RoBERTa	**0.73**
EN	XL-NET	**0.73**
EN	Ensemble	0.71
PT	BERT	**0.64**
PT	DistilBERT	0.58
PT	XLM-RoBERTa	0.63
PT	XL-NET	0.62
PT	Ensemble	0.53
EN-PT	BERT	0.67
EN-PT	DistilBERT	0.70
EN-PT	XLM-RoBERTa	0.71
EN-PT	XL-NET	**0.73**
EN-PT	Ensemble	0.68

Table 1: Zero-shot evaluation results

LN	Emb Model	Siamese F1	Relation F1
EN	BERT	0.79	**0.85**
EN	DistilBERT	0.79	0.83
EN	XLM-RoBERTa	0.83	**0.85**
PT	BERT	0.81	0.84
PT	DistilBERT	0.80	**0.85**
PT	XLM-RoBERTa	0.85	**0.85**
EN-PT	BERT	0.80	**0.85**
EN-PT	DistilBERT	0.79	0.84
EN-PT	XLM-RoBERTa	0.84	**0.85**

Table 2: One-shot evaluation results

Zero-shot learning

Table 1 shows F1-scores for different configurations, both ensemble and individual language models, with the baseline model being Multilingual BERT. We observe that the ensemble model performs better than the baseline in case of EN (0.71 F1 score) and EN-PT (0.68 F1 score) as compared to PT (0.53 F1 score) data. We further observe that

Setting	Language	Test F1
Zero-shot	EN	0.7869
Zero-shot	PT	0.7201
Zero-shot	GL	0.5588
Zero-shot	EN,PT,GL	0.7235
One-shot	EN	0.8410
One-shot	PT	0.8162
One-shot	GL	0.7918
One-shot	EN,PT,GL	0.8243

Table 3: Test evaluation results

XL-NET outperforms other models in case of English and Portuguese inputs. Our best performing zero-shot setting results in a 0.72 F1 score on the test split of the dataset,, which is a significant boost from the 0.65 F1 score in the baseline provided by (Tayyar Madabushi et al., 2021).

One-shot learning
Table 2 reports F-1 scores for one-shot learning. We found the best results of our Siamese and Relation network with XLM-RoBERTa (0.85 F1-score). We also observed a better score for Portuguese dataset as compared to English dataset on all of our models. Our best performing relation networks get 0.82 F1 score on the test split, which is competitive with (Tayyar Madabushi et al., 2021).

Table 3 breaks down our test set evaluation results by language. GL in the table stands for Galician, which had data only in the test split.

6 Analysis and Conclusion

In this paper we analyzed the effectiveness of large Language Models towards identifying idiomaticity in a given phrase using zero-shot and one-shot classification tasks.

In zero-shot classification, we use inference-level ensembling of different language models and observe that it outperforms BERT baseline in cases where the input language consists of English. This highlights a high degree of disagreement amongst the language models w.r.t Portuguese input, highlighting their brittleness.

For one-shot classification, through Siamese and Relation Networks, we are able to represent the latent semantic relationship among MWEs leading to a much better F1 score than zero-shot classification and competitive with prior work. We believe that the improvement in performance of the relation network comes due to the learn-able nature of the distance function used between query and support data sample, as well as our novel inference methodology which also takes into account the dissimilarity score. Future work for one-shot classification could aim at breaking the barrier of 0.85 F1 score we seem to have hit on the dev set with all embedding base models.

References

Timothy Baldwin and Aline Villavicencio. 2002. Extracting the unextractable: A case study on verb-particles. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Reyhaneh Hashempour and Aline Villavicencio. 2020. Leveraging contextual embeddings and idiom principle for detecting idiomaticity in potentially idiomatic expressions. *In Proceedings of the Workshop on the Cognitive Aspects of the Lexicon*, pages 72–80.

Neel Kant, Raul Puri, Nikolai Yakovenko, and Bryan Catanzaro. 2018. Practical text classification with large pre-trained language models. *CoRR*, abs/1812.01207.

Graham Katz and Eugenie Giesbrecht. 2006. Automatic identification of non-compositional multiword expressions using latent semantic analysis. *In Proceedings of the Workshop on Multiword Expressions*, pages 12–19.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. *In Proceedings of the 32nd International Conference on Machine Learning*.

Dekang Lin. 1999. Automatic identification of non-compositional phrases. *In Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 317—-324.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. 2018. Learning to compare: Relation network for few-shot learning.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

CardiffNLP-Metaphor at SemEval-2022 Task 2: Targeted Fine-tuning of Transformer-based Language Models for Idiomaticity Detection

Joanne Boisson and **Luis Espinosa Anke** and **Jose Camacho Collados**
School of Computer Science and Informatics
Cardiff University
`BoissonJC@cardiff.ac.uk`

Abstract

This paper describes the experiments ran for SemEval-2022 Task 2, subtask A, zero-shot and one-shot settings for idiomaticity detection. Our main approach is based on fine-tuning transformer-based language models as a baseline to perform binary classification. Our system, CardiffNLP-Metaphor, ranked 8th and 7th (respectively on zero- and one-shot settings on this task. Our main contribution lies in the extensive evaluation of transformer-based language models and various configurations, showing, among others, the potential of large multilingual models over base monolingual models. Moreover, we analyse the impact of various input parameters, which offer interesting insights on how language models work in practice.

1 Introduction

Idiomatic language identification is an important task for language understanding. Recent language models are surprisingly accurate at distinguishing literal and figurative use of language, but very little work has been done on measuring their ability to generalize across languages. Even for mainstream languages such as English, there is still little understanding on the way language models process idiomatic expressions (IE's). This SemEval task (Tayyar Madabushi et al., 2022) focuses, in particular, on multi-word expressions (MWE), adding the challenge of representing such expressions in models.

The Subtask A of SemEval Task 2 invited participants to extend the range of existing experiments for multilingual idiomatic language detection. Data were provided in English, Portuguese, and Galician. The task is framed as a binary classification of MWE between an idiomatic and a literal usage. In the zero-shot setting, no Galician example is provided in the training and development set. The MWE of the development, evaluation and test sets are unseen in the training set. In the one-shot setting, exactly one example of the MWE encountered respectively in the practicing and test phase is added to the training data. Therefore, these settings provide a challenging framework in which language models have to learn from few or no examples.

As part of the CardiffNLP-Metaphor team, we used a simple strategy similar to the method employed in the original paper releasing the dataset Tayyar Madabushi et al. (2021). In particular, we assessed the performance of monolingual and multilingual language models on the task. To this end, we compared the performance of these models using different input formats and training parameters. The best results are obtained with a XLM-RoBERTa large (Lample and Conneau, 2019) with 7 epochs, 8 instances per batch, a maximum sequence length of 350, the longest three-sentence context, and including target information (i.e., the embedding and the position of the target in the sentence). Our submitted model was based on the best performance in the development set across both tasks, using a wide range of different inputs and parameters.

Our system ranked 8th with a best f1-macro score of 0.7378 for the zero-shot competition and 7th with a score of 0.8934 for the one-shot competition.[1] The main contributions of this paper are the following:

- We show that the multilingual large RoBERTa model (Liu et al., 2019) performs better than monolingual and base models on the one-shot track, which differs from what was found in the original paper (Tayyar Madabushi et al., 2021).
- We found that XLM-RoBERTa base and large can be unstable, also in comparison with simi-

[1]The script written for our experiments is available in a GitHub repository: `https://github.com/Mionies/CardiffNLP-SemEval-2022-Task2`

Data	Total	EN	PT	GL	%Id.
tr. 0-shot	4491	3327	1164	0	56
tr. 1-shot[2]	140	87	53	0	43
dev.	739	466	273	0	45
eval.	762	483	279	0	?
test	2342	916	713	713	?

Table 1: Dataset description. Number of example for each language and percentage of idiomatic MWE expressions. The labels of the evaluation and test sets are unknown.

lar models. This could also explain the difference in our conclusion and that of Tayyar Madabushi et al. (2021) after exploratory runs with large models.

- We confirm the importance of providing the embeddings of the MWE separately to the model, and running a relatively large amount of epochs (up to 9 leads to improvements). Our best model is obtained with seven epochs.

- We test various input formats, including maximum sequence length and context length, and the impact of shuffling the training data on the results, allowing us to discuss the results obtained in previous experiments with this dataset.

2 Related Work

In this task, idiomatic expressions are either frozen (well-known) metaphors, or frozen noun compounds involved in longer metaphors. This dataset relates to other datasets labelled for metaphorical usage of words such as the VU Amsterdam corpus (VUAC) (Steen, 2010) used in a SemEval 2020 task (Leong et al., 2020). However, such datasets are not restricted to idioms or compounds. All the words occurring in texts are labeled. This could ultimately lead to a design of NLP tasks focusing on idioms, but has in the main been used for the predictions of metaphors at the word level. Other metaphor datasets built for NLP such as the LCC corpus (Mohler et al., 2016) may contain some MWE but are not focusing on the specific issues posed by idioms, and also include creative metaphors in their scope.

To the best of our knowledge, there are other five datasets particularly designed for the study of the compositionality of MWE in context in English. The idioms in context (IDIX) corpus (Sporleder et al., 2010) includes idiomatic constructions with non consecutive words (e.g. *raise* one's *eyebrows*) and the phrasal verb corpus (Tu and Roth, 2012) is restricted to *V+PRP* constructions. The SemEval 2013 Task 5b on phrasal semantics is very similar to the task addressed this year, with a division between *known phrases* and *unknown phrases* settings within the binary classification task, but restricted to English. More recently, the MAGPIE corpus (Haagsma et al., 2020), a large repository of 56,622 sentences containing potential idiomatic expressions has been shared with the NLP community. The selection of its initial list of idioms differ from our dataset: after a semi automatic selection of idiomatic expressions, a crowdsourced annotation approach is adopted to determine whether the expression is used metaphorically or literally. In a similar design than the dataset used for Subtask B, Zhou et al. (2021) constructed a curated dataset of sentences pairs: one element containing an idiomatic expression and the second element being the same sentences with the IEs replaced by its literal paraphrase.

As for its connection with language models, Garcia et al. (2021) compared various language models for probing idiomaticity in vector space models. In this work, we go beyond the capabilities of vector space models and test the capabilities of finetuning multilingual language models on the task. The most related work to our analysis is perhaps that done by Zeng and Bhat (2021). They proposed a neural architecture that uses attention flow, designed for the task of detecting whether a sentence has an idiomatic expression and localizing it when it occurs in a figurative sense.

3 Data

Our team participated in Subtask A (zero and one-shot tracks) of the SemEval-2022, Task 2 on Idiomaticity Detection (Tayyar Madabushi et al., 2022). The tasks tackles binary classifiation of MWE in three languages, with variable amount and type of data seen in the training set by the model. Table 1 summarizes the distribution of the instances per language and label. The MWE are all noun compounds, sourced from the Noun Coumpound Senses dataset (Cordeiro et al., 2019).The examples consists of excerpts of text of the Web.

As shown in Table 2, literal instances in our task

[2]One-shot addition designed for the development and the evaluation sets

Example	label	Orig. label
To avoid a **blood bath**, prison officials ordered the gate to be opened.	0	idio.
Remind me *to shed a* **crocodile tear** or two over't.	0	meta usage
Marketing consultant Katy Williams saw the potential of social media.	1	non-idio.
Deborah Loomis is [...] known for [...] Foreplay (1975) and **Blood Bath** (1976).	1	prop. n.

Table 2: Examples of labelled instances with their original four labels in Tayyar Madabushi et al. (2021) and grouping to two labels idiomatic/non-idiomatic for the SemEval binary classifiation Subtask A.

include *non-idiomatic* use of MWE and *proper nouns*. Idiomatic instances gathers *idiomatic* use and *literal use within a longer metaphor*.

The experiments are organized along six splits of the data (c.f. Table 1) : training zero-shot, training one shot for evaluation phase, training one shot for test phase, development, evaluation and test sets. Labels were provided to the participants for the training and development sets. The practice and test phase were run on Codalab.

4 System overview and experiments

4.1 System configurations

We test two different configurations to address this binary classification task: one multilingual classifier trained on all the training data at once, and one monolingual classifier per language. For the zero-shot setting in the monolingual classification configuration, we do not have any training examples of Galician. Therefore, we replace the Galician model by a multilingual model trained on the English and Portuguese examples.

We use well-known transformer-based language models: English, Portuguese BERT and Multilingual BERT (Devlin et al., 2019), XLM-RoBERTa base and large (Conneau et al., 2020). For the monolingual models of Galician, we use Bertinho Vilares et al. (2021) model[3], trained on Wikipedia. The cased version of the language models is used in all our experiments, following Tayyar Madabushi et al. (2021) and because the target MWE contain proper nouns.

4.2 Preprocessing

The data are preprocessed to find all the occurrences of the expressions and record their positions in the three sentences provided for each instance of the datasets.

We search for lower case and upper case occurrences, with words separated by a space or a hyphen. Only in the cases where an exact match cannot be found, we also rely on their lemmata to identify MWEs in plural form. For this, we relied on Stanza[4], which covers the three languages of the experiments including Galician.

We find 80% instances with only one occurrence, and 20% with multiple occurrences in the training and development sets. We considered contexts of one or three sentences (the previous and following sentence in the latter case). The positions of the target are recorded for both contexts length. We then generate two versions of tagged sentences, one where only the first occurrence of the target in the core sentence is marked and one with all the occurrences are marked, using special tokens.

4.3 Experiments

All the experiments are done using the Simple Transformers library[5] with a Quadro RTX 8000 GPU. In order to analyse the effect of several variables in the performance, we performed the following experiments on the development set.

Experiment 1: Shuffling the training set. We study the variation of the performances for three seeds (1,2,3), after three shuffles of the training set (A, B, C), for different batch sizes (8, 16, 32, 64). Our goal is to distinguish the variations in the performances due to various parameters modifications from the variation induced by the order in which instances are fed into the model during training.

Experiment 2: Context and input format. A context limited to the core sentence provided for each example (noted *core-sent* in Table 4) is compared to the concatenation of this core sentence with its previous and following sentence (noted *3-sent*). Different maximum sequence lengths (128, 300, 350, 400, 512) are also tested.

We further test the various ways to encode information about the target and its position in the

[3]Huggingface ID: dvilares/bertinho-gl-base-cased

[4]`https://stanfordnlp.github.io/stanza/`

[5]Version 0.62.0, `https://simpletransformers.ai/`

sentence: tagging only the first occurrence of the target in the core sentence (*first*) is compared to tagging all the occurrences of the target within the input text (*multiple*); one option allows the embedding of the target expression to be passed to the model independently from the sentence (*pair*).[6]

When the *tagged* and the *pair* parameters are both set to *False*, the sentence is provided to the model without any indication concerning the target. This baseline configuration is very interesting in order to evaluate the impact of the topic of the text on idiom detection. For example, in the training data, all the occurrences of *blood bath* are idiomatic except for one occurrence of a proper noun (c.f. Table 2). *Blood bath* is more likely to be used idiomatically than literately in many corpora, as long as they are not rare domain-specific archives on vampires relaxing habits. On the contrary, all 21 occurrences of *marketing consultant* are literal.

Experiment 3: Monolingual and multilingual models. Monolingual and multilingual language models are compared with the two configurations introduced in Section 4.1. In this experiment, we measure the ability of the multilingual models to transfer knowledge across English and Portuguese with a comparison between two additional training methods, bringing the experiment to a comparison between three configurations:

1. Fine-tuning monolingual BERT models for English and Portuguese, and Galician for the one-shot setting. Data are split by language, three classifiers are trained.

2. Fine-tuning three multilingual models using the same settings as in 1.

3. Fine-tuning one single monolingual model, with all the data in the two languages for the zero-shot track and three languages for the one-shot track

Experiment 4: Language models size. Previous initial experiments from Tayyar Madabushi et al. (2021) concluded that large models were not performing better than base models, after a few attempts. We explore further the performance of large models in comparison with base ones under various classifier parameters and for different shuffles of the training set.

[6]Table 7 in the Appendix includes more details about the input formats.

Zero-shot					
train set	seed	Batch size			
		8	16	32	64
A	1	0,70	**0,75**	0,74	0,69
	2	0,70	0,73	0,73	0,38
	3	0,31	0,73	0,71	0,73
B	1	0,31	0,73	0,74	0,71
	2	0,72	0,74	0,72	0,72
	3	0,74	0,72	0,73	**0,75**
C	1	0,73	0,73	0,72	0,71
	2	0,72	**0,75**	0,71	0,68
	3	0,74	0,74	0,68	0,71
One-shot					
train set	seed	Batch size			
		8	16	32	64
A	1	0,73	**0,80**	0,73	0,70
	2	0,75	0,74	0,73	0,70
	3	0,31	0,75	0,74	0,73
B	1	0,31	0,75	0,76	0,73
	2	0,71	0,75	0,76	0,72
	3	0,73	**0,78**	0,71	0,71
C	1	0,61	0,71	0,72	0,72
	2	0,71	0,73	0,71	0,69
	3	0,70	**0,76**	0,75	0,71

Table 3: Experiment 1. Results of XLM-RoBERTa base with 1 epoch, max-seq-length=128, for 3 data shuffles and 3 random seeds. A context of 1 sentence is used, with multiple occurrences of the target tagged, and the MWE embedding provided separately to the classifier (pair). Displayed scores are F1-macro for the development set, aggregated for both English and Portuguese.

5 Results

During the exploratory phase, we tested 111 different parameter configurations, shuffling the data before each run. The twenty best models (sorted according to their performances in the one-shot setting) are shown in Table 8 in the Appendix[7]. These results are used in complement to the following experiments for drawing our conclusions.

Experiment 1: Shuffling the training set. With XLM-RoBERTa-base, Table 3 shows that the classifier is very sensitive to the order in which the input data are passed to the model. When the model does not attribute the same label to all instances of the development set, it may vary by 2 points for a given random seed. The model fails to converge for

[7]The complete results are available in the GitHub repository of this paper `https://github.com/Mionies/CardiffNLP-SemEval-2022-Task2/blob/main/param_optimization_shuffe/data.csv`.

Input parameters				Zero-shot			One-shot		
Context	Occ.	Tagged	Pair	EN	PT	EN,PT	EN	PT	EN,PT
3-sent	first	True	True	0.758	0.646	0.737	0.851	0.818	0.846
3-sent	multiple	True	True	0.743	0.627	0.722	0.661	0.361	0.644
3-sent	-	False	True	0.749	0.624	0.723	0.806	0.789	0.810
core-sent	first	True	True	0.735	0.650	0.718	0.855	0.832	0.850
core-sent	multiple	True	True	0.744	0.603	0.708	0.866	0.841	**0.863**
core-sent	-	False	True	**0.769**	0.564	0.724	0.826	**0.853**	0.841
3-sent	first	True	False	0.740	**0.688**	**0.741**	**0.872**	0.773	0.845
3-sent	multiple	True	False	0.281	0.361	0.313	0.788	0.686	0.768
core-sent	first	True	False	0.764	0.513	0.706	0.716	0.541	0.69
core-sent	multiple	True	False	0.774	0.58	0.724	0.777	0.799	0.794
Below, the target not indicated to the model :				Zero-shot			One-shot		
3-sent	-	False	False	0.695	**0.652**	0.699	0.649	0.361	0.611
core-sent	-	False	False	**0.753**	0.588	**0.711**	**0.688**	**0.579**	**0.667**

Table 4: Experiment 2. Contextual and input format parameters. This experiment is run with XLM-RoBERTa-base, 3 epochs, a batch size=8, max-seq-length=512, a lr=4e-05, on 3 seeds with training set shuffle A (c.f. Experiment 1). The results obtained with the best seed is displayed. An average over the three seeds was impossible because the model often does not converge. The metric used is F1 macro, computed on the development set.

Languages		Zero-shot		
Pre-train	Fine-tune	EN	PT	EN,PT
mono	mono	0.786	0.645	0.747
multi	mono	**0.793**	0.664	**0.764**
multi	multi	0.76	**0.686**	0.748
Languages		One-shot		
Pre-train	Fine-tune	EN	EN	EN,PT
mono	mono	**0.897**	**0.873**	**0.892**
multi	mono	0.835	0.783	0.829
multi	multi	0.851	0.809	0.843

Table 5: Experiment 3. Mono and multilingual training data configurations for pretrained models and fine-tuning. XLM-RoBERTa base is used. The experiment ran with 4 epochs, a batch size=8, a lr=2e-05 using one seed [3] and training set shuffle A. The metric used is F1 macro, computed on the development set.

some seeds and shuffle combinations. The problem arises more often with a small batch size of 8, but it also fails to converge once with batch sizes as large as 64 in our experiment. The issue does not disappear for a larger number of epochs. In the exploratory phase, we tried a broad range of training hyper-parameters, and encountered this issue for models trained with 6, 7 and 8 epochs, both with XLM-RoBERTa base and XLM-RoBERTa large.

The multilingual BERT language model shows more stability. With the same datasets and parameters as those used in Table 3, it always obtains a f-score >0.70 in the zero-shot track, and >0.72 in the one shot track. BERT and XLM-RoBERTa perform comparably in the zero-shot experiment but XLM-RoBERTa obtains the best performance in the one-shot setting.

Experiment 2: Context and input format. The results are presented in Table 4. With the experimental settings chosen, it is difficult to draw any conclusion on which context window (*core-sentence* or *3-sentences*) or tagging scheme (*first* or *multiple*) is better for the task. Both Table 8 in the Appendix and the results obtained by Tayyar Madabushi et al. (2021) suggest that providing the embedding of the target MWE separately to the model (*pair*) improves the performance.

Among the two configurations which input the sentences (*core-sentence* or *three-sentences* contexts) to the model without giving any information about the target, one performs consistently better than random for English examples in the development set zero-shot, with F1-scores of 75.3. It suggests that performances of the model may not mainly be due to the discrimination between compositional and non compositional interaction between the target and the context. The topic of the sentence may also have an important influence, which we did not fully analyze in this work.

Experiment 3: Monolingual and multilingual models. The base monolingual and multilingual settings show similar performance, in preliminary experiments (Table 8) and Experiment 3 (Table

Training parameters				Zero-shot			One-shot		
shuffle	length	batch size	model size	EN	PT	EN/PT	EN	PT	EN/PT
A	350	8	base	0.8	0,677	0,77	0,877	0,867	0,879
A	350	8	large	0.785	0.673	0.761	**0.902**	0.902	**0.905**
B	350	8	base	0.795	0.677	0.768	0.888	0.882	0.89
B	350	8	large	0.776	0.698	0.762	0.892	**0.903**	0.9
C	350	8	base	0.774	0.667	0.749	0.868	0.825	0.859
C	350	8	large	0.782	0.677	0.756	0.863	0.825	0.857
A	350	16	base	0.794	0.673	0.764	0.868	0.885	0.879
A	350	16	large	**0.807**	0.689	**0.778**	0.891	0.893	0.896
A	350	32	base	0.797	0.683	0.771	0.871	0.857	0.871
A	350	32	large	0.775	**0.723**	0.768	0.89	0.882	0.891
A	256	8	base	0.768	0.641	0.737	0.898	0.768	0.895
A	256	8	large	0.777	0.719	0.768	0.884	0.886	0.888
A	128	8	base	0.787	0.675	0.761	0.866	0.897	0.882
A	128	8	large	0.784	0.685	0.764	0.867	0.839	0.862

Table 6: Experiment 4. Base and Large XLM-RoBERTa models comparison. The results are averaged over three seeds. All the models are trained with 7 epochs. The input parameters are set to pair=True, multiple=True, context=paragraph, lr.=2e-05. The metric used is F1 macro, computed on the development set.

5). Experiment 3 is a comparison of the models, for one fixed set of parameters and one fixed shuffle of the training set. In this case, monolingual pre-training or fine-tuning with BERT outperforms the exclusive usage of the multilingual XLM-RoBERTa configuration. Overall, XLM-RoBERTa large obtains higher scores than monolingual BERT models, base and large[8], for the two settings and languages. In conclusion, XLM-RoBERTa base is outperformed by monolingual BERT models for some parameters and shuffles, but XLM-RoBERTa large attains 7 of the 10 best overall scores in the one-shot settings, and of 5 of the 10 best results in the zero-shot settings.

Experiment 4: Language Model Size. Table 6 and Table 8 in Appendix A both show that the best performances reached are obtained by XLM-RoBERTa large. The gap between the models is clear with the one-shot track, and unclear for the zero-shot. The pairwise comparison of the base and large models for the zero-shot track shows that the base model often outperforms the large one.

In the one-shot setting, a closest look at the results per seed reveals base and large models show similar results only when a large standard deviation between seeds affect the overall performance of the large model [9].

Tracks and optimal number of epochs. The evolution of the scores between Table 3 and Table 6 shows that the one-shot setting needs more epochs to reach its highest performances than than the zero-shot setting. The F1-macro score increases by 2 points in the Zero-shot between 1 and 7 epochs when it gains 10 points in the one-shot training configuration.

6 Conclusion

In this system description paper, we explained our method to fine-tune transformer-based language models for the task of idiomaticity detection. Beyond the implementation, we also attempted to answer a few practical questions on how these models learn the task, and particularly their optimal parameters and input settings.

As future work, we would like to explore unsupervised approaches (e.g. sentence embeddings especially tuned on in-domain data such as news corpora of English, Portuguese and Galician). We are also planning to explore various methods to input the three contextual sentences, beyond simple concatenation as explored in this paper. Another interesting topic for further research would be to explore the complex compositionality relations occurring also withing the idiomatic expression, as exemplified sometimes in the examples labelled *meta-usage* in this dataset.

[8]Large and base models are both tested for the English classifier during the preliminary experiments (c.f. Table 8).

[9]The F1-macro for EN and PT and seed [1,2,3] in shuffle C are 0.907, 0.764 and 0.9, the standard deviation is 0.081.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Silvio Cordeiro, Aline Villavicencio, Marco Idiart, and Carlos Ramisch. 2019. Unsupervised compositionality prediction of nominal compounds. *Computational Linguistics*, 45(1):1–57.

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

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Hessel Haagsma, Johan Bos, and Malvina Nissim. 2020. MAGPIE: A large corpus of potentially idiomatic expressions. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 279–287, Marseille, France. European Language Resources Association.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 VUA and TOEFL metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29, Online. Association for Computational Linguistics.

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

Michael Mohler, Mary Brunson, Bryan Rink, and Marc Tomlinson. 2016. Introducing the LCC metaphor datasets. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4221–4227, Portorož, Slovenia. European Language Resources Association (ELRA).

Caroline Sporleder, Linlin Li, Philip Gorinski, and Xaver Koch. 2010. Idioms in context: The IDIX corpus. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Gerard Steen. 2010. *A method for linguistic metaphor identification: from MIP to MIPVU*, volume v. 14 of *Converging evidence in language and communication research*. John Benjamins Pub. Co., Amsterdam.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuancheng Tu and Dan Roth. 2012. Sorting out the most confusing English phrasal verbs. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 65–69, Montréal, Canada. Association for Computational Linguistics.

David Vilares, Marcos García, and Carlos Gómez-Rodríguez. 2021. Bertinho: Galician BERT representations. *CoRR*, abs/2103.13799.

Ziheng Zeng and Suma Bhat. 2021. Idiomatic expression identification using semantic compatibility. *Transactions of the Association for Computational Linguistics*, 9:1546–1562.

Jianing Zhou, Hongyu Gong, and Suma Bhat. 2021. PIE: A parallel idiomatic expression corpus for idiomatic sentence generation and paraphrasing. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 33–48, Online. Association for Computational Linguistics.

A Appendix

Context	Pair input	Tag type	Instance input format
core-sent	F	first	**text**: If you sell your **<idiom> insurance company </idiom>**’s stock privately or publicly to raise capital, you’re considered a stock insurance company.
core-sent	F	mult.	**text**: If you sell your **<idiom> insurance company </idiom>**’s stock privately or publicly to raise capital, you’re considered a stock **<idiom> insurance company </idiom>**.
core-sent	T	-	**text a**: If you sell your insurance company’s stock privately or publicly to raise capital, you’re considered a stock insurance company. **text b** : insurance company
core-sent	T	mult.	**text a**: If you sell your **<idiom> insurance company </idiom>**’s stock privately or publicly to raise capital, you’re considered a stock i**<idiom> insurance company </idiom>**. **text b**: insurance company
core-sent	F	-	**text**: If you sell your insurance company’s stock privately or publicly to raise capital, you’re considered a stock insurance company.
3-sent	T	first	**text a**: For example, River Stone Insurance Limited, a domestic insurance company in the United Kingdom, is an alien insurance company in the U.S. Alien insurance companies must file financial statements, auditor’s reports and meet the requirements of the National Association of Insurance Commissioners (NAIC) International Insurers Department before being permitted to sell policies in the U.S. If you sell your **<idiom> insurance company </idiom>**’s stock privately or publicly to raise capital, you’re considered a stock insurance company. Stock insurance companies are owned by their stockholders. **text b**: insurance company
3-sent	T	mult.	**text a**: For example, River Stone Insurance Limited, a domestic **<idiom> insurance company </idiom>** in the United Kingdom, is an alien **<idiom> insurance company </idiom>** in the U.S. Alien insurance companies must file financial statements, auditor’s reports and meet the requirements of the National Association of Insurance Commissioners (NAIC) International Insurers Department before being permitted to sell policies in the U.S. If you sell your **<idiom> insurance company </idiom>**’s stock privately or publicly to raise capital, you’re considered a stock **<idiom> insurance company </idiom>**. Stock insurance companies are owned by their stockholders. **text b**: insurance company

Table 7: Examples of various input formats obtained with the variation over three parameters : Context, pair-input, and and tag type. When tag type is set to first, the first occurrence of the target MWE in the core sentence is marked, even in a three sentences context. Occurrences of the target MWE in its plural form are not marked in this instance, because the singular form of the MWE is found in the sentence.

0 shot EN	0 shot PT	0 shot EN,PT	1 shot EN	1 shot PT	1 shot EN,PT	ep.	batch size	l.r.	tag	pair	ctxt.	occ.	max sq length	split lang.	mod. EN	mod. PT/GL	mod. ML
0,807	0,677	0,772	0,922	0,911	0,921	7	8	2e-05	T	T	par.	mult.	350	F	-	-	xlm rob. large
0,756	0,678	0,740	0,909	0,922	0,916	8	8	2e-05	T	T	par.	mult.	350	F	-	-	xlm rob. large
0,728	0,673	0,722	0,934	0,876	0,915	9	32	4e-05	T	T	par.	mult.	512	T	bert-base	bert-base	bert-base
0,793	0,726	0,783	0,919	0,892	0,911	7	16	2e-05	T	T	par.	mult.	512	F	-	-	xlm rob. large
0,793	0,698	0,767	0,896	0,911	0,904	9	8	2e-05	T	T	par.	mult.	400	F	-	-	xlm rob. base
0,791	0,706	0,773	0,895	0,911	0,904	9	8	2e-05	T	T	par.	mult.	512	F	-	-	xlm rob. large
0,799	0,705	0,774	0,905	0,884	0,900	8	8	2e-05	T	T	par.	mult.	512	F	-	-	xlm rob. large
0,796	0,687	0,771	0,879	0,922	0,899	4	16	4e-05	T	T	par.	mult.	512	F	-	-	xlm rob. large
0,757	0,667	0,740	0,900	0,876	0,895	9	32	4e-05	F	T	par.	first	512	T	bert-base	bert-base	bert-base
0,769	0,710	0,763	0,903	0,867	0,894	5	16	4e-05	T	T	par.	mult.	512	F	-	-	xlm rob. large
0,281	0,361	0,313	0,886	0,891	0,891	7	8	2e-05	T	T	par.	mult.	300	F	-	-	xlm rob. large
0,768	0,659	0,743	0,882	0,895	0,891	6	8	2e-05	T	T	par.	mult.	350	F	-	-	xlm rob. base
0,740	0,683	0,736	0,889	0,875	0,889	9	16	4e-05	F	T	par.	first	512	T	bert-large	bert-base	bert-base
0,765	0,691	0,752	0,905	0,848	0,888	4	8	4e-05	T	T	par.	mult.	512	T	bert-base	bert-base	bert-base
0,736	0,718	0,746	0,903	0,847	0,888	9	16	4e-05	F	T	par.	mult.	512	T	bert-large	bert-base	bert-base
0,762	0,690	0,747	0,886	0,881	0,888	5	32	4e-05	T	T	par.	mult.	512	T	bert-base	bert-base	bert-base
0,757	0,602	0,722	0,860	0,922	0,886	4	16	4e-05	F	T	par.	first	512	F	-	-	xlm rob. large
0,767	0,699	0,757	0,895	0,861	0,886	9	32	4e-05	F	T	par.	mult.	512	T	bert-base	bert-base	bert-base
0,750	0,669	0,736	0,895	0,862	0,886	9	16	4e-05	T	T	par.	first	512	T	bert-large	bert-base	bert-base
0,763	0,701	0,754	0,891	0,860	0,885	9	32	2e-05	F	T	par.	first	512	T	bert-base	bert-base	bert-base

Table 8: Top twenty scores obtained during initial parameter optimization on the development set, sorted according to the F1 macro for one shot on English and Portuguese. The training data is shuffled between each run. All the scores provided for English and Portuguese in the zero-shot and one-shot settings are F1 macro scores.

kpfriends at SemEval-2022 Task 2: NEAMER - Named Entity Augmented Multi-word Expression Recognizer

Min Sik Oh
Alexa AI[*]
ohtrent@amazon.com

Abstract

We present NEAMER - Named Entity Augmented Multi-word Expression Recognizer. This system is inspired by non-compositionality characteristics shared between Named Entity and Idiomatic Expressions. We utilize transfer learning and locality features to enhance idiom classification task. This system is our submission for SemEval Task 2: Multilingual Idiomaticity Detection and Sentence Embedding Subtask A OneShot shared task. We achieve SOTA with F1 0.9395 during post-evaluation phase. We also observe improvement in training stability. Lastly, we experiment with non-compositionality knowledge transfer, cross-lingual fine-tuning and locality features, which we also introduce in this paper.

1 Introduction

Multi-Word Expressions (MWEs) are defined as "idiosyncratic interpretations that cross word boundaries (or spaces)" (Sag et al., 2002). Recent advances in pre-trained language models such as BERT (Devlin et al., 2019) have enhanced performance of Sentence Classification task, however tasks that specifically identify Multi-Word Expressions (MWE) remain unsolved due to its specific idiomatic properties (Garcia et al., 2021; Yu and Ettinger, 2020). This SemEval shared task (Tayyar Madabushi et al., 2022) aims to understand Multi-Word Expressions better by novel classification and sentence similarity tasks.

Named Entity Recognition (NER) is a task to identify Named Entities (People, Organizations etc.) in a sentence. Multiple datasets exist that specifically perform this task, including CoNLL-02/03 Shared Tasks for English, German, Spanish and Dutch (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003). Multi-Word Expressions and Named Entities are similar in a way that they consist of more than one word but they form a single semantic unit. Thus, Named Entities could be seen as a specific type of Multi-Word Expressions (Jackendoff, 1997; Vincze et al., 2011). However they are different from idiomatic expressions.

MWE	Target	Label
gold mine	This means that search data is a **gold mine** for marketing strategy.	0 (Idiomatic)
gold mine	The hashtag “Qixia **gold mine** incident” has been viewed many million of times on the social media site Weibo.	1 (Non-idiomatic)
gold mine	The **Gold Mine**’s plain frontage & sparse, white-walled dining room suggest that it’s a quick-fix refuelling stop rather than a place to linger.	1 (Non-idiomatic)

Table 1: Dataset samples, table from (Tayyar Madabushi et al., 2021). Note that 3rd example is a named entity (The Gold Mine referring to a restaurant).

We propose **NEAMER - Named Entity Augmented Multi-word Expression Recognizer** that aim to utilize non-compositionality shared between two streams of NLP research. We explore transfer learning between NER and idiom classification tasks. We also experiment with "locality features" to augment representations of text.

We have participated in Subtask A which is a multilingual classification task to determine if a given sentence has correct idiomatic usage or not. We have focused our efforts on the OneShot setting, where the goal is to classify the target sentence utilizing the ZeroShot dataset consisting of idioms not found in test set and the OneShot dataset consisting of 1 idiom-label pair for all idioms in test set.

[*]Research unrelated to work

The dataset has been provided by task organizers (Tayyar Madabushi et al., 2021).

Contributions of this paper are :

- NEAMER system which utilizes transfer learning, NER and other locality features to improve performance and stability of MWE classification task.
- Investigation into transfer learning between NER and idiom classification task.
- Performance and error analysis to understand capabilities of transfer learning, cross-lingual fine-tuning and locality features.

2 Methodology

2.1 Idiom and Named Entity

Idioms and named entities are similar in the way that when they are comprised of multiple words, collocated words encode extra semantics while individual words lose their semantics partially or completely. This property is referred as non-compositionality (Baldwin and Kim, 2010). "In a nutshell" means "very briefly, giving only the main points" (Cambridge, 2022) as an idiom; individual words lose their concrete semantics and only the combination specifies intended meaning. Similarly, "Papa John's" refers to "an American pizza restaurant chain" (Wikipedia, 2022) when used as a named entity; in this case, even grammatical functions of individual words are mostly ignored. This similarity is the basis for the transfer learning experiments we performed.

We have discussed similarities, but what about differences? Idioms and named entities refer to completely different usage of MWEs. Idioms are utilized to improve fluency and understandability, or make language more colloquial (Baldwin and Kim, 2010). Named entities are utilized to specify name of persons, organizations and locations (Tjong Kim Sang and De Meulder, 2003) and do not have such social purpose. Correspondingly we can expect certain knowledge to be easily transferable between two tasks, while it may take more epochs to obtain best final performance due to fundamental difference between tasks leading to necessity for "unlearning" the previous fine-tuned task. We explore the ideas in the experiments.

2.2 Transfer Learning and Stability

As discussed in Section 2.1, idioms and named entities show similar non-compositionality. Thus this is the basis for our transfer-learning experiments, where large language models finetuned on NER task are further trained on idiomatic expression classification task. We investigate following ideas in the experiments:

1. We hypothesize that disparity between task types can bring instability. Large language models are known to be unstable during training (McCoy et al., 2019; Zhou et al., 2020). Language models are trained using Masked LM pre-training task. The aim of the Masked LM task is to classify every masked word to original word, which results in classification of each tokens to 30,000 possible labels. In contrast, the task at hand is much simpler, with the aim being to classify whole sentence into 2 labels according to usage of relevant MWE. NER task can bridge this task complexity gap since the aim is to classify each tokens to 9 labels.

2. We hypothesize that non-compositionality understanding of the model can be shared between tasks. NER systems need to understand non-compositionality to correctly predict B-XXX tags. It also predicts multiple named entities per sentence. Thus we assert that enough non-compositionality understanding is learnt during the NER fine-tuning process compared to Masked LM task where each token is predicted independently.

We additionally hypothesize that language-specific knowledge could be improved for the model through fine-tuning with similar language data, which we perform experiments on.

2.3 Locality Features

We design 5 features that are closely related to MWE usage types. Those are the following:

1. Entity - Whether an MWE contains an NER output span, or an NER output span contains an MWE.

2. Capitalization - Whether any word in the MWE is intentionally capitalized (excluding the first word in a sentence and the case where MWE itself is explicitly capitalized in the dataset).

3. "Be a *" - Whether the MWE starts with a be-verb and the article 'a/an'. Same for Portuguese.

4. "The *" - Whether the MWE starts with "the".

5. Quotation - Whether the MWE is surrounded by quotation marks (" or ').

We name them "locality features" because they expand upon specific position of an MWE by looking at adjacent characters. We encode locality features using a deep neural network to give enough

Feature	Total	0 (Idiomatic)	1 (Not-idiomatic)
All	4491	2535	1956
"The *"	720	**366**	**354**
Entity	650	94	**556**
Capitalized	634	50	**584**
Quotation	165	**124**	41
"Be a *"	80	**68**	12
Parenthesis	6	**5**	1

Table 2: Label statistics in ZeroShot data

Model	ENG F1
mBERT-base (baseline)	70.7
xlm-roberta-base	75.5
xlm-roberta-large	**79.0**

Table 3: English ZeroShot F1 on validation data

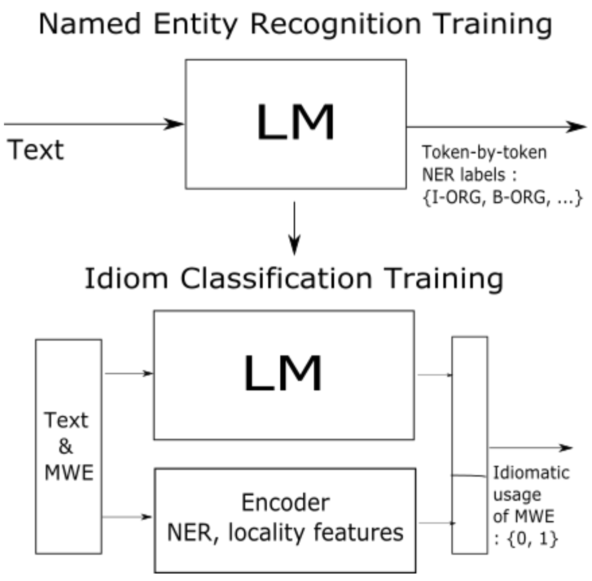

Named Entity Recognition Training
Text
LM
Token-by-token NER labels : {I-ORG, B-ORG, ...}
Idiom Classification Training
Text & MWE
LM
Encoder NER, locality features
Idiomatic usage of MWE : {0, 1}

Figure 1: NER augmented model, see Section 3.2 for details.

significance to the features during training / inference while enabling them to learn complex relationships between the text. This is further informed by label imbalance (excluding "The *" label, which is balanced) shown in Table 2. We perform experiments on whether or not locality features improve the performance on the idiom classification task.

3 Experiment Setup

3.1 Model Selection

Experimental results on English ZeroShot (shown in Table 3) were used to determine pre-trained checkpoints with best performance. We thus selected XLM-Roberta-Large (Conneau et al., 2020) as a starting point for training OneShot models.

The list of checkpoints is: xlm-roberta-base, xlm-roberta-large, xlm-roberta-large-finetuned-conll03-english, xlm-roberta-large-finetuned-conll02-spanish, xlm-roberta-large-finetune-conll03-german, Davlan/xlm-roberta-base-ner-hrl, Davlan/xlm-roberta-large-ner-hrl.

3.2 Model Architecture

Our model training scheme and architecture is presented in Figure 1. We fine-tune the model on NER task with selected language. For the experiments, we utilize NER fine-tuned checkpoints as described in Section 3.1 instead of actually performing NER fine-tuning. Then, we train the NER fine-tuned model with text and idiom (MWE) data for the idiom classification task along with selected locality features. We use two layers of fully connected network to encode locality features that are concatenated to the text representation. Locality features used are described in Section 4.5 and implemented in Python to obtain one-hot vectors which are fed into the fully connected network. The feature encoding and hidden layers of FCN are of size 200. In comparision, LM text encoding is 768 as originally used by XLMRobertaForSequenceClassification class in HuggingFace. The size of encoder feature representation is selected to enhance importance of locality features in comparison to LM representation. We use the classification head provided by the same XLMRobertaForSequenceClassification class.

3.3 Training Procedure

We mostly focus on OneShot setting, using both ZeroShot and OneShot data provided. We used a learning rate of 2×10^{-5} and a batch size of 16 for training our models. Models were trained for 24 epochs and the best checkpoints on the evaluation data were selected. Random seeds of 0, 1, 3, 5, 42 are used for initial experiments. If any of the seeds exhibit training failures due to instability (F1 < 0.5), we perform additional experiments with random seeds 49, 81, 100, 121. This resulted in at least 5 checkpoints for our experiments. All provided training data was used for training the models. We picked checkpoints that perform best on respective languages (EN / PT) for evaluation and submission.

Model	Success
XLM-R	55.6%
XLM-R-EngNER	100%
XLM-R-GermanNER	88.9%
XLM-R-EngNER, Augmented	100%

Table 4: Model training success percentage.

Phase	ALL	EN	PT	GL
Baseline	87.7	88.1	87.0	85.4
Evaluation	93.5	**96.1**	89.9	92.1
Post-Evaluation[3]	**94.0**	**96.1**	**91.1**	**92.8**

Table 5: Best submissions.

[1] We implemented our models in HuggingFace (Wolf et al., 2019) and Pytorch (Paszke et al., 2019). We utilize Tesla V100 NVIDIA GPU for training.

4 Results

4.1 Model Stability

We present observed training success rate for each of the models in Table 4. We define training failure as an observance where F1 of the checkpoint is smaller than 0.5. We observe a very high training failure rate for the XLM-R$_{\text{large}}$ model (44.4%). We assert that this is due to discrepancy between the pre-training task of MaskedLM and the idiom classification task (more discussion in Section 2.2.)

4.2 Best Submissions

We show our best submissions in Table 5. Our best official submission during evaluation phase is ensemble of 3 checkpoints per language consisting of XLM-R$_{\text{large}}$-EngNER & SpaNER, with exception of one XLM-R$_{\text{base}}$-EngNER checkpoint[2]. Best post-evaluation submission is ensemble of 5 checkpoints per language consisting of XLM-R$_{\text{large}}$-EngNER & SpaNER, selected via process described in Section 3.3. We achieved top 2 during the competition (Section 7). We are currently first place in the post-competition leaderboard (4/15/2022).

4.3 Ensemble Model Performance

We submit our models based on the ensemble model performance shown in Table 6. Checkpoints for ensemble were selected via the process described in Section 3.3. XLM-R$_{\text{large}}$ + NER models (xlm-roberta-large-finetuned-conll03-english, xlm-roberta-large-finetuned-conll02-spanish) that represent transfer learning characteristics perform best, with high F1 score across all languages. Interestingly, locality feature augmentation does not seem to enhance the final output compared to the transfer learning only method. This could be due to model checkpoints not having enough variance between them caused by over-reliance on label imbalance. (More discussion in Section 4.5)

Model	ALL	EN	PT	GL
XLM-R	92.7	94.5	89.5	92.3
XLM-R$_{\text{NER}_{\text{HRL, 36}}}$	92.5	**96.1**	88.4	90.3
XLM-R$_{\text{NER}_{\text{ENG, SPA}}}$	**94.0**	**96.1**	**91.1**	**92.8**
XLM-R$_{\text{NER}}{}^{\text{Aug}}$	92.8	95.6	89.4	90.8

Table 6: Test data F1 performance for ensemble models. All XLM-R models are large variant.

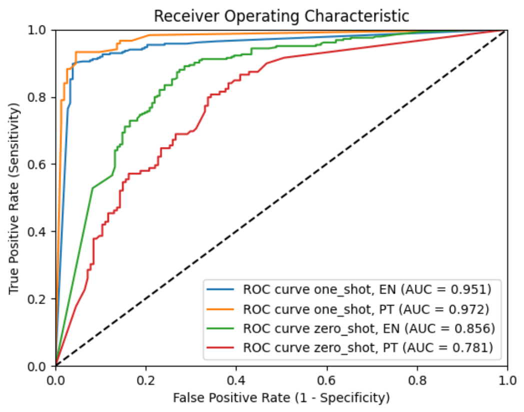


Figure 2: ROC curve of XLM-R$_{\text{NER}}$ on validation data for all tasks. We observe very strong prediction ranking capability for both EN and PT (AUC > 0.950) for OneShot task.

4.4 Average Model Performance

The average F1 scores are presented in Table 7. We observe that additional finetuning on English NER data results in higher performance compared to the baseline XLM-R$_{\text{large}}$ model. Augmentation of the model using locality features results in a slight performance increase. Results suggest that NER fine-tuning assists in the idiom classification task, while locality features help relatively less. NER fine-tuning is helpful due to the language model adapting to the non-compositionality expressed in both tasks (more discussion in Section 2.2.)

[1]Galician test data was inferred by Portuguese model for submission.

[2]The checkpoints were selected according to best performance on validation set.

[3]Experiment performed after end of competition.

Model	Average	Ensemble
XLM-R$_{large}$	93.0	94.5
XLM-R$_{large}$-Eng	**94.0**	**96.1**
XLM-R$_{large}$-Eng, Aug	**94.2**	95.6
XLM-R$_{base}$-HRL	91.1	-
XLM-R$_{large}$-HRL	92.9	-
XLM-R$_{large}$-HRL, 36	**94.2**	**96.1**

Table 7: English test data F1

Model	ALL	EN	PT	GL
XLM-R$_{NER}$	62.3	70.8	67.7	44.4
XLM-R$_{NER}{}^{Aug}$	**64.9**	**72.6**	67.4	**49.2**

Table 8: ZeroShot ensemble test data F1 performance. We note comparatively higher performance for locality feature augmented model on English and Galician data.

4.5 Locality Features

Effect of locality features seem to be marginal, since average F1 (Table 7) only slightly improves in comparison with transfer-learning only model. We also observe lower ensemble performance (Table 6). An enhanced architecture (attention layer in which features explicitly interact with each other) with layer-wise learning rate tuning (to lessen the adverse impact of a cold-start of the feature encoding layers) and dropout (to randomize model training for ensemble enhancement) might be beneficial. We leave it to future work.

We hypothesize that while locality features may be a promising feature to utilize for enhanced architectures, using it by itself may be a relatively too simple indicator. Locality features only require looking at 1~2 specific tokens[4], thus non-compositionality expressed between the tokens themselves is very simple compared to complexity of MWE. An explicit NER feature may also be already encoded in the model via NER fine-tuning step such that no new information is provided during training.

Lastly, we note that we achieve the best ZeroShot setting performance in our experiments with XLM$_{NER}{}^{Aug}$ model which is an ensemble of 3 checkpoints (Table 8). Thus, the locality features could be more promising in the ZeroShot setting where there is less information regarding specific MWE usage. We leave a thorough evaluation to future work.

[4]i.e. Capitalization - first letter of words in MWE, Quotation - ' or " before and after MWE. Parenthesis - (or) before and after MWE.

Model	EN	PT	GL
XLM-R$_{large}$-Eng, Spa	94.0	87.5	88.5
XLM-R$_{large}$-German	93.6	87.2	84.2
XLM-R$_{base}$-HRL	91.1	83.6	83.2
XLM-R$_{large}$-HRL	92.9	84.0	83.7
XLM-R$_{large}$-HRL, 36	**94.2**	85.9	87.2

Table 9: Test data average F1 performance for HRL model variants and English, Spanish and German NER fine-tuned model.

4.6 Crosslingual NER Transfer Learning

XLM-R$_{large}$-HRL is an XLM-R$_{large}$ model trained on NER tasks for 10 languages (Arabic, German, English, Spanish, French, Italian, Latvian, Dutch, Portuguese and Chinese). Rationale for fine-tuning this model is to observe the following :

1. Impact of fine-tuning on a model from a pre-trained model trained on NER data from multiple languages. This model has been trained on all CONLL02 / 03 datasets for English, Spanish, Dutch and German, as well as 8 language specific datasets.

2. Impact of fine-tuning on a model which has been pre-trained with capability to perform Portuguese NER task. This model has been trained on Paramopama and Second Harem (Freitas et al., 2010) Portuguese NER datasets.

We show the results in Table 9. We observe that while XLM-R$_{large}$-HRL performs worse on EN F1 than the similarly fine-tuned XLM-R$_{large}$-English and German, training for 36 epochs (50% epoch increase) yields comparable performance. This aligns with our hypothesis that task-to-task training requires "unlearning" partial aspects of the previous task and thus may take longer to train (more discussion in Section 2.1). XLM-R$_{large}$-English was only trained on CoNLL03 English NER task, while HRL models were trained on NER datasets corresponding to 10 languages - this may result in a higher amount of NER task and language specific knowledge that needs to be removed for the model to train properly.

Similarly, we observe worse performance on Portuguese and Galician results for HRL models compared to Spanish fine-tuned model. Portuguese and Galician seem to require more training epochs than English to achieve comparable performance. This may be due to the difference in dataset size per language in both the ZeroShot and OneShot training data for idiom classification task

Feature	LM	NER	Aug
Capitalized (137)	94.2	94.2	91.2
Entity (131)	93.1	93.1	90.8
"The *" (52)	86.5	**92.3**	**92.3**
"Be a *" (13)	100.0	100.0	100.0
Quoted (12)	90.5	90.5	90.5

Table 10: Micro F1 Metrics (validation data) for each locality feature tagged samples corresponding to XLM-R, XLM-R$_{NER}$ and XLM-R$_{NER}{}^{Aug}$. We observe that transfer learning has improved the performance for "The *" feature. More discussion in Section 5.1.

	Pred 0	Pred 1
Label 0 (Idiomatic)	5	9
Label 1 (Non-idiomatic)	0	117

	Pred 0	Pred 1
Label 0 (Idiomatic)	2	12
Label 1 (Non-idiomatic)	0	117

Table 11: Confusion matrix for Entity in non-augmented models(XLM-R, XLM-R$_{NER}$) vs augmented model (XLM-R$_{NER}{}^{Aug}$).

(English:Portuguese = 2.9:1). We leave training the models on more Portuguese idiom classification datasets and longer epochs to future work.

We also experiment with a model fine-tuned on CoNLL 03 German NER task. We note slightly worse performance for German fine-tuned model compared to models fine-tuned on highly similar languages (English and Spanish NER fine-tuned models). This result seems to suggest that fine-tuning the model on same language for both NER task and Idiom Classification task achieves best performance. More experiments with many languages from other parts of the world could be performed.

5 Error Analysis

5.1 Categorical Performance

We show the F1 metrics for the validation data per each feature in Table 10. We find that the F1 score of "The *" locality feature has increased by 5.8 points after transfer learning is introduced. This locality feature does not directly correspond to NER, and is the only sample-balanced locality feature as shown in Table 2. Thus, we argue that this is further proof of NER transfer learning teaching general non-compositionality to LM that is transferred to MWE classification task.

We also find that Capitalized and Entity F1 scores have stayed the same after the introduction of NER transfer learning, and it has actually decreased by 2~3 points after locality feature augmentation. We also observe a recall decrease of 0.214 (0.357 -> 0.143) as shown in Table 11. As discussed in Section 4.5, this is due to over-reliance on training data label imbalance.

5.2 Sample Analysis

We list the prediction improvements between base XLM-R$_{large}$ model and NER transfer-learning based models in Appendix A. Interestingly, we observe that 6 out of 9 sample prediction improvements for English model are also observed with HRL, German[5] models. This strongly suggests that shared characteristics are present between NER transfer-learning based models. We also observe that the model output changes are not associated with named entities, strengthening our hypothesis of general non-compositionality knowledge transfer between tasks.

6 Conclusion

We present NEAMER - Named Entity Augmented Multi-word Expression Recognizer. This system explores how we can utilize non-compositionality shared between Named Entity and Idiomatic Expressions. We find that the NER transfer learning variant achieves the best MWE classification OneShot performance. We also observe high training stability. We investigate non-compositionality knowledge transfer between tasks and obtain promising results across experiments.

7 Rank Information

During the official evaluation phase, we were top 2 in Subtask A (One-Shot) leaderboard with F1 score of 0.9346 (Table 5). We trained 50 checkpoints and measured F1 on English and Portuguese separately. Checkpoints were generated via process described in 3.3. Best English performing checkpoints inferred on English test submission data, while best Portuguese performing checkpoints inferred on Galician as well as Portuguese test submission data. Finally, we ensembled best performing models on each language using different strategies (including top 3, top 5, top 10) to optimize generalization performance.

[5]German model is not trained on CoNLL03 English data, making the result more interesting.

8 Acknowledgements

We'd like to sincerely thank SemEval Task 2 Organizers, especially Harish Tayyar Madabushi, for organizing the Shared Task and providing valuable correspondence during review. We also thank Young Sun Yoon for sharing his knowledge of Ibero-Romance languages.

References

Timothy Baldwin and Su Nam Kim. 2010. *Handbook of Natural Language Processing, Second Edition*. CRC Press, Boca Raton, USA.

Cambridge. 2022. In a nutshell. In *Cambridge Dictionary*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Cláudia Freitas, Cristina Mota, Diana Santos, Hugo Gonçalo Oliveira, and Paula Carvalho. 2010. Second HAREM: Advancing the state of the art of named entity recognition in Portuguese. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

R. Jackendoff. 1997. *The Architecture of the Language Faculty*. Linguistic inquiry monographs. MIT Press.

R. Thomas McCoy, Junghyun Min, and Tal Linzen. 2019. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. *CoRR*, abs/1911.02969.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for nlp. In *Computational Linguistics and Intelligent Text Processing*, pages 1–15, Berlin, Heidelberg. Springer Berlin Heidelberg.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

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

Veronika Vincze, István Nagy T., and Gábor Berend. 2011. Multiword expressions and named entities in the wiki50 corpus. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 289–295, Hissar, Bulgaria. Association for Computational Linguistics.

Wikipedia. 2022. Papa John's — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Papa%20John's&oldid=1081001743`. [Online; accessed 07-April-2022].

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Lang Yu and Allyson Ettinger. 2020. Assessing phrasal representation and composition in transformers. In *Proceedings of the 2020 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pages 4896–4907, Online. Association for Computational Linguistics.

Xiang Zhou, Yixin Nie, Hao Tan, and Mohit Bansal. 2020. The curse of performance instability in analysis datasets: Consequences, source, and suggestions. *CoRR*, abs/2004.13606.

A Prediction Improvements

We list the classification improvements[6] in validation dataset observed across NER transfer learning models in comparison to base XLM-R$_{\text{large}}$ model. The NER transfer learning models we compare are English, German, and HRL (10 languages). We find 6 samples that prediction have improved consistently across all 3 models, which is 66.7% of prediction improvements in English model.

MWE	**Sentence**	**Feature**
high life	"This is the story of “Memo Fantasma” or “Will the Ghost,” who started life in the Medellín Cartel, funded the bloody rise of a paramilitary army, and today lives the high life in Madrid."	"The *"
home run	He is the only player to hit at least 30 home runs in 15 seasons and is one of only four players to produce at least 17 seasons with 150 or more hits.	-
health check	Big Tech Show · Why your DNA may be your next health check	-
pillow slip	By morning most of it is on the pillow slip, and soap and water will clean up the rest."	"The *"
pillow slip	"Her pillow slip by now was very much askew; one ear pointed northward, the other southeast, and she could only see out of one eye."	-
dry land	And God called the dry land Earth; and the gathering together of the waters called he Seas: and God saw that it was good.	"The *"

Table 12: Improved samples due to NER fine-tuning.

[6]Wrong prediction in XLM-R$_{\text{large}}$ model, but correct prediction in NER transfer learning models.

daminglu123 at SemEval-2022 Task 2: Using BERT and LSTM to Do Text Classification

Daming Lu
ByteDance
daming.lu@bytedance.com

Abstract

Multiword expressions (MWEs) or idiomaticity are a common phenomenon in natural languages. Current pre-trained language models cannot effectively capture the meaning of these MWEs. The reason is that two single words, after combined together, could have an abruptly different meaning than the compositionality of the meanings of each word, whereas pre-trained language models reply on words' compositionality. We propose an improved method of adding an LSTM layer to the mBERT model to get better results on a text classification task (Subtask A). Our result is slightly better than the baseline. We also tried adding TextCNN to mBERT and adding both LSTM and TextCNN to mBERT. We participate in SubTask A and find that adding only LSTM gives the best performance.

1 Introduction

Machine learning has made deep impacts on various areas, such as computer vision (He et al., 2015, 2017; Lu, 2018), computational biology (Jumper et al., 2021; Huang et al., 2019; Lu, 2010, 2009), and natural language processing (Yang et al., 2019b; Lewis et al., 2019; Madabushi et al., 2020) . In natural language processing, large pre-trained models are prevailing and have achieved great successes. Models such as BERT (Devlin et al., 2018), RoBERTA (Liu et al., 2019), XLNet (Yang et al., 2019a), ALBERT (Lan et al., 2020), Ernie (Sun et al., 2019), etc. performed pretty well in tasks such as sentiment analysis, commonsense reasoning (Lin et al., 2019; Lu, 2020), QA system (Chen and Yih, 2020; Yu et al., 2015) and many other tasks. However, these models are not good at certain tasks such as assessing humor and capturing idiomaticity. This shortcoming is largely due to natural languages' flexibility.

In this paper, we focus on how to use large pre-trained language models to determine whether a multiword expression (MWE) has a trivial meaning (Tayyar Madabushi et al., 2022), a.k.a, the compositionality of each word's meaning, or it is an idiomatic usage. We use the dataset provided in (Tayyar Madabushi et al., 2021). In the training set, the target MWE is given. The previous sentence, the target sentence and the next sentence are also given. We need to decide if the MWE has an idiomatic meaning or its meaning is trivial. This task then can be treated as a text classification problem.

The rest part of this paper is organized as follows:

- We first introduce the dataset and the task with details.
- Then we describe how we built up our pipeline with BERT, LSTM and TextCNN.
- We give our results in section 4.
- Lastly, we provide our discussion in section 5.

2 Dataset and Task

As mentioned in (Tayyar Madabushi et al., 2021), the dataset for Subtask A consists of naturally occurring (target) sentences, previous sentences and next sentences. The target sentence contains potentially idiomatic MWEs annotated with a fine-grained set of meanings: compositional meaning and idiomatic meaning(s). Table 1 shows two samples from the training data. One has an idiomatic expression, and the other not.

3 Methods

Our core pre-trained language model is mBERT (Wolf et al., 2020). We chose mBERT over BERT hoping that it could better fit the task's multi-language specification. In traditional methods, n-gram was used to detect and group the MWEs. In

Table 1: Sample data for Subtask A.

previous sentence	target sentence	next sentence	target MWE	label (0 means idiomatic)
"The job has traditionally been non-political, but Mrs. Trump's decision to hire a Trump Organization employee added partisanship to the role, even though Mr. Harleth tried to frame his work there as one stop in a long career in the hospitality industry."	"The White House job was well compensated — former chief ushers say salaries run in the $200,000 range — but the days are long, particularly if the president is an early riser or a night owl; Mr. Trump was both."	Mr. Biden is not a morning person, people familiar with his schedule say.)	night owl	0
Demography expert Piotr Szukalski told Dziennik Gazeta Prawna he thinks that deep concerns about the spread of the coronavirus are to blame.	Minister of Family and Social Policy Marlena Malag ascribed the high death rate to the pandemic and said it would take a long time for the current government program of family benefits intended to boost the birth rate to reverse the negative trend.	"Commenting on data the state agency Statistics Poland released in December for 11 months of 2020, economist Rafal Mundry said the number of deaths was the highest since World War II, and the number of births the lowest in 15 years."	birth rate	1

our methods, we tried to use either LSTM (Hochreiter and Schmidhuber, 1997) or TextCNN (Kim, 2014) to capture the MWEs. We concatenate LSTM or TextCNN to mBERT in order to increase the performance.

3.1 LSTM

Unlike RNN (Jordan, 1997), LSTM is good at remembering only the important parts of a sentence. We hope it can help us group up the MWEs and improve the performance. We add a bidirectional LSTM layer at the output of the sequential transformers. The bidirectional LSTM layer was initialized as 1-layer and bidirectional, with a dropout of 0.1.

3.2 TextCNN

Similar to traditional CNN (Schmidhuber, 2015) in computer vision, TextCNN (Kim, 2014) extracts features from a small area of text. We suppose this layer can help us detect the span of the MWEs so that performance can be improved.

4 Results

We use the mBERT with 12 hidden layers. We did experiments on dropouts with 0.1 and 0.2. As mentioned in Section 3, we explored of adding either a LSTM or a CNN to the final fully connected layer of the transformer from mBERT. Table 2 provides our experiments and results. We were expecting that mBERT + TextCNN could give us the best results. But it turned out that mBERT + LSTM performs best for Subtask A among our experiments. The author has put the code for this paper on GitHub[1].

[1]`https://github.com/daming-lu/semeval_2022_task2_sub_a`

Table 2: Subtask A Experiment Results

Method	Zero-Shot	One-Shot
mBERT	0.6448	0.6987
+LSTM, dp=0.1	0.6546	0.6998
+LSTM, dp=0.2	0.6333	0.6613
+TextCNN, dp=0.1	0.6501	0.6827
+TextCNN, dp=0.2	0.6254	0.6309
+TextCNN+LSTM	0.6502	0.6977
+LSTM, dp=0.1(**test**)	0.654	0.704

5 Discussion

One reason that our method does not boost the performance a lot might be that we add the LSTM or TextCNN to the end, whose effect is limited to the whole pipeline. Another new method, according to (Gao et al., 2021), is that we can turn this classification problem into a masked word problem. In PROMPT, it claims the integration is more genuine, but choosing the prompt could be technical.

Another important reason is overfitting. We tried to increase dropout from 0.1 to 0.2 in order to get rid of overfitting. But the effect was opposite. According to (Tan et al., 2015), adding LSTM could boost question answering tasks, whereas our task is in fact a text classification. This might be the reason of the tiny improvement.

Acknowledgements

The author would like to thank Dr. Harish Tayyar Madabushi for helpful feedback and suggestions.

References

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. 2017. Mask R-CNN. *CoRR*, abs/1703.06870.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

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

Liang Huang, He Zhang, Dezhong Deng, Kai Zhao, Kaibo Liu, David A Hendrix, and David H Mathews. 2019. LinearFold: linear-time approximate RNA folding by 5'-to-3' dynamic programming and beam search. *Bioinformatics*, 35(14):i295–i304.

Michael I. Jordan. 1997. Serial order: A parallel distributed processing approach. *Advances in psychology*, 121:471–495.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589.

Yoon Kim. 2014. Convolutional neural networks for sentence classification.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. *arXiv preprint arXiv:1909.02151*.

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

Daming Lu. 2009. A combined motif discovery method.

Daming Lu. 2010. A gibbs sampling algorithm for motif discovery using a linear mixed model. In *Proceedings of the International Symposium on Biocomputing*, pages 1–6.

Daming Lu. 2018. Use online dictionary learning to get parts-based decomposition of noisy data. In *17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018, Orlando, FL, USA, December 17-20, 2018*, pages 1492–1494. IEEE.

Daming Lu. 2020. Masked reasoner at SemEval-2020 task 4: Fine-tuning RoBERTa for commonsense reasoning. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 411–414, Barcelona (online). International Committee for Computational Linguistics.

Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle. 2020. Cost-sensitive bert for generalisable sentence classification with imbalanced data. *arXiv preprint arXiv:2003.11563*.

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

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2019. Ernie 2.0: A continual pre-training framework for language understanding.

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019a. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Zhou Yu, Alexandros Papangelis, and Alexander I. Rudnicky. 2015. Ticktock: A non-goal-oriented multimodal dialog system with engagement awareness. In *AAAI Spring Symposia*.

HiJoNLP at SemEval-2022 Task 2: Detecting Idiomaticity of Multiword Expressions using Multilingual Pretrained Language Models

Minghuan Tan
School of Computing and Information Systems
Singapore Management University
mhtan.2017@phdcs.smu.edu.sg

Abstract

This paper describes an approach to detect idiomaticity only from the contextualized representation of a MWE over multilingual pretrained language models. Our experiments find that larger models are usually more effective in idiomaticity detection. However, using a higher layer of the model may not guarantee a better performance. In multilingual scenarios, the convergence of different languages are not consistent and rich-resource languages have big advantages over other languages.

1 Introduction

In the past several years, there have been breakthroughs in a variety of natural language processing tasks with the power of pretrained language models. These include but are not limit to question answering (Devlin et al., 2019), language generation (Radford et al., 2018, 2019) and machine translation (Liu et al., 2020). However, it's still not clear whether pretrained language models have the ability in capturing the meanings of multiword expressions (MWEs), especially idioms. Given the prevalent usage of idioms in different languages, identifying the correct meaning of a phrase in a certain context is crucial for many downstream tasks including sentiment analysis (Williams et al., 2015), automatic spelling correction (Horbach et al., 2016) and machine translation (Isabelle et al., 2017).

In literature, idiomaticity detection has been a research topic drawing much attention from the NLP community. MWEs which have both an idiomatic interpretation and a literal interpretation are also referred as Potentially Idiomatic Expressions (PIEs), for example, *spill the beans*. There has been both supervised (Sporleder and Li, 2009) and unsupervised (Haagsma et al., 2018; Kurfalı and Östling, 2020) approaches to solve this problem. For example, Feldman and Peng (2013) treated idiom recognition as outlier detection, which does not rely on costly annotated training data. Peng et al. (2014) incorporated the affective hypothesis of idioms to facilitate the identification of idiomatic operations.

Due to the limited understanding of how pretrained language models may handle representation of phrases, a series of works are proposed to investigate phrase composition from their contextualized representations. Yu and Ettinger (2020) conduct analysis of phrasal representations in state-of-the-art pre-trained transformers and find that phrase representation in these models still relies heavily on word content, showing little evidence of nuanced composition. Shwartz and Dagan (2019) confirm that contextualized word representations perform better than static word embeddings, more so on detecting meaning shift than in recovering implicit information. Therefore, it remains a challenging problem to resolve the idiomaticity of phrases.

Specifically on idiomaticity, recent approaches are trying to further diagnose pretrained language models using new metrics and datasets. Garcia et al. (2021a) analyse different levels of contextualisation to check to what extent models are able to detect idiomaticity at type and token level. Garcia et al. (2021b) propose probing measures to assess Noun Compound (NC) idiomaticity and conclude that idiomaticity is not yet accurately represented by contextualised models. AStitchInLanguageModels (Tayyar Madabushi et al., 2021) design two tasks to first test a language model's ability to detect idiom usage, and the effectiveness of a language model in generating representations of sentences containing idioms. Tan and Jiang (2021) conduct two probing tasks, PIE usage classification and idiom paraphrase identification, suggesting that BERT indeed is able to separate the literal and idiomatic usages of a PIE with high accuracy and is also able to encode the idiomatic meaning of a PIE to some extent. However, there's still much more to explore in idiomaticity.

Based upon AStitchInLanguageModels (Tayyar Madabushi et al., 2021), SemEval-2022

Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 190 - 196
July 14-15, 2022 ©2022 Association for Computational Linguistics

Task2 (Tayyar Madabushi et al., 2022) is proposed with a focus on multilingual idiomaticity. The task is arranged consisting the two subtasks:

1. Subtask A: A binary classification task aimed at determining whether a sentence contains an idiomatic expression.

2. Subtask B: Pretrain or finetune a model which is expected to output the correct Semantic Text Similarity (STS) scores between sentence pairs, whether or not either sentence contains an idiomatic expression.

In this paper, we focus on Subtask A and investigate how the span representation of a MWE can tell about its idiomaticity. We extend one of the monolingual idiomaticity probing method (Tan and Jiang, 2021) to multilingual scenario and compare multiple settings using multi-lingual BERT (mBERT) (Devlin et al., 2019) and XLM-R (Conneau et al., 2020). Following Yu and Ettinger (2020), we also consider variations of phrase representations across models, layers, and representation types. Different from them, we use more representation types to conduct the experiments.

Our main conclusion from these experiments are two folds:

1. Larger models are usually more effective in idiomaticity detection. However, a higher layer may not contribute more to the idiomaticity detection task, or more contextualization does not guarantee a better performance.

2. For multilingual scenario, the convergence of different languages are not consistent. Rich resource languages have initiative advantages over other languages.

2 System Overview

2.1 Subtask A

For Subtask A, to test models' ability to generalise, both zero-shot and one-shot settings are considered.

1. zero-shot: PIEs in the training set are completely disjoint from those in the test and development sets.

2. one-shot: one positive and one negative training examples for each MWE in the test and development sets

Note that the actual examples in the training data are different from those in the test and development sets in both settings.

Data Each row of the data of Subtask A has attributes like language and the potentially idiomatic MWE. The "Target" is the sentence that contains this MWE. The previous and next sentences for context are also provided. The label provides the annotation of that row, and a label of 0 indicates "Idiomatic" and a label of 1 indicates "non-idiomatic", including proper nouns.

Baseline The baseline model (Tayyar Madabushi et al., 2022) is based on mBERT. In the zero-shot setting, the model uses the context (the sentences preceding and succeeding the one containing the idioms) and does not add the idiom as an additional feature (in the "second input sentence"). In the one shot setting, the model is trained on both the zero-shot and one-shot data, but exclude the context (the sentences preceding and succeeding the one containing the idioms) and add the idiom as an additional feature in the "second sentence".

2.2 Span-based Model

While the common practice for classification tasks using pretrained language models usually needs concatenation of text sequences, this does not tell us enough information how representations of MWEs may lead to the change of performance. Therefore, in this work, we focus on the contextualized representations of MWEs to predict its idiomaticity.

Problem Formulation Consisting with the definition in (Tan and Jiang, 2021) , given a sentence denoted as $(w_1, w_2, \ldots, w_n)$, which contains a MWE with m words denoted as $(w_i, \ldots, w_{i+m-1})$, The task is to decide whether the MWE is used with its *literal* meaning or its *idiomatic* meaning, or if a sentence contains an idiomatic expression as describe in the task.

Span Identification In this work, our method requires a pair of span indices of the target MWE to extract their hidden representation from the encoded sequence. However, in this task, no such indices is offered explicitly from the dataset. We empirically find these indices by using editing distances in characters between the MWE and the sentence. This method works for most of the cases.

Span Representation For each MWE, we have a pair of span offsets in the original context. We use an L-layer BERT to process the tokenized context by prepending `[CLS]` to the beginning and append-

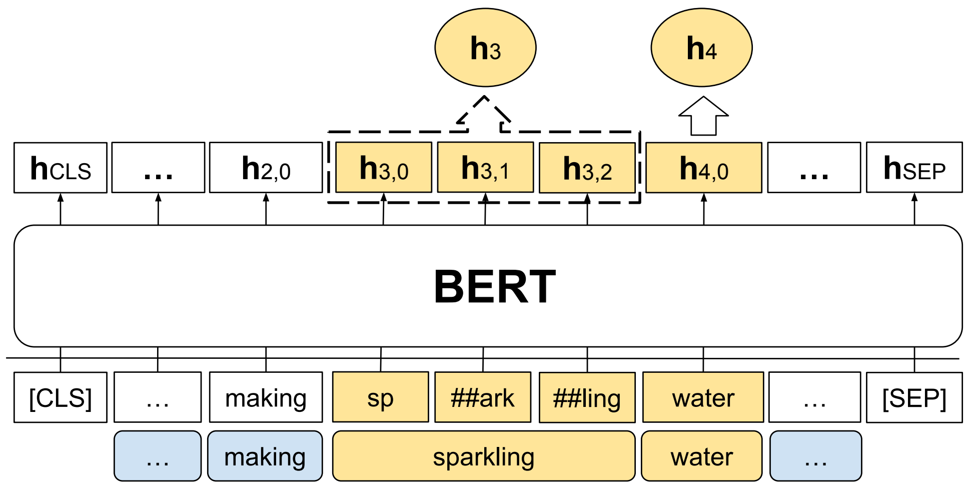


Figure 1: Mismatched transformer-based span representation.

ing `[SEP]` to the end. Let $\mathbf{h}_i^k \in \mathbb{R}^d$ denote the hidden vector produced by the kth layer of BERT representing w_i. We extract the hidden representations of the span to get its contextualized representations. For each MWE, we get a sequence of hidden vectors at the k-th layer for the m tokens inside this MWE as follows: $\mathbf{p}^k = (\mathbf{h}_i^k, \mathbf{h}_{i+1}^k, \ldots, \mathbf{h}_{i+m-1}^k)$.

In transformer-based models, a word might be tokenized into several pieces. We adopt the mismatched tokenization trick offered by Allennlp [1] to reconstruct its hidden vector. The hidden vector will be the average embeddings of constituent pieces. The mismatched encoding is illustrated in Figure 1.

We represent the target MWE using the span by six different kinds of combinations of the span's words. The first four of them are only using their endpoints. We use $\mathbf{x} = \mathbf{h}_i^k$ to denote the start of the span and $\mathbf{y} = \mathbf{h}_{i+m-1}^k$ to denote the end of the span.

1. **x,y** The span is represented by a direct concatenation of two endpoints.
2. **x,y,x-y** The span is represented by a direct concatenation of two endpoints and the difference of them.
3. **x,y,x*y** The span is represented by a direct concatenation of two endpoints and the elementwise product of them.
4. **x,y,x*y,x-y** The span is represented by a direct concatenation of two endpoints, the elementwise product and the difference of them.
5. **SelfAttentive** We firstly compute an unnormalized attention score for each word in the document. Then we compute spans representations with respect to these scores by normalising the attention scores for words inside the span.
6. **MaxPooling** A span is represented through a dimension-wise max-pooling operation. Given a span, the resulting value of a dimension is using the maximum value of this dimension across all the span tokens.

Span Classification We use a binary linear classifier upon the span representation.

3 Experiments

In this paper, we want to test how the pretrained model, the transformer layer and the representation type, affect performance of idiomaticity detection.

3.1 Settings

This subtask is evaluated using the Macro F1 score between the gold labels and model predictions (see the details in the evaluation script).

All the multilingual pretrained langauge models are hold by Huggingface, including mBERT[2], XLM-R[3] and XLM-R-L[4].

Since we are focusing on comparison of span representation across different layers and representation types, we conduct experiments with the 4-th,

[1] https://github.com/allenai/allennlp

[2] BERT multilingual base (cased) : https://huggingface.co/bert-base-multilingual-cased

[3] XLM-RoBERTa (base-sized model): https://huggingface.co/xlm-roberta-base

[4] XLM-RoBERTa (large-sized model) : https://huggingface.co/xlm-roberta-large

Model	Type	Layer	EN	PT	GL	Avg
mBERT (Tayyar Madabushi et al., 2022)	-	12	70.70	68.03	50.65	65.40
mBERT	x,y,x-y	12	76.24	72.27	64.27	72.85
XLM-R	x,y	8	**77.62**	71.61	64.88	72.68
XLM-R-L	x,y,x-y	24	75.22	**75.80**	**69.01**	**74.66**

Table 1: Experiment results of zero-shot setting for different multilingual pretrained models, in macro F1 score.

Model	Type	Layer	EN	PT	GL	Avg
mBERT (Tayyar Madabushi et al., 2022)	-	12	88.62	86.37	81.62	86.46
mBERT	MaxPooling	8	86.59	85.82	85.77	86.63
XLM-R	MaxPooling	8	89.49	83.71	82.19	86.17
XLM-R-L	x,y,x*y,x-y	24	**91.26**	**86.96**	**89.06**	**89.79**

Table 2: Experiment results of one-shot setting for different multilingual pretrained models, in macro F1 score.

8-th and 12-th layer of mBERT and XLM-R and the 8-th, 12-th and 24-th layer of XLM-R-L. All six representation types are considered for each layer-based models.

We run most of our experiments with an NVIDIA 1080ti GPU with 11GB memory, and use a NVIDIA A100 for XLM-R-L-based experiments. We finetune each experiment for 10 epochs with the learning rate set to 5e-5. We notice that the training process converges with training accuracy 1 in a short period. To reduce the effect of overfitting, we use a dropout probability of 0.5 before the classification layer. Our code is built over Allennlp2 and will be released on Github[5].

3.2 Results and Analyses for Subtask A

We list the overall experiment results in Table 3 in the Appendix. The table contains three main parts with each part showing the detailed experiment results for a multilingual pretrained language model. In each part, we test all six combinations of span representations using encoded sequences from different layers. To better illustrate our major conclusions, we select the best settings for each multilingual model from Table 3, and rearrange the zero-shot results to Table 1 and one-shot results to Table 2.

Table 1 shows us that using only endpoints of the span can be effective in predicting its idiomaticity and representation type **x,y,x-y** is a good choice for the zero-shot setting. We think representation using only endpoints is working well might due to most of the MWEs in current dataset consist of two words.

Table 2 shows us that representation type **MaxPooling** is a good choice for the one-shot setting and the best performance may be achieved using middle layers.

Combining both zero-shot setting and one-shot setting, we find that larger models are usually more effective in idiomaticity detection. For a specific pretrained model, using contextualized representation from a higher layer may not guarantee a better performance. For example, from the perspective of overall score for the One Shot scenario, the highest scores are all reached at the 8-th layer. However, we didn't observe a consistent advantage of using a specific representation type across different models and layers.

From the perspective of language, span-based models are achieving relative larger gains in both settings for GL. On one hand, the corpus used for training pretrained language models is not balanced across different languages. For example, in XLM-R, data from EN is several times than that of PT and hundrands times than that of GL. The data for GL may just surpass a minimal size for learning a BERT model and restricts performance in both settings for GL compared with PT and EN. On the other hand, this tells us that better span representation still help in detection of idiomaticity.

3.3 Endpoints-based Representation

This work focuses on the contextualized representation of the span of a target MWE. As pointed out by

[5]https://github.com/VisualJoyce/CiYi

others, phrase representations, especially idioms, are not always compositional and rely more than the constituent words in the span. Not to mention, it is a much easier case which only uses the endpoints of the span. However, in both zero-shot setting and one-shot setting, we notice that endpoints-based methods works almost as well. We suspect this may due to the following reasons: (1) Endpoints of MWEs are highly correlated with these MWEs and can be very indicative about their representation. (2) Most of the MWEs covered in this dataset contain two words.

4 Conclusion

In conclusion, our experiments find that larger models are usually more effective in idiomaticity detection. And for a specific pretrained model, using contetualized representation from a higher layer may not guarantee a better performance. As the data used for multilingual pretrained language models is not well-balanced, rich resource languages have significant advantages over other languages. In the future, with the community contributing stronger language models with more balanced language distribution and more multilingual idiom-annotated datasets, idiomaticity detection still has large potentials to be explored from more angles.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Anna Feldman and Jing Peng. 2013. Automatic detection of idiomatic clauses. In *Computational Linguistics and Intelligent Text Processing*, pages 435–446, Berlin, Heidelberg. Springer Berlin Heidelberg.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021a. Assessing the representations of idiomaticity in vector models with a noun compound dataset labeled at type and token levels. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2730–2741, Online. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021b. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Hessel Haagsma, Malvina Nissim, and Johan Bos. 2018. The other side of the coin: Unsupervised disambiguation of potentially idiomatic expressions by contrasting senses. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 178–184, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Andrea Horbach, Andrea Hensler, Sabine Krome, Jakob Prange, Werner Scholze-Stubenrecht, Diana Steffen, Stefan Thater, Christian Wellner, and Manfred Pinkal. 2016. A corpus of literal and idiomatic uses of German infinitive-verb compounds. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 836–841, Portorož, Slovenia. European Language Resources Association (ELRA).

Pierre Isabelle, Colin Cherry, and George Foster. 2017. A challenge set approach to evaluating machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2486–2496, Copenhagen, Denmark. Association for Computational Linguistics.

Murathan Kurfalı and Robert Östling. 2020. Disambiguation of potentially idiomatic expressions with contextual embeddings. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 85–94, online. Association for Computational Linguistics.

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

Jing Peng, Anna Feldman, and Ekaterina Vylomova. 2014. Classifying idiomatic and literal expressions using topic models and intensity of emotions. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2019–2027, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Caroline Sporleder and Linlin Li. 2009. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 754–762, Athens, Greece. Association for Computational Linguistics.

Minghuan Tan and Jing Jiang. 2021. Does BERT understand idioms? a probing-based empirical study of BERT encodings of idioms. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1397–1407, Held Online. INCOMA Ltd.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lowri Williams, Christian Bannister, Michael Arribas-Ayllon, Alun Preece, and Irena Spasić. 2015. The role of idioms in sentiment analysis. *Expert Systems with Applications*, 42(21):7375 – 7385.

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

Model	Type	Layer	Zero Shot				One Shot			
			EN	PT	GL	Avg	EN	PT	GL	Avg
mBERT	-	12	70.70	68.03	50.65	65.40	88.62	86.37	81.62	86.46
mBERT	x,y	4	75.11	69.63	64.20	72.49	86.32	85.17	76.50	83.84
mBERT	x,y,x-y	4	73.69	71.69	57.96	70.31	86.51	85.68	77.04	84.25
mBERT	x,y,x*y	4	76.76	70.67	60.27	71.69	87.76	86.15	80.16	85.93
mBERT	x,y,x*y,x-y	4	75.54	73.56	60.18	71.62	89.28	85.16	80.21	86.17
mBERT	SelfAttentive	4	72.13	73.19	62.16	70.79	85.48	82.86	78.76	83.50
mBERT	MaxPooling	4	71.27	73.11	58.46	69.49	85.23	83.40	76.56	82.93
mBERT	x,y	8	75.95	68.49	65.07	72.21	85.87	84.91	81.21	84.97
mBERT	x,y,x-y	8	72.45	66.88	61.95	69.11	86.86	83.95	82.47	85.41
mBERT	x,y,x*y	8	75.14	67.73	61.81	70.49	86.55	81.82	81.29	84.23
mBERT	x,y,x*y,x-y	8	72.59	73.18	61.91	70.87	86.09	84.21	81.30	84.79
mBERT	SelfAttentive	8	73.84	68.60	62.22	69.78	89.69	82.72	83.65	86.46
mBERT	MaxPooling	8	77.24	68.59	62.16	71.89	86.59	85.82	85.77	86.63
mBERT	x,y	12	76.31	70.77	58.80	70.36	86.45	84.06	79.69	84.47
mBERT	x,y,x-y	12	76.24	72.27	64.27	72.85	85.91	85.19	82.60	85.47
mBERT	x,y,x*y	12	74.04	71.76	64.24	71.65	87.58	84.27	79.92	85.00
mBERT	x,y,x*y,x-y	12	78.63	69.01	62.91	72.62	86.66	85.24	79.05	84.75
mBERT	SelfAttentive	12	75.03	69.71	60.75	70.32	86.31	82.62	83.69	85.14
mBERT	MaxPooling	12	75.33	71.08	59.00	69.90	88.37	85.38	81.19	86.07
XLM-R	x,y	4	80.70	65.29	54.57	68.82	89.26	80.22	73.15	82.48
XLM-R	x,y,x-y	4	79.49	67.51	54.98	69.20	89.27	82.26	74.10	83.47
XLM-R	x,y,x*y	4	78.66	70.77	57.66	71.19	88.38	79.47	70.03	80.79
XLM-R	x,y,x*y,x-y	4	74.10	67.11	56.48	68.49	88.30	81.13	73.96	82.73
XLM-R	SelfAttentive	4	**81.51**	70.99	55.49	71.91	88.46	80.84	74.82	82.78
XLM-R	MaxPooling	4	78.57	67.48	59.38	70.69	88.97	81.83	79.99	84.81
XLM-R	x,y	8	77.62	71.61	64.88	72.68	89.18	80.98	78.43	84.22
XLM-R	x,y,x-y	8	76.86	66.31	60.52	69.38	86.57	81.33	72.91	81.51
XLM-R	x,y,x*y	8	73.51	62.41	55.22	65.02	87.58	81.31	75.57	82.73
XLM-R	x,y,x*y,x-y	8	77.70	70.43	65.19	72.43	87.74	78.24	80.44	83.36
XLM-R	SelfAttentive	8	78.43	68.01	61.40	71.08	89.03	83.19	75.55	83.82
XLM-R	MaxPooling	8	76.61	68.45	64.50	71.35	89.49	83.71	82.19	86.17
XLM-R	x,y	12	76.95	66.35	57.73	68.54	86.66	81.73	77.73	83.15
XLM-R	x,y,x-y	12	75.98	63.26	55.31	66.31	86.12	79.82	73.51	80.92
XLM-R	x,y,x*y	12	77.70	70.18	61.05	71.05	87.65	79.54	74.17	81.83
XLM-R	x,y,x*y,x-y	12	78.07	71.51	59.04	70.91	88.19	82.09	76.30	83.45
XLM-R	SelfAttentive	12	76.16	70.54	62.92	71.36	90.05	79.77	77.26	83.82
XLM-R	MaxPooling	12	74.98	75.23	63.80	72.31	85.67	81.03	75.50	81.88
XLM-R-L	x,y	8	79.10	72.78	61.68	72.82	91.89	85.56	75.63	85.86
XLM-R-L	x,y,x-y	8	76.96	59.09	57.83	68.44	89.08	85.94	76.44	85.25
XLM-R-L	x,y,x*y	8	73.51	62.41	55.22	65.02	87.58	81.31	75.57	82.73
XLM-R-L	x,y,x*y,x-y	8	80.19	71.09	62.12	73.45	91.92	81.79	72.77	84.06
XLM-R-L	SelfAttentive	8	77.25	71.92	59.94	70.56	**92.66**	84.59	77.57	86.37
XLM-R-L	MaxPooling	8	77.83	70.92	61.18	71.40	89.85	81.79	69.14	81.71
XLM-R-L	x,y	12	76.92	70.40	60.11	70.17	90.26	85.19	82.76	87.10
XLM-R-L	x,y,x-y	12	77.48	67.52	60.25	69.85	92.24	81.30	81.00	86.30
XLM-R-L	x,y,x*y	12	80.54	65.49	55.46	68.72	91.43	83.91	78.78	86.00
XLM-R-L	x,y,x*y,x-y	12	79.77	69.66	60.84	71.54	90.28	84.42	83.46	87.14
XLM-R-L	SelfAttentive	12	78.13	74.44	61.92	72.63	90.48	86.23	78.90	86.43
XLM-R-L	MaxPooling	12	80.68	71.01	62.90	73.06	92.46	86.03	77.26	86.62
XLM-R-L	x,y	24	78.55	74.83	65.72	74.46	90.15	85.58	85.98	88.10
XLM-R-L	x,y,x-y	24	75.22	**75.80**	**69.01**	**74.66**	90.27	85.40	85.50	87.94
XLM-R-L	x,y,x*y	24	80.55	67.54	63.38	73.08	87.66	81.48	79.72	84.08
XLM-R-L	x,y,x*y,x-y	24	76.63	73.76	64.52	72.65	91.26	86.96	**89.06**	**89.79**
XLM-R-L	SelfAttentive	24	73.17	71.93	62.14	69.99	88.64	**87.81**	80.86	86.73
XLM-R-L	MaxPooling	24	75.39	72.33	66.15	72.26	89.30	85.47	85.39	87.55

Table 3: Experiment results for different multilingual pretrained models, in macro F1 score. We use bold font to highlight the maximum score across all settings and underline to highlight the maximum score in each part.

ZhichunRoad at SemEval-2022 Task 2: Adversarial Training and Contrastive Learning for Multiword Representations

Xuange Cui , Wei Xiong , Songlin Wang
JD.com, Beijing, China
{cuixuange,xiongwei9,wangsonglin3}@jd.com

Abstract

This paper presents our contribution to the SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. We explore the impact of three different pre-trained multilingual language models in the SubTaskA. By enhancing the model generalization and robustness, we use the exponential moving average (EMA) method and the adversarial attack strategy. In SubTaskB, we add an effective cross-attention module for modeling the relationships of two sentences. We jointly train the model with a contrastive learning objective and employ a momentum contrast to enlarge the number of negative pairs. Additionally, we use the alignment and uniformity properties to measure the quality of sentence embeddings. Our approach obtained competitive results in both subtasks.

1 Introduction

In recent years, the pre-trained models have been widely used and play a vital role in the natural language processing tasks. The success of language models relies on huge amounts of unlabeled data and the useful representation layers that are designed to draw on information from the surrounding context (Devlin et al., 2018;Nedumpozhimana and Kelleher, 2021). However, more recent studies show that even state-of-the-art pre-trained contextual models (e.g. BERT) can't accurately represent idiomatic expressions (Yu and Ettinger, 2020;Garcia et al., 2021). One reason for this is that many expressions can be used both literally and idiomatically.

Specifically, the size of vocabulary can't increase indefinitely, which makes representing idiomatic phrases particularly challenging (Shwartz, 2021;Tayyar Madabushi et al., 2021). Idioms occur in almost all languages, to distinguish whether an expression has an idiomatic sense would leverage both cross-lingual models and multiword expressions (MWEs). The SemEval 2022 Task 2(Tayyar Madabushi et al., 2022) is aimed at detecting and representing MWEs and presents a novel multilingual dataset across English, Portuguese and Galician. And this task consists of two different subtasks to evaluate the model's ability to identify and capture idiomaticity.

SubTask	Train	Dev	Test	Desc
zero-shot	4492	740	763	binary classification
one-shot	4492	740	763	
pre-train	24498	2000	3827	semantic similarity
fine-tune	6573	2182	2263	

Table 1: The statistics of datasets.

Our contributions can be summarized as follows: 1) We choose three transformer-based language models from the XTREME LeaderBoard (Hu et al., 2020), and compare the effectiveness of mBERT$_{base}$ (Devlin et al., 2018), XLM-R$_{base}$(Conneau et al., 2020), and InfoXLM$_{base}$(Chi et al., 2021). 2) We adopt the exponential moving average method (EMA) and adversarial training strategy to improve the model's generalization and robustness. We achieved considerably performance gain of 2.91%, 3.87% over the baseline solution, and ranked 12th in the zero-shot settings and 4th in the one-shot settings. 3) With finetuning on the supervised target datasets, we use cross-attention module and jointly train the model with an extra contrastive loss layer on top of the BERT encoder. Our approach achieved an 8.22%, 4.5% improvement compared to the baseline solution, and ranked top-4 in the pre-train and fine-tune settings. We release the source code and pre-trained models associated with this work. [1]

Moreover, we find that adversarial training can

[1] https://github.com/cuixuage/SemEval2022-Task2

achieve good performance by setting the appropriate batch size. In the SubTaskB, we show that joint training regularizes the sentence embeddings' anisotropic space to be more uniform but also suffers a degeneration in alignment slightly. The trade-off between the alignment and uniformity (Wang and Isola, 2020) indicates that perfect alignment and perfect uniformity are likely hard to simultaneously achieve in practice.

2 Background

SemEval-2022 Task 2(Tayyar Madabushi et al., 2022) provides two subtasks. Subtask A consists of a binary classification task aimed at determining whether a sentence contains an idiomatic expression. The sample of the dataset consists of the previous sentence, target sentence, next sentence, and MWE. The target sentence contains the potentially idiomatic MWE, and the label of 0 indicates "Idiomatic" and the label of 1 indicates "non-idiomatic". Our model receives the context sentences as input in the zero-shot setting, and receives the target sentence by adding the MWE as an additional feature in the one-shot setting. This is based on the results presented in the dataset paper (Tayyar Madabushi et al., 2021).

SubtaskB consists of a novel task which requires the model to output the correct Semantic Text Similarity (STS) scores. The task is designed to test a model's ability to generate sentence embeddings that accurately represent sentences regardless of whether or not they contain idiomatic expressions. When evaluating the trained model, we first obtain the sentence embeddings, then we calculate the Spearman correlation between the cosine similarity scores of sentence embeddings and the gold labels. The statistics of the corpus are shown in Table 1. Our team participated in both subtasks, and the next section will introduce an overview of our system.

3 System Overview

We focus on comparing the impact of different training techniques adopted in our system. In this section, we first present the BERT-like text encoder, then we introduce several strategies for improving models' robustness. Finally, we talk about the design of the cross-attention module and the jointly training way of incorporating supervised signals and unsupervised signals.

3.1 Transformer-based Models

In the zero-shot and one-shot settings, we compare several pre-trained multilingual language models from the XTREME Leaderboard[2] as the text encoder . The models shown below are also available on the hugging-face website[3].

mBert$_{base}$,the bert-base-multilingual-cased model is pre-trained on the top 104 languages with the Wikipedia dataset, and consists of 12-layer, 768-hidden, 12-heads, 109M parameters and a shared vocabulary size of 110000 (Devlin et al., 2018).

XLM-R$_{base}$,the xlm-roberta-base model consists of 100 languages and pre-trained with filtered CommonCrawl dataset, and consists of 12-layer, 768-hidden, 12-heads, $\tilde{2}$70M parameters and a shared vocabulary size of 250002 (Conneau et al., 2020).

InfoXLM$_{base}$,we use the "microsoft/infoxlm-base" model containing 94 languages and pre-trained with CCNet dataset, and has the same configurations of XLM-R and a shared vocabulary size of 250002 (Chi et al., 2021).

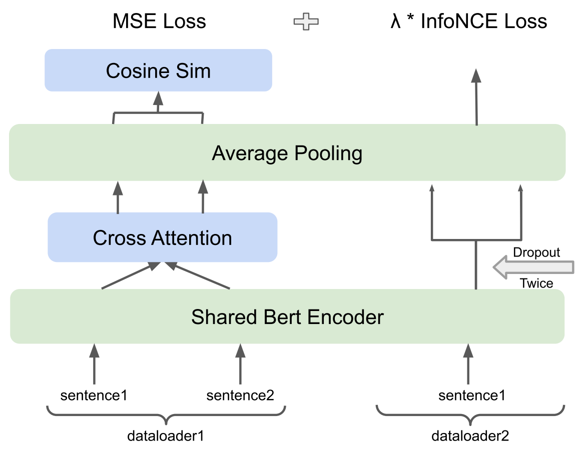


Figure 1: Incorporating supervised and unsupervised signals. MSE Loss: the mean squared error, InfoNCE Loss: the contrastive objective.

3.2 Training Procedures

There are two ways of enhancing the model generalization and robustness.

Exponential Moving Average Our model uses EMA to smooth the trained parameters. Evaluations that use averaged parameters sometimes produce significantly better results than the final trained values. Formally, we define the smoothed

[2]https://sites.research.google/xtreme
[3]https://huggingface.co/models

SubTaskB	Model	ALL Data	Idiom Data	STS Data
pre-train	mBert$_{base}$	53.90	21.87	80.82
	mBert$^{\diamondsuit}_{base}$	55.58	27.18	82.09
	mBert$^{\clubsuit}_{base}$	56.32	28.26	83.59
fine-tune	mBert$_{base}$	62.29	34.59	52.29
	mBert$^{\diamondsuit}_{base}$	63.16	36.95	53.49
	mBert$^{\clubsuit}_{base}$	64.01	39.56	56.15

Table 2: Performance of Our Approach on the Sentence Representation Task. We report the Spearman correlation $\times$ 100 on the test sets, $\diamondsuit$: jointly train the model with the contrastive objective, $\clubsuit$: jointly train the model with the cross-attention module and the contrastive objective.

variables and trained variables as θ_s and θ_t, EMA decay weight as: η. After each training step, we update θ_s by:

$$\theta_s \leftarrow \eta\theta_s + (1-\eta)\theta_t \tag{1}$$

Adversarial Training Recently, adversarial attack has been widely applied in computer vision and natural language processing (Yan et al., 2021). Many works use it during fine-tuning, because computing adversarial perturbations relies on supervised signals. We explore the influence of adversarial training strategies with different batch size, and compare the FGSM (Goodfellow et al., 2015), PGD (Madry et al., 2019), FREELB (Zhu et al., 2020) and SMART (Jiang et al., 2020) methods in the zero-shot and one-shot settings. It works by augmenting the input with a small perturbation that maximizes the adversarial loss:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\Delta x\in\Omega} L(x+\Delta x, y; \theta)\right] \tag{2}$$

where the $\mathcal{D}$ is dataset, x is input, y is the gold label, θ is the model parameters, $L(x, y; \theta)$ is the loss function and Δx is the perturbation. In our experiments, we adopt SMART method in zero-shot setting, and FREELB method in one-shot setting. These choices are based on actual performance.

3.3 Sentence Representation

Reimers and Gurevych (2019) propose a siamese architecture with a shared BERT encoder to compute the sentence representations for each input text. By making use of unlabeled texts, SimCSE (Gao et al., 2021) proposes an unsupervised contrastive learning method to alleviate the collapse issue of BERT. Compared to unsupervised SimCSE, we use extra supervised signals during training. Our approach is mainly inspired by ConSERT (Yan et al., 2021) and EsimCSE (Wu et al., 2021). As shown in Figure 1, there are two major objectives and an extra cross-attention module to exchange information with the token-wise embeddings.

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{mse}} + \lambda\mathcal{L}_{\text{con}} \tag{3}$$

the λ is a hyperparameter to balance two objectives. $\mathcal{L}_{\text{mse}}$ is Mean Squared Error, $\mathcal{L}_{\text{con}}$ is Contrastive Loss.

During training, each data point is trained to find out its counterpart among $(N-1)$ from in-batch negative samples and the queue of data samples. The samples in the queue are progressively replaced (He et al., 2020).

$$-\log \frac{e^{\text{sim}\left(\mathbf{h}_i,\mathbf{h}_i^{+}\right)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}\left(\mathbf{h}_i,\mathbf{h}_j^{+}\right)/\tau} + \sum_{q=1}^{Q} e^{\text{sim}\left(\mathbf{h}_i,\mathbf{h}_q^{+}\right)/\tau}} \tag{4}$$

The h_* is the sentence representation, where h_i and h_i^+ are semantically related. The h_q^+ is denotes a sentence embedding in the momentum-updated queue. And the Q is the size of the queue, $sim(h1, h2)$ is the cosine similarity scores of sentence representations, τ is a temperature hyperparameter. In the end, we average the all N Li losses to calculate the contrastive loss $\mathcal{C}_{\text{con}}$.

4 Experiments

4.1 Settings

We use InfoXLM$_{base}$ (Chi et al., 2021) as the text encoder, the EMA decay weight is set to 0.999, the learning rate is set to 2e-5 with warmup ratio over 10% in the SubTaskA. We compare the impact of batch size $\in$ 16, 32, 64 with different adversarial training strategies. By default, We set ε to 1.0 in FGM, set K steps to 3 in PGD, FREELB and SMART. That means calculate 3 steps in the adversarial attack. We adopt λ as 0.5 and μ as 0.2 to smooth the logits and embeddings in the SMART method (Jiang et al., 2020). We use SMART method and batches of size 16 in zero-shot setting, FREELB method and batches of size 32 in one-shot setting.

SubTask	Model	Practice	Post-Eval
zero-shot	mBert$_{base}$	68.71	-
	infoxlm$^{\diamondsuit}_{base}$	76.21	68.31
one-shot	mBert$_{base}$	84.77	-
	infoxlm$^{\clubsuit}_{base}$	94.07	90.33

Table 3: Performance of Our Approach on the Idiomaticity Detection Task. We report the F1 Score × 100 on the dev and test sets, $\diamondsuit$: set bath size to 16 and use SMART, $\clubsuit$: set bath size to 32 and use FREELB.

Method	Practice	Post-Eval
mBert$_{base}$	68.71	-
InfoXLM$_{base}$	73.10	65.2
+EMA	75.75	67.85
+EMA+SMART	76.21	68.31

Table 4: The effect of different strategies and keep accumulating from top to bottom. We report the dev-F1 Score × 100 in zero-shot setting.

In the SubTaskB, we use mBERT$_{base}$ (Devlin et al., 2018) as the text encoder, set batch size to 32 and set warmup ratio to 10%. During the jointly training, λ is set to 0.15 and τ is set to 0.05 that used in the $\mathcal{L}_{\text{con}}$. We use the dev set of STS-B and ASSIN2 to tune the hyperparameter and evaluate the model every 250 steps during training. The best checkpoint is saved for testing, we further discuss the results of our experiments in the subsequent section.

4.2 Main Results

Our submitted results were evaluated on F1 Score in SubTaskA, and Spearman correlation in SubTaskB. We jointly train the model with contrastive objective and the supervised signals on the Semantic Text Similarity dataset, including STSBenchmark and ASSIN2 datasets. We compare several models as the text encoder and different training methods, as described in Section 3. The main results shown in Table 2 and Table 3. As shown in Table 3, we achieve a performance gain of 2.91%, 3.87% over the baseline solution by finetuning the InfoXLM$_{base}$ model with using EMA method and adversarial training. In the Table 2, our approach achieves a 8.22%, 4.5% improvement compared to the baseline solution that indicates the usefulness of the cross-attention module and jointly training way. In the next section, we study the effect of different strategies.

5 Ablation Studies

5.1 Effect of Pre-trained Models

We investigate the impact of adopting different multi-lingual models in the zero-shot setting. In Figure 2, we show the results of different language models fine-tuning in 50 epochs. We find that the best f1 score on validation dataset is provided by InfoXLM$_{base}$ (Chi et al., 2021).

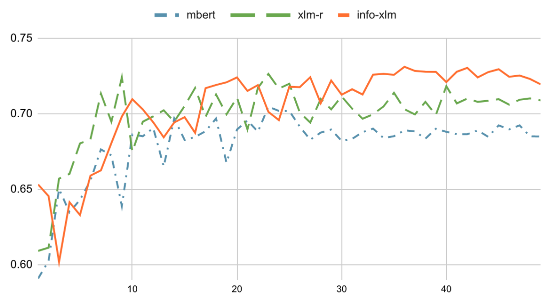


Figure 2: The fine-tuning of multi-lingual language models. We report the dev-F1 Score in zero-shot setting.

5.2 Effect of Training Techniques

As shown in Figure 3, we set bath size to 16 and use 0.999 as the EMA decay weight to obtain the best score. In the zero-shot and one-shot settings, we find that the performance is extremely sensitive to the batch size. And with the benefit of smoothing performance, using the EMA method can improve the model robustness when evaluating the trained model.

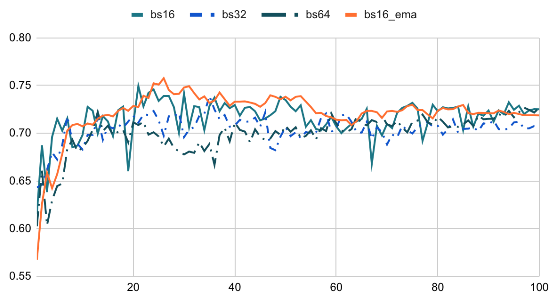


Figure 3: The batch size with EMA method. We report the dev-F1 Score in zero-shot setting.

The experimental results of adversarial training are presented in Figure 4. We set the size of mini-batch to 16 and use SMART in the zero-shot setting, and the batch size is set to 32 and use FREELB as adversarial attack in one-shot setting. We observe that the SMART and FREELB strategies have better performance than FGM and PGD strategies.

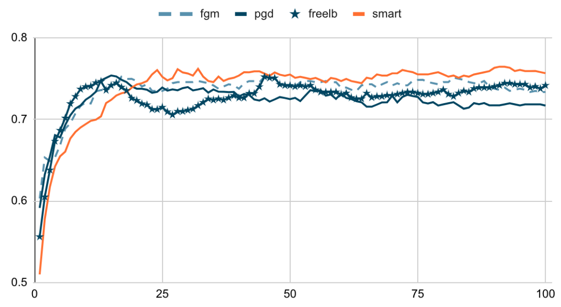


Figure 4: The performance of different adversarial attack strategies. We report the dev-F1 Score in zero-shot setting.

As presented in Table 5, we explore the impact of InfoXLM$_{base}$ model, smaller batch size, EMA method and adversarial training. These strategies can effectively improve the performance of our approach.

5.3 Effect of Contrastive Learning

In this section, we investigate the contrastive learning how to further improve the performance of sentence representations. As shown in Table 5, we use contrastive learning as unsupervised signals which yields a substantial improvement on the STS-Test dataset. We also use an extra cross-attention layer to achieve a 0.8%, 1.2% improvement in the pre-train and fine-tune settings. The cross attention idea is inspired by Reimers and Gurevych (2019), the paper shows that the Cross-Encoder achieves better performances than Bi-Encoders.

Method	Practice	Post-Eval
mBert$_{base}$	70.33	-
+CrossAttention	70.96	55.94
+ + InfoNCE	71.11	56.09
+ + + MoCo	71.34	56.32

Table 5: The effect of different strategies on the STSTest dataset. We report the Spearman correlation × 100 in pre-train setting.

In general, models which have both better alignment and uniformity obtain better sentence representations, confirming the findings in Wang and Isola (2020). We also evaluate these metrics to measure the quality of learned embeddings, including alignment of the positive pairs and uniformity of the whole representation space. We calculate uniformity on the STS-B and ASSIN2 datasets, and alignment from the positive pairs that have the gold label more than or equal to the number 4.

As shown in Figure 5, we also find that: 1) Though pre-trained embeddings have good alignment, their uniformity is poor, e.g. mBert$_{base}$. 2) Unsupervised SimCSE$_{base}$ (Gao et al., 2021) has better uniformity of pre-trained embeddings than mBert$_{base}$. 3) Jointly training regularizes the sentence embeddings' anisotropic space to be more uniform than others, but also suffers a degeneration in alignment slightly. 4) The trade off between the alignment and uniformity indicates that perfect alignment and perfect uniformity are likely hard to simultaneously achieve in practice.

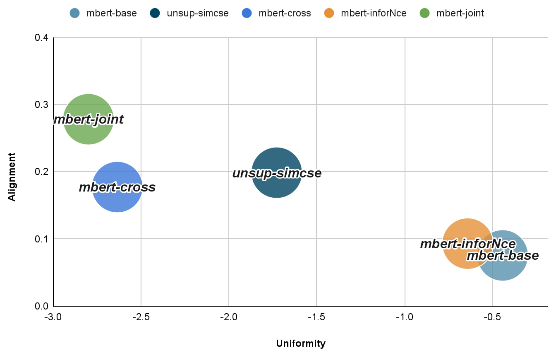


Figure 5: The alignment and uniformity of different pre-trained models. The closer to the origin of the coordinate axis, the better sentence representations.

6 Conclusion and Future Work

In this work, we provide an overview of the combined approach to detect and represent multiword expressions. We use InfoXLM$_{base}$ model as the text encoder and enhance the model generalization and robustness with exponential moving average (EMA) method and the adversarial attack strategy in the SubTaskA. In the SubTaskB, experimental results show that the cross-attention module and the contrastive learning task can considerably improve the performance. Finally, we analyze the alignment and uniformity properties to measure the quality of sentence embeddings. Future work of our system includes: 1) Using the larger pre-trained language models, such as mBert$_{large}$, InfoXLM$_{large}$. 2) Adopting other data augmentation, including Token-Shuffle, Token-Cutoff and Mix-Up.

References

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2019. Towards deep learning models resistant to adversarial attacks.

Vasudevan Nedumpozhimana and John Kelleher. 2021. Finding BERT's idiomatic key. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 57–62, Online. Association for Computational Linguistics.

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

Vered Shwartz. 2021. A long hard look at MWEs in the age of language models. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, page 1, Online. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere.

Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2021. Esimcse: Enhanced sample building method for contrastive learning of unsupervised sentence embedding.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075, Online. Association for Computational Linguistics.

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

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*.

NER4ID at SemEval-2022 Task 2: Named Entity Recognition for Idiomaticity Detection

Simone Tedeschi
Babelscape & Sapienza University of Rome
tedeschi@babelscape.com

Roberto Navigli
Sapienza University of Rome
navigli@diag.uniroma1.it

Abstract

Idioms are lexically-complex phrases whose meaning cannot be derived by compositionally interpreting their components. Although the automatic identification and understanding of idioms is essential for a wide range of Natural Language Understanding tasks, they are still largely under-investigated. This motivated the organization of the SemEval-2022 Task 2, which is divided into two multilingual subtasks: one about idiomaticity detection, and the other about sentence embeddings. In this work, we focus on the first subtask and propose a Transformer-based dual-encoder architecture to compute the semantic similarity between a potentially-idiomatic expression and its context and, based on this, predict idiomaticity. Then, we show how and to what extent Named Entity Recognition can be exploited to reduce the degree of confusion of idiom identification systems and, therefore, improve performance. Our model achieves 92.1 F_1 in the one-shot setting and shows strong robustness towards unseen idioms achieving 77.4 F_1 in the zero-shot setting. We release our code at https://github.com/Babelscape/ner4id.

1 Introduction

One of the main challenges in Natural Language Processing (NLP) is to embed the meaning of a piece of raw text (e.g. a word or a sentence) in a low-dimensional dense vector. With the advent of pretrained language models, which exploit contextual information and assume compositionality of word representations, significant improvements have been made in this direction (Peters et al., 2018; Devlin et al., 2019). On the other hand, very little attention has been paid to idiomatic expressions, i.e. multi-word expressions (MWEs) with an established meaning unrelated to the meanings of the individual constituents. However, since idiomaticity is a frequent phenomenon that can be observed in all languages, idiomatic expressions should play an important role in NLP. Indeed, their identification and understanding is crucial not only for Natural Language Understanding tasks such as Word Sense Disambiguation (Bevilacqua et al., 2021b), Semantic Role Labeling (Conia et al., 2021) and Semantic Parsing (Bevilacqua et al., 2021a), but also for Machine Translation (Edunov et al., 2018; Liu et al., 2020), Question Answering (Mishra and Jain, 2016) and Text Summarization (Chu and Wang, 2018), *inter alia*.

In the SemEval-2022 Task 2: *Multilingual Idiomaticity Detection and Sentence Embedding* (Tayyar Madabushi et al., 2022), research on idioms has been promoted by adapting datasets and tasks from the work carried out by Tayyar Madabushi et al. (2021). Specifically, the organizers propose two subtasks:

- **Subtask A**: a binary classification task in which potentially-idiomatic expressions (PIEs) must be labeled as either "Idiomatic" or "Literal", based on the context they appear in. To better test models' generalization capabilities, two different settings are provided: *zero-shot* and *one-shot*;

- **Subtask B**: requires models to output the correct Semantic Text Similarity (STS) scores between sentence pairs based on whether or not each sentence contains an idiomatic expression. Subtask B is also available in two settings: *pre-train* and *fine-tune*.

Both subtasks cover three languages: English, Portuguese and Galician[1]. In addition, the organizers provide strong baseline systems to compare with.

In this paper, we present the NER4ID submission to the SemEval-2022 Task 2 which focuses on Subtask A. Specifically, we successfully tackle

[1]Galician is included only in the test sets to test transfer-learning abilities of the models.

the idiom identification task by introducing a two-step system that: i) uses Named Entity Recognition (NER) to pre-identify non-idiomatic expressions, and ii) exploits a novel Transformer-based dual-encoder architecture to compute the semantic similarities between the remaining potentially-idiomatic expressions and their contexts and, based on these, predict idiomaticity. Finally, we extensively evaluate our system on both one-shot and zero-shot settings. We release our code at `https://github.com/Babelscape/ner4id`.

2 Related Work

Approaches to idiom identification were initially built on the notion that idiomatic expressions, like other MWEs, are less syntactically and lexically flexible than non-idiomatic and compositional ones. Indeed, initial studies focused on specific syntactic constructions. Fazly and Stevenson (2006) focused on verb/noun idioms, e.g. *shoot the breeze*, and used the Pointwise Mutual Information (PMI, Church et al., 1991) measure to quantify the degree of lexical, syntactic, and overall fixedness of a given verb+noun combination. Cook et al. (2007) and Diab and Bhutada (2009) also focused on verb/noun idioms using similar strategies. Other studies, instead, focused on verb/particle idioms, e.g. *call off* (Ramisch et al., 2008), or on idioms satisfying specific restrictions, i.e. subject/verb, such as *tension mounted*, and verb/direct-object, e.g. *break the ice* (Shutova et al., 2010).

The following generation of approaches exploited semantic idiosyncrasy, i.e. the linguistic property in which the meaning of an idiomatic expression cannot be completely derived from the meaning of its individual constituents. This property causes idioms to appear in contexts typically unrelated to the meaning of their individual components, hence it provides a key aspect to be exploited in an automatic approach. In particular, Muzny and Zettlemoyer (2013) introduced new lexical and graph-based features that use WordNet[2] and Wiktionary[3], and proposed a simple yet efficient binary Perceptron classifier to distinguish idiomatic and literal expressions by exploiting their components and dictionary definitions. A similar, but unsupervised approach that relied on the dictionary definitions of each component of a given idiom was adopted by Verma and Vuppuluri (2015).

[2]`https://wordnet.princeton.edu/`
[3]`https://www.wiktionary.org/`

Finally, these latter methods have been superseded by approaches making use of distributional similarity in the form of both static and contextualized word embeddings (Gharbieh et al., 2016; Ehren, 2017; Senaldi et al., 2019; Liu and Hwa, 2019; Hashempour and Villavicencio, 2020; Kurfalı and Östling, 2020; Fakharian, 2021; Garcia et al., 2021; Nedumpozhimana and Kelleher, 2021), while keeping the underlying assumption unchanged, that is, the vector representation of the component words should be distant from the vector representation of the context, or of the expression as a whole.

Although efforts have been made in this direction, most of the studies to date have focused on the English language. Additionally, the low performance of current idiomaticity detection systems makes them not very reliable, and therefore such systems tend not to be included in downstream applications. In this work, instead, we propose a high-performance multilingual system for idiomaticity identification.

3 NER4ID

We first describe our architecture for idiomaticity detection (Section 3.1), and then we show how Named Entity Recognition can be included to obtain a more robust idiom identification system (Section 3.2). Figure 1 provides a graphical representation of the overall idiom identification system.

3.1 Dual-Encoder Architecture

In order to distinguish between compositional and idiomatic phrases, we exploit the semantic idiosyncrasy property of idiomatic expressions. This property often implies that when a MWE occurs with its idiomatic meaning, then the meaning of its individual components is unrelated to the surrounding context. On the other hand, when the expression has a compositional meaning, individual words are related to the context. To better explain, consider the following two alternatives in which the potentially idiomatic expression *piece of cake* occurs:

a) Decryption is a <u>*piece of cake*</u> if you know the override codes;

b) Tom ate the last <u>*piece of cake*</u>, but if you want, I'm making another dessert.

In the first case, where *piece of cake* has an idiomatic meaning (i.e. it means *straightforward*),

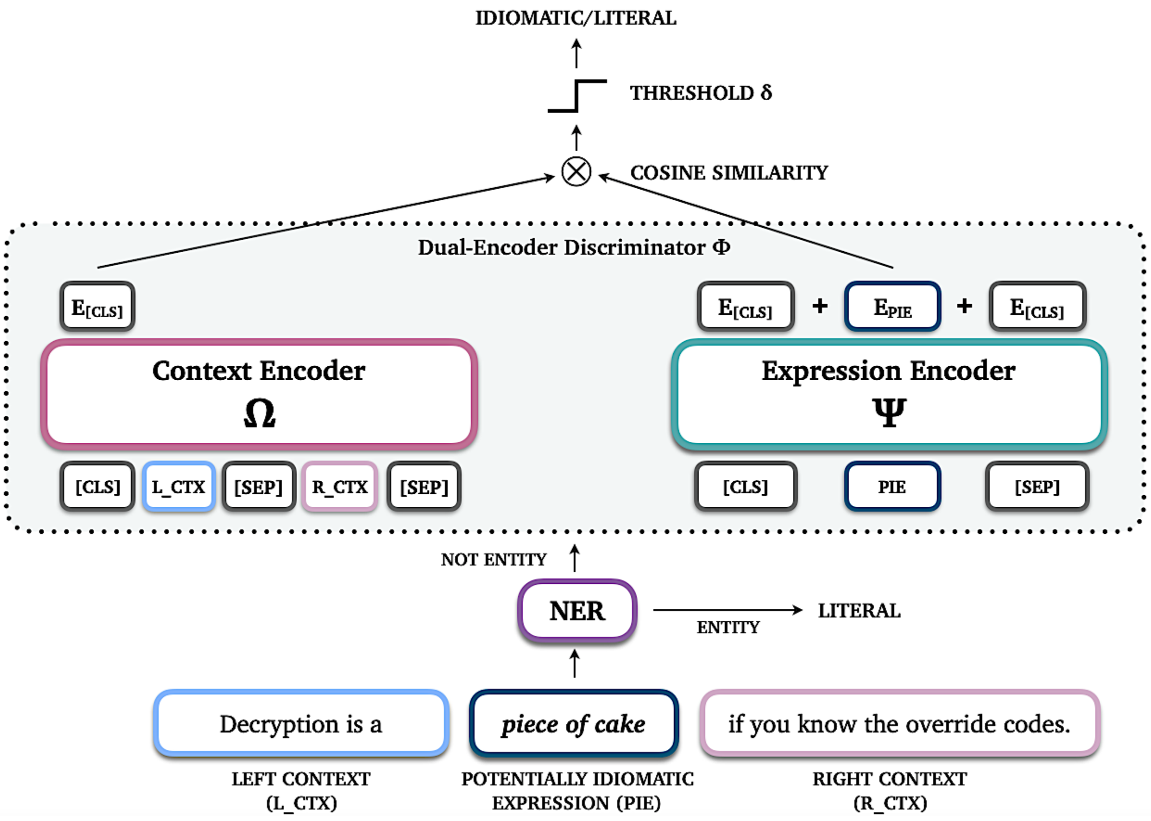


Figure 1: Graphical representation of our architecture for idiomaticity detection. “E" stands for Embedding. A potentially idiomatic expression e is labeled as idiomatic when: i) e is not an entity, and ii) the cosine similarity score between the representations $\Omega(c)$ and $\Psi(e)$, where c is the surrounding context, is lower than the threshold δ.

the word *cake* has nothing to do with the surrounding context. In the second case, instead, we find multiple words whose meaning is related to the meaning of *cake*, i.e. *ate* and *dessert*.

Following the above described intuition, and taking inspiration from recent advances in the main disambiguation tasks (Blevins and Zettlemoyer, 2020; Botha et al., 2020; Tedeschi et al., 2021a), we design a dual-encoder architecture to produce a vector representation for both the expression and its context, and then, based on their cosine similarity, we label the expression as either idiomatic or literal. More formally, let us define an expression encoder Ψ and a context encoder Ω. Then, given an expression-context pair $\langle e, c\rangle$, the output of the dual-encoder architecture Φ is defined as follows:

$$\Phi(e,c) = \begin{cases} 0, & \text{if } \dfrac{\Psi(e)^T\Omega(c)}{\|\Psi(e)\|\,\|\Omega(c)\|} \leq \delta \\ 1, & \text{otherwise} \end{cases} \quad (1)$$

where $\Phi(e,c) = 0$ means that e is idiomatic in c, while $\Phi(e,c) = 1$ if e has a literal meaning in c. δ is a manually-tuned threshold. Both encoders are BERT-based architectures that take as input the tokenized versions of expressions and their contexts, respectively, surrounded by the special tokens [CLS] and [SEP]. To encode an expression, we take the sum of the individual representations of all its subwords. Instead, for the context we take the representation of the [CLS] token.

3.2 Entity or Idiom?

As we discussed in the previous Section, semantic idiosyncrasy is essential for discriminating between idiomatic and literal expressions. However, there are cases in which the individual constituents of a potentially idiomatic expression are unrelated to the context, but the expression as used in that particular context is not idiomatic. Many of these cases correspond to named entities. Table 1 provides a selection of examples – extracted from the Subtask A datasets – in which PIEs are named entities. For instance, in the first example, *Blood Bath* is a movie and, therefore, it does not have an idiomatic meaning. Nevertheless, its constituents (i.e. *blood* and *bath*) are unrelated to the context, hence misleading our dual-encoder architecture (Section 3.1) to classify it as idiomatic.

PIE	Context
blood bath	Deborah Loomis is an actress, known for Hercules in New York (1970), Foreplay (1975) and ***Blood Bath*** (1976).
fine line	***Fine Line*** received generally positive reviews from music critics, particularly towards its production and stylistic influences.
monkey business	***Monkey Business*** is an Action, Adventure, Comedy, Crime movie that was released in 1998 and has a run time of 1 hr 29 min.
rocket science	After finishing "Confrontation", the band shifted to "***Rocket Science***".
night owl	Andrew Gonzalez, owner, ***Night Owl*** Cookies: "Nobody believed in me except for Deco Drive.","They got me on air very quickly!"
silver spoon	Not only is it endorsed by the UK's biggest food brands – Weetabix, Shredded Wheat, ***Silver Spoon***, Carling lager, Marriage's flour – but being Red Tractor also means you can supply different retailers without lots of different requirements.

Table 1: Examples of sentences where potentially idiomatic expressions (PIEs) are named entities.

In order to cope with this issue, we exploit Named Entity Recognition, i.e. the task of identifying specific words as belonging to predefined semantic types, such as Person, Location and Organization (Nadeau and Sekine, 2007). Specifically, we introduce an auxiliary NER module in our classification pipeline that, given as input a raw text sequence of n tokens $X = x_1, \ldots, x_n$ containing a potentially idiomatic expression p, predicts all the entities $E = e_1, \ldots, e_m$ in X. Then, if $p \in E$, p is labeled as literal, otherwise p is provided to the dual encoder, together with its context. To detect further entities, we also exploit capitalization.

4 Experiments

In this Section, we describe our experimental setup (Section 4.1), the datasets we use to train and evaluate our idiom identification system (Section 4.2), and the obtained results (Section 4.3).

4.1 Experimental Setup

We implement our dual-encoder architecture (Section 3.1) with PyTorch (Paszke et al., 2019), using the Transformers library (Wolf et al., 2019) to load the weights of `BERT-base-cased` for English and of `BERT-base-portuguese-cased` for Portuguese and Galician. We fine-tune our idiom identification system for 100 epochs with a Mean-Squared Error loss criterion, adopting an early stopping strategy with a patience value of 20, Adam (Kingma and Ba, 2015) optimizer, and a learning rate of 10^{-5}. Additionally, we set $\delta = 0$[4], and use 32 as batch size, with 4 steps of gradient accumulation. To identify entities, instead, we employ `wikineural-multilingual-ner`[5], a Multilingual BERT (mBERT) model fine-tuned on the WikiNEuRal dataset (Tedeschi et al., 2021b). We compare systems by means of their Macro F_1 scores, as specified by the competition rules. Our final scores are obtained by ensembling the predictions of $N = 9$ model checkpoints[6] and taking the class with the highest number of votes.

Model training was carried out on a NVIDIA GeForce RTX 3090. Each training (i.e. for each model configuration) required ∼1min/epoch on average, for a mean of ∼30 epochs.

4.2 Training, Validation and Test Data

The training, validation and test sets we use in our experiments are those provided for SubTask A[7]. Data statistics are provided in Table 2.

[4]We train our system to produce a cosine similarity score s between a MWE e and its context c, which is $s = -1$ when e is idiomatic in c, or $s = 1$ otherwise. Therefore, in Eq. 1, $\delta = 0$ means that negative similarity scores are mapped to 0 (*Idiomatic*), while positive scores are mapped to 1 (*Literal*).

[5]`https://huggingface.co/Babelscape/wikineural-multilingual-ner`

[6]We use the dev set to search for the optimal value of N by choosing from $N = \{1, 3, 5, 7, 9, 11, 13\}$.

[7]`https://github.com/H-TayyarMadabushi/SemEval_2022_Task2-idiomaticity`

Split	EN	PT	GL	Total
`train-one-shot`	73	73	63	209
`train-zero-shot`	3327	1164	0	4491
`dev`	466	273	0	739
`test`	916	713	713	2342

Table 2: Number of examples in the training, validation and test sets for each of the covered languages: English (EN), Portuguese (PT) and Galician (GL).

	System	EN	PT	GL	ALL
zero-shot	Baseline	70.1	68.0	50.7	65.4
	Our System w/o NER	76.4	63.5	59.7	69.9
	Our System	**86.8**	**70.4**	**65.5**	**77.4**
one-shot	Baseline	88.6	86.4	81.6	86.5
	Our System w/o NER	91.0	86.8	83.9	88.8
	Our System	**95.8**	**88.9**	**87.4**	**92.1**

Table 3: Results of our system with and without the inclusion of the NER module on English (EN), Portuguese (PT) and Galician (GL) languages using the Macro-F_1 score metric. The ALL column reports the overall results. The baselines are provided by task organizers.

In the *zero-shot* setting, potentially idiomatic expressions in the training set are completely disjoint from those in the validation and test sets. In the *one-shot* setting, instead, one positive and one negative example are included for each MWE in the test and validation sets. Finally, note that the zero-shot training set and the validation set cover only English and Portuguese languages, while the test set also contains the Galician language, hence further increasing the difficulty of the zero-shot setting.

4.3 Results

In preliminary experiments, we measure the impact that context inclusion (i.e., the sentences preceding and following the one containing the PIEs) has on our system's performance. Similar to Tayyar Madabushi et al. (2021), we observe a slight drop in performance (i.e., -0.3 F_1 points, on average on the zero-shot and one-shot settings) and longer training times, hence we do not include context in our experiments. Then, in order to show the effectiveness of our dual-encoder architecture (Section 3.1) and of our entire idiomaticity detection system that includes the NER module (Section 3.2), we compare them with the strong mBERT-based baselines provided by the task organizers (Tayyar Madabushi et al., 2022): for the zero-shot setting, their model takes as input the context, while for the one-shot setting, they exclude the context and provide as input only the sentence containing the PIE, where the latter is separated from the rest of the input by using the "[SEP]" special token.

In both zero-shot and one-shot settings, our system far exceeds the performance of the competitive baselines. Specifically, in the zero-shot setting we observe an average improvement of 12 F_1 points for the complete system (Figure 1), and of 4.5 F_1 points using only the dual-encoder architecture (Section 3.1). Likewise, in the one-shot setting we point out an average improvement of 5.6 F_1 points for the overall architecture, and of 2.3 F_1 points for the dual encoder. Therefore, the findings are twofold: i) dual encoders that exploit semantic idiosyncrasy discriminate well between idiomatic and literal expressions, and ii) an idiomaticity detection system can greatly benefit from the inclusion of a NER module in the classification pipeline to manage such ambiguous cases (cf. the examples in Table 1, extracted by using our NER classifier).

5 Conclusions

In this paper, we presented our NER4ID submission to SemEval-2022 Task 2 focusing on the Multilingual Idiomaticity Detection subtask. We started by exploiting the semantic idiosyncrasy property of idiomatic expressions and introduced a novel dual-encoder Transformer-based architecture that encodes both the potentially idiomatic expression (PIE) and its context, and based on their similarity predicts idiomaticity. Further, by manually inspecting our system's errors we discovered critical cases in which, although the individual constituents of a PIE were unrelated to the context, the expressions were not idiomatic in that particular context in which they were used. In most of these cases, the PIEs were part of a named entity. Hence, our second main contribution was devoted to the inclusion of an auxiliary NER module in the idiomaticity detection pipeline in order to avoid these errors. Our experiments showed that: i) our dual-encoder architecture was able to successfully solve the idiom identification task by consistently outperforming the strong baselines provided by the task organizers, and ii) the inclusion of NER in the pipeline provided further improvements of up to 7.5 F_1 points.

As future work, we plan to follow the research line proposed by Tedeschi et al. (2022), and explore the identification of idioms directly on raw texts, i.e., without pre-identified potentially idiomatic expressions, and study a broader set of languages.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487 under the European Union's Horizon 2020 research and innovation programme.

References

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021a. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12564–12573. AAAI Press.

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021b. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4330–4338. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017, Online. Association for Computational Linguistics.

Jan A. Botha, Zifei Shan, and Daniel Gillick. 2020. Entity Linking in 100 Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7833–7845, Online. Association for Computational Linguistics.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kenneth Church, W Gale, P Hanks, D Hindle, and Uri Zernik. 1991. Lexical acquisition: Exploiting on-line resources to build a lexicon. *U. Zernik (Ed.)*, pages 115–164.

Simone Conia, Andrea Bacciu, and Roberto Navigli. 2021. Unifying cross-lingual semantic role labeling with heterogeneous linguistic resources. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 338–351, Online. Association for Computational Linguistics.

Paul Cook, Afsaneh Fazly, and Suzanne Stevenson. 2007. Pulling their weight: Exploiting syntactic forms for the automatic identification of idiomatic expressions in context. In *Proceedings of the Workshop on A Broader Perspective on Multiword Expressions*, pages 41–48, Prague, Czech Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Mona Diab and Pravin Bhutada. 2009. Verb noun construction mwe token classification. In *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications (MWE 2009)*, pages 17–22.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Rafael Ehren. 2017. Literal or idiomatic? identifying the reading of single occurrences of German multiword expressions using word embeddings. In *Proceedings of the Student Research Workshop at the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 103–112, Valencia, Spain. Association for Computational Linguistics.

Samin Fakharian. 2021. *Contextualized embeddings encode knowledge of English verb-noun combination idiomaticity*. Ph.D. thesis, University of New Brunswick.

Afsaneh Fazly and Suzanne Stevenson. 2006. Automatically constructing a lexicon of verb phrase idiomatic combinations. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Waseem Gharbieh, Virendra Bhavsar, and Paul Cook. 2016. A word embedding approach to identifying verb-noun idiomatic combinations. In *Proceedings of the 12th Workshop on Multiword Expressions*, pages 112–118, Berlin, Germany. Association for Computational Linguistics.

Reyhaneh Hashempour and Aline Villavicencio. 2020. Leveraging contextual embeddings and idiom principle for detecting idiomaticity in potentially idiomatic expressions. In *Proceedings of the Workshop on the Cognitive Aspects of the Lexicon*, pages 72–80.

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

Murathan Kurfalı and Robert Östling. 2020. Disambiguation of potentially idiomatic expressions with contextual embeddings. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 85–94, online. Association for Computational Linguistics.

Changsheng Liu and Rebecca Hwa. 2019. A generalized idiom usage recognition model based on semantic compatibility. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6738–6745.

Xiaodong Liu, Kevin Duh, Liyuan Liu, and Jianfeng Gao. 2020. Very deep transformers for neural machine translation.

Amit Mishra and Sanjay Kumar Jain. 2016. A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28(3):345–361.

Grace Muzny and Luke Zettlemoyer. 2013. Automatic idiom identification in Wiktionary. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1417–1421, Seattle, Washington, USA. Association for Computational Linguistics.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Vasudevan Nedumpozhimana and John Kelleher. 2021. Finding bert's idiomatic key. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 57–62.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Carlos Ramisch, Aline Villavicencio, Leonardo Moura, and Marco Idiart. 2008. Picking them up and figuring them out: Verb-particle constructions, noise and idiomaticity. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 49–56, Manchester, England. Coling 2008 Organizing Committee.

Marco Silvio Giuseppe Senaldi, Yuri Bizzoni, and Alessandro Lenci. 2019. What do neural networks actually learn, when they learn to identify idioms? *Proceedings of the Society for Computation in Linguistics*, 2(1):310–313.

Ekaterina Shutova, Lin Sun, and Anna Korhonen. 2010. Metaphor identification using verb and noun clustering. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1002–1010, Beijing, China. Coling 2010 Organizing Committee.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Simone Tedeschi, Simone Conia, Francesco Cecconi, and Roberto Navigli. 2021a. Named Entity Recognition for Entity Linking: What works and what's next. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2584–2596, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Simone Tedeschi, Valentino Maiorca, Niccolò Campolungo, Francesco Cecconi, and Roberto Navigli. 2021b. WikiNEuRal: Combined neural and knowledge-based silver data creation for multilingual NER. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2521–2533, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Simone Tedeschi, Federico Martelli, and Roberto Navigli. 2022. ID10M: Idiom Identification in 10 Languages. In *Findings of the Association for Computational Linguistics: NAACL 2022*, Seattle, Washington. Association for Computational Linguistics.

Rakesh Verma and Vasanthi Vuppuluri. 2015. A new approach for idiom identification using meanings and the web. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 681–687, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

YNU-HPCC at SemEval-2022 Task 2: Representing Multilingual Idiomaticity based on Contrastive Learning

Kuanghong Liu, Jin Wang and Xuejie Zhang
School of Information Science and Engineering
Yunnan University
Kunming, China
liukh@mail.ynu.edu.cn, {wangjin, xjzhang}@ynu.edu.cn

Abstract

This paper will present the methods[1] we use as the YNU-HPCC team in the SemEval-2022 Task 2, Multilingual Idiomaticity Detection and Sentence Embedding. We are involved in two subtasks, including four settings. In subtask B of sentence representation, we used novel approaches with ideas of contrastive learning to optimize model, where method of CoSENT was used in the pre-train setting, and triplet loss and multiple negatives ranking loss functions in fine-tune setting. We had achieved very competitive results on the final released test datasets. However, for subtask A of idiomaticity detection, we simply did a few explorations and experiments based on the xlm-RoBERTa model. Sentence concatenated with additional MWE as inputs did well in a one-shot setting. Sentences containing context had a poor performance on final released test data in zero-shot setting even if we attempted to extract effective information from CLS tokens of hidden layers.

1 Introduction

Meaning of sentence could be captured by compositionality of word representations. However, there widely exists potentially idiomatic phrases in different languages, which are multiword expressions (MWEs) with idiomatic and literal meanings. Therefore, representation of idiomatic phrases is not directly compositional. A previous study has shown that the representation of idiomatic phrases by contextual models, such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019) and some of its variants, are not accurate(Garcia et al., 2021). It will be challenging to represent the MWEs correctly in the downstream tasks (Tayyar Madabushi et al., 2021). SemEval-2022 Task 2, Multilingual Idiomaticity Detection and Sentence Embedding (Tayyar Madabushi et al., 2022), involves English (EN), Portuguese (PT), and Galician (GL). This task includes two subtasks and each subtask contains two settings:

- Subtask A: Determining whether a sentence contains an idiomatic expression. This is a binary classification task with two settings of zero-shot and one-shot. Zero-shot setting means that idiomatic phrases (MWEs) in training examples are completely disjoint to those in development, evaluation, and test sets. In a one-shot setting, MWEs appearing in development and test sets include in the training sentences.

- Subtask B: Outputing the correct Semantic Text Similarity (STS) scores between sentence pairs whether or not either sentence contains an idiomatic expression. The STS scores represent semantical similarity between two sentences ranging from 0 (least similar) to 1 (most similar). This is a regressive task with two setting of pre-train and fine-tune. In the pre-train setting, models require to be trained on any semantic text similarity dataset without idiom. The fine-tune setting should use provided training sets included MWEs.

The remainder of this paper is organized as follows. In Section 2, we describe the structures of model and system. The details about data and implementation and comparative results are presented in Section 3. Finally, a conclusion is drawn in Section 4.

2 System Overview

2.1 Subtask A: Idiomaticity Detection

This is a binary classification task that requires classifying sentences into either *Idiomatic* or *Non-idiomatic*. mBERT (BERT multilingual base model) was used as a pre-trained model in the baseline method. It is a masked language models pre-

[1]The code of this paper is available at: https://github.com/lkh-meredith/SemEval2022_Task2_YNU-HPCC

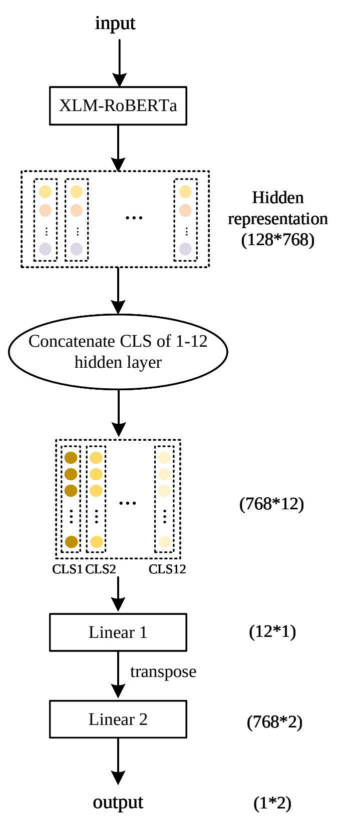


Figure 1: Model architecture of zero-shot setting

trained on the top 104 languages with Wikipedia, which is able to map multilingual representation to the same semantic space, but unable to master cross-lingual information. XLM-RoBERTa (Conneau et al., 2020) was a cross-lingual language model used for our experiment in this subtask, which combined XLM (CONNEAU and Lample, 2019) and RoBERTa (Liu et al., 2019) pre-trained on the 2.5TB of filtered CommonCrawl data containing 100 languages. It is good for the scarce language corpus that could use information learned from larger corpus of other languages. Figure 1 is process of zero-shot setting. Linear 1 layer extracted effective information from concatenated CLS tokens from 1-12 hidden layer of model's output. In the one-shot setting, we simply extracted CLS from the last hidden layer of model's output to classify.

2.2 Subtask B: Sentence Representation

In the pre-train setting , the methodology of baseline was that a sentence transformer model was created by mBERT model adding MWE tokens and training it. As shown in Figure 2 (a), it based SBERT (Reimers and Gurevych, 2019) architecture with the regression objective function, which is a good way of breaking compositionality of idiomatic phrase (Tayyar Madabushi et al., 2021). Figure 2 (b) illustrated one of the methods that we took, which was a siamese network structure of SBERT with classification objective function. Vectors $u, v, \parallel u - v \parallel$ was concatenated as a feature, and $\parallel u - v \parallel$ could play a crucial role in determining if two sentences were similar. Figure 2 (c) illustrates a new method of optimizing cosine similarity, CoSENT (Cosine Sentence), which was proposed by Jianlin Su in his blog post[2]. In the method (c), sentences in a batch are composed of sentences pairs, where two sentences that belong to one sentence pair are adjacent, so they can not be shuffled. The most important part of it was that a loss function based on contrastive learning was designed to maintain training and prediction consistency. CoSENT loss function defined as follows,

$$\log\left[1+\sum_{\mathrm{sim(i,j)>sim(m,n)}} e^{\lambda(\cos(\mathrm{u_m,u_n})-\cos(\mathrm{u_i,u_j}))}\right] \quad (1)$$

where (i, j) and (m, n) are sentence pairs, u_i, u_j, u_m, u_n are sentence embeddings. λ is a hyper-parameter of 20 in our experiment. $e^{\lambda(\cos(u_m,u_n)-\cos(u_i,u_j))}$ is added when label of (i, j) is greater than (m, n), so $\cos(u_i, u_j) > \cos(u_m, u_n)$ is expected in the loss function.

The methodology of baseline used for the fine-tune setting is similar to the pre-train setting: create a sentence transformer model with mBERT adding MWE tokens. This sentence transformer firstly output scores for some of fine-tune data that had no scores (details about data in section 3.1) and trained on them. The questions about the method is that: 1) It is not good to calculate similarity directly between sentence vectors generated by pre-trained model without fine-tuing, which generate static labels and may not accurate. 2) mBERT are not trained on the parallel data, so their vector space across languages are not aligned, which may result in poor results on other languanges. We chose a sentence transformer, distiluse-base-mutilingual-cased (Reimers and Gurevych, 2019), provided in Hugging Face models hub[3], which had been demonstrated to generate good sentence embeddings in

[2]https://kexue.fm/archives/8847
[3]distiluse-base-mutilingual-cased-v1: https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v1
distiluse-base-mutilingual-cased-v2: https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2

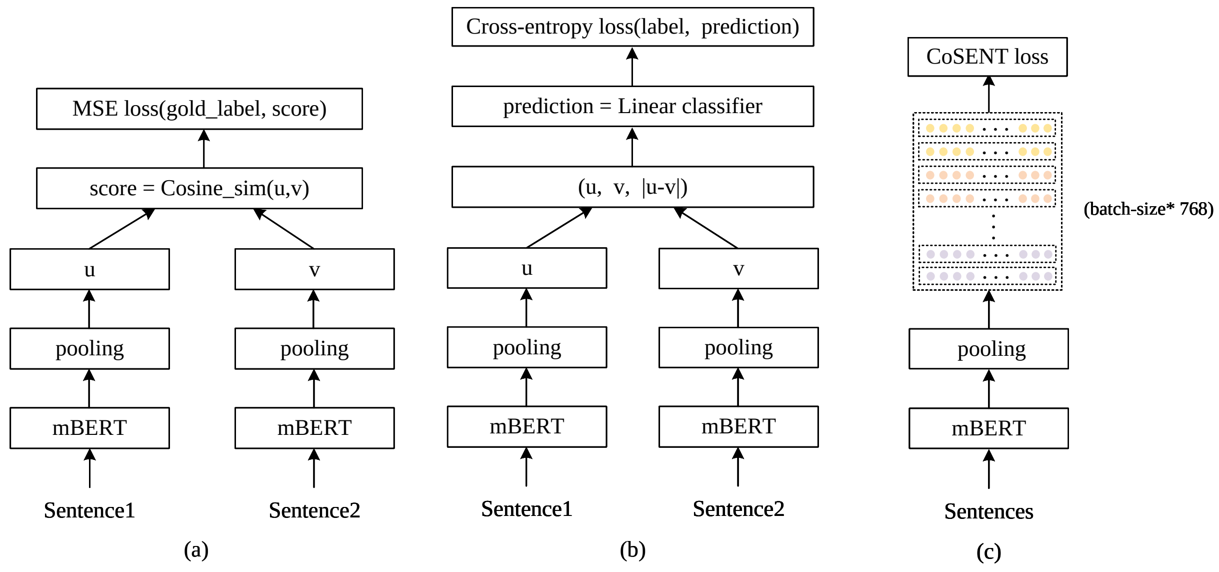
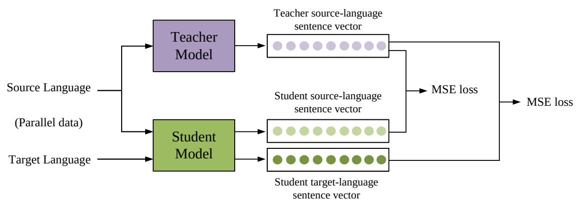


Figure 2: The methods of subtask B pre-train setting



Teacher Model
Student Model
Source Language
(Parallel data)
Target Language
Teacher source-language sentence vector
Student source-language sentence vector
Student target-language sentence vector
MSE loss
MSE loss

Figure 3: The method of making monolingual sentence embeddings multilingual using knowledge distillation

multilingual language and been evaluated in task of multilingual sematic textual similarity. The the difference of its two version, v1 and v2, is that mUSE sentence encoder supports 15 languages and v2 supports 50 including GL respectively. The training approach was achieved by using knowledge distillation. It is able to extend sentence embedding from source language to target ones by sentence pairs of translation (Reimers and Gurevych, 2020). The training procedure of this pre-trained model is shown in Figure 3 using mUSE (Yang et al., 2020) as teacher model and distilmBERT (a distilled version of the mBERT) as student model. Mean pooling of outputs are as sentence vector and minimize the mean-squared loss. Therefore, this model applies to semantic similarity task, and it will be removed last dense layer was as extractor of feature in our experiment.

Triplet loss and multiple negatives ranking loss functions were used to fine-tune. Triplet loss function was used to fine-tune model so that the distance of correct sentence pairs should be closer than incorrect sentence pairs. It computed as follows:

$$\max(\| anchor - pos \| - \| anchor - neg \| \\ + margin, 0) \tag{2}$$

The multiple negatives ranking loss (Henderson et al., 2017) was used to implement and optimize, as shown in Figure 4. $(S_1, S_1') \dots (S_n, S_n')$ were positive pairs. The matrix X represented cosine similarity between sentence pairs in a batch. For a batch of size n, there would be n targets ($y = (0, 1, \dots, n-1)$) treated as labels to represent position of positive pairs in the matrix. With increasing batch sizes, the performance usually is better. The approximated mean negative log probability of data was realized to calculated by cross entropy loss in Pytorch:

$$\begin{aligned} & -\frac{1}{n}\sum_{i=0}^{n-1} \log \frac{e^{X_{i,y_i}}}{\sum_{j=0}^{n-1} e^{X_{i,y_j}}} \\ & = -\frac{1}{n}\sum_{i=0}^{n-1}\left[X_{i,y_i} - \log\left(\sum_{j=0}^{n-1} e^{X_{i,y_j}}\right)\right] \end{aligned} \tag{3}$$

3 Experiments

3.1 Dataset

There includes EN, PT, and GL in datasets. GL is only provided in test dataset and one-shot training data, aiming to test models' ability to transfer learning across languages. Besides, Modern Galician (GL) is part of the West Iberian languages group, which belongs to a family of Romance languages including the Portuguese. A brief intro-

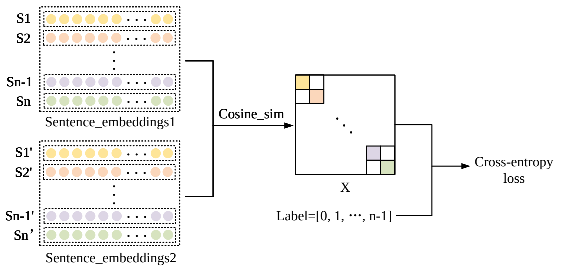


Figure 4: Multiple negatives ranking loss used for positive pairs

duction about datasets describes as follows, and more details are in the task description paper (Tayyar Madabushi et al., 2022).

In the subtask A, training data provided includes zero-shot and one-shot data. A label of 0 indicates *Idiomatic* and a label of 1 indicates *Non-idiomatic*. Zero-shot training data combined with the context, sentences preceding and succeeding the one containing the idioms, are used as training sentences in the zero-shot setting. In the one-shot setting, both the zero-shot and one-shot data are used to train, which exclude the context but add the MWEs as an additional feature. Sentence was concatenated to MWE and separated them by SEP.

Train split of STSBenchmark datasets in English and ASSIN2 datasets in Portuguese are as training dataset in the subtask B pre-train setting. Development and test datasets consist of sentence pairs, some of whom contains idiomaticity and other has no idiomaticity, which replaced by non-idiomatic paraphrases. They will be computed a score (cosine similarity) after model's outputing.

In the subtask B fine-tune setting, development and test datasets are the same as the pre-train setting. Train datasets provided contain EN and PT training examples with type of one-shot and zero-shot totally, and also contain some GL training one-shot examples. For integrity of idiom tokens, MWEs in datasets were added into the tokenizer of model. Training data have positive and negative types. The positive examples mean that sentence with an idiom (S_{MWE}) is the same as the sentence in which the idiom has been replaced by a phrase that correctly represents the meaning of the idiom in context (S_c). So their STS score are equal to 1 ($sim(S_{MWE}, S_c) = 1$). For the negative examples, a sentence with an idiom and the same sentence in which the idiom has been replaced by a phrase that incorrectly represents the meaning of the idiom in context (S_i) should have a low STS score. The score is approximately equal to the STS score between a sentence where the idiom has been replaced by a phrase that correctly represents its meaning and one wherein it incorrectly represents the meaning ($sim(S_{MWE}, S_i) = sim(S_c, S_i)$).

3.2 Evaluation Metrics

Subtask A is evaluated by the Macro F1 score between the gold labels and predictions. F1 score is defined as follows:

$$F1\ score = 2 \times \frac{precision \times recall}{precision + recall} \tag{4}$$

Macro F1 score will calculate average of the precision and recall for all classes firstly, and does not take label imbalance into account.

The metric of subtask B is the Spearman Rank correlation to evaluate outputing STS scores, which is defined as follows:

$$\rho_s = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{5}$$

$$d_i = rank(X_i) - rank(Y_i) \tag{6}$$

where n and d_i denote amount of data and difference between position of variate X and Y after sort, respectively. It computes Pearson correlation coefficient using rank of data sets. This metric uses to evaluate correlation between STS of model's output and gold label.

3.3 Implementation Details

Herein all experiments, we set seed of 4 and AdamW optimizer (Loshchilov and Hutter, 2019) with learning rate 2e-5. Besides, batch-size was set as 32, 4, 16 in zero-shot , one-shot setting and subtask B respectively. A linear learning rate warm-up was 5% and 10% in subtask A and subtask B respectively. A max sequence length of 128 was in subtask A and pre-train setting, 512 in fine-tune setting. Each model was fine-tuned for 10 epochs, where the model exhibiting the best performance on the dev set was used to predict the test set in the competition.

For subtask A of binary classification, cross-entropy loss function was used. The methods of Figure 2 were experimented in the pre-train setting. We set our default pooling strategy was mean, and method (c) used mean pooling of last hidden layer and first-last hidden layer respectively.

In the fine-tune setting, a sentence transformer based on the pre-training model was created. The

Methods	Macro F1 score	
	dev	test
zero-shot setting		
baseline	0.6482	0.6540
mBERT(CLS of last layer)	0.6820	0.6209
mBERT(CLS of 12 layer)	0.6838	0.6030
xlm-R(CLS of last layer)	0.7129	0.6074
xlm-R(CLS of 12 layer)	0.7293	0.6369
one-shot setting		
baseline	0.8691	0.8646
mBERT(CLS)	0.8062	0.7429
xlm-R(CLS)	0.9002	0.8948

Table 1: Results of subtask A

Methods	Spearman's R (All)		Spearman's R (Idiom only)		Spearman's R (STS only)	
	dev	test	dev	test	dev	test
pre-train setting						
baseline	0.6790	0.4810	0.2187	0.2263	0.8182	0.8311
method(a)	0.6736	0.5117	0.1762	0.2582	0.8936	0.8006
method(b)	0.6484	0.5170	0.1905	0.2616	0.6991	0.6195
method(c) first-last-avg	0.7321	0.5602*	0.2111	0.2628*	0.8519	0.7990*
method(c) last-avg	0.7464	0.5650	0.2152	0.2586	0.8574	0.8044
fine-tune setting						
baseline	0.6629	0.5951	0.3459	0.3990	0.5429	0.5961
distiluse-v1	0.7514	0.5523	0.1881	0.2251	0.7939	0.7574
distiluse-v1+2losses	0.8127	0.6648	0.5097	0.4277	0.7248	0.6627
distiluse-v2+2losses	0.7882	0.6391	0.4022	0.3898	0.7154	0.6472

* *0.5602*, *0.2628*, and *0.7990* are the revised results, which are different from the results had been submitted on the evaluation. Because we found out later that there existed something wrong in our former data processing. It had be trained after correcting and used the same method and parameters as before.

Table 2: Results of subtask B

sentence transformer we chose, distiluse-base-multilingual-cased, was extended the vocabulary of MWEs and removed last dense layer was as extractor of feature. Then, mean pooling was applied to output, which generated sentence representation of 768 dimension. Besides, training data were divided into two datasets and used two loss function to fine-tune model. According to negative training data, S_{MWE}, S_c, and S_i constituted a triplet of $(anchor, pos, neg)$. Triplet loss function was used, and we set margin as 0.1 in our experiment. The positive training data only had S_{MWE} and S_c, which composed positive pairs $(anchor_i, pos_i)$. Similar to the training process of unsupervised SimCSE (Gao et al., 2021), $(anchor_i, pos_j) for (i \neq j)$ were as negative pairs. So multiple negatives ranking loss function were used in this part.

3.4 Comparative Results

All of Experimental results are listed in Tables 1 and 2. The reason for results of development and test set difference may be the inconsistent data distribution of them that GL was added in test data. However, the difference of them in zero-shot setting appeared that the sentences including of contiguous context did not make the model learn strong ability of generalization so that all the results on the test set were below the baseline. The results of one-shot setting indicated that classify the sentence and MWE by splicing them together as one texts could be integrated and obtained a better results and xlm-RoBERTa performed better.

In the results of pre-train setting, in comparative to method (a), while method (b) grasped better semantic information of idiom, it could not output good semantic similarity on STS dataset because of training method. It could be found that method (c) could ease their problem and achieved a better result compared with method (a) and (b). The results of fine-tune setting show that these two loss functions play a important role, and the model's understanding of idioms can be improved to some extent by contrastive learning. The distiluse-base-multilingual-cased also used baseline method based on $sim(S_{MWE}, S_i) = sim(S_c, S_i)$. Besides, although v2 supported more languages including GL than v1, it performed a bit poorer instead, which explains further that the model training on more languages appear loss of performance for the information that has been learned.

4 Conclusions

Herein, we discuss and experiment the methods we used in SemEval-2022 Task 2. We participated in two subtasks, idiomaticity detection and representation of idiomaticity. Each of subtasks including two settings, achieved the 19th, 6th, 5th, and 1th places in the final test sets, respectively. Our results showed that the idea introduced contrastive learning in representation of idiomaticity achieved good results, but methods of zero-shot setting in idiomaticity detection did not well and were lack of ability of generalization. We intended to explore and improve the performance of zero-shot and few-shot learning further in future work.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61966038. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Alexis CONNEAU and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32.

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

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 3551–3564.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2227–2237.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94.

OCHADAI at SemEval-2022 Task 2: Adversarial Training for Multilingual Idiomaticity Detection

Lis Kanashiro Pereira[1], **Ichiro Kobayashi**[2]
Ochanomizu University
kanashiro.pereira@ocha.ac.jp[1], koba@is.ocha.ac.jp[2]

Abstract

We propose a multilingual adversarial training model for determining whether a sentence contains an idiomatic expression. Given that a key challenge with this task is the limited size of annotated data, our model relies on pre-trained contextual representations from different multilingual state-of-the-art transformer-based language models (i.e., multilingual BERT and XLM-RoBERTa), and on adversarial training, a training method for further enhancing model generalization and robustness. Without relying on any human-crafted features, knowledge bases, or additional datasets other than the target datasets, our model achieved competitive results and ranked 6th place in SubTask A (zero-shot) setting and 15th place in SubTask A (one-shot) setting.

1 Introduction

Large-scale pre-trained language models such as BERT (Devlin et al., 2019) have achieved great success in a wide range of natural language processing (NLP) tasks. However, more recent studies show that even such contextual models have a limited ability to capture idiomaticity (Garcia et al., 2021). Idiomatic expressions denote a group of words that behave as single words to some extent. Their linguistic behavior cannot be inferred from the characteristics of their components, and still pose a challenge to natural language processing (NLP) systems.

This paper describes the system developed by the OCHADAI team for SemEval-2022 Task 2 - Multilingual Idiomaticity Detection and Sentence Embedding (Tayyar Madabushi et al., 2022). Given that a key challenge in this task is the limited size of annotated data, we follow best practices from recent work on enhancing model generalization and robustness and propose a model ensemble that leverages multilingual pre-trained representations and adversarial training. Our model ranked 6th on SubTask A (zero-shot), and 15th on SubTask A (one-shot).

Setting	Language	Train	Dev	Eval	Test
zero-shot	English	3,327	-	-	-
	Portuguese	1,164	-	-	-
	Galician	0	-	-	-
one-shot	English	87	466	483	916
	Portuguese	53	273	279	713
	Galician	0	0	0	713

Table 1: Summary of the SemEval 2022 Task 2 Subtask A dataset. Note that the dev, eval and test sets are used in both settings.

2 Task Description

SemEval-2021 Task 2 SubTask A consists of a binary classification task that requires classifying sentences with a target multiword expression (MWE) into either "Idiomatic" or "Literal" across English, Portuguese and Galician (Tayyar Madabushi et al., 2021). Further, it is subdivided into two settings to better test models' ability to generalize: zero-shot and one-shot. In the zero-shot" setting, multiword expressions (potentially idiomatic phrases), in the training set are completely disjoint from those in the test and development sets. In the "one-shot" setting, one positive and one negative training examples are included for each MWE in the test and development sets. Note that the actual examples in the training data are different from those in the test and development sets in both settings. Only the datasets provided by the organizers are allowed to train the models. Participants can use only the data provided for the zero-shot setting to train the zero-shot model. However, participants were allowed to use data provided for both settings to train models in the one-shot setting. The statistics of the corpus are presented in Table 1. Our team submitted results for both settings, and the next section outlines the overview of our model.

Setting	Language	Sentence	Target MWE	Label
zero-shot	English	This song is about unconditionally supporting someone you love. This is a **love song**. Let's be there for each other.	love song	1
one-shot	Portuguese	Estamos honrando o teto e construindo as paredes", afirmou. Em outro momento, voltando ao tema do impasse do Orçamento, ele reforçou que a busca é por uma solução, que a briga tem "mérito" e que "sempre, em grande rebanho, tem uma **ovelha negra**"."Mas não é o Congresso nem o grosso do ministério", completou.	ovelha negra	0
one-shot	Galician	Non podemos abandonalos á súa sorte, porque sen mozos e mozas no campo e sen a súa actividade agraria suporía, entre outras cousas, a deslocalización da produción e a dependencia alimentaria Máis alá das políticas comunitarias, ¿cres que poden desenvolverse alternativas para a **xente nova** a partir de medidas municipais, autonómicas ou estatais? Estase a facer?	xente nova	0

Table 2: Example sentences and labels for Subtask A. Note that "Idiomatic" is assigned the label 0 in the dataset and "non-idiomatic" (including proper nouns) are assigned the label 1.

3 System Overview

We focus on exploring different training techniques using BERT and RoBERTa, given their superior performance on a wide range of NLP tasks. Each text encoder and training method used in our model are detailed below.

3.1 Text Encoders

M-BERT (Devlin et al., 2019): We use the M-BERT$_{\text{BASE}}$ model released by the authors. It is pre-trained on the top 104 languages with the largest Wikipedia using a masked language modeling (MLM) objective. This model is case sensitive: it makes a difference between english and English.
XLM-R (Conneau et al., 2019): XLM-RoBERTa (XLM-R) is a multilingual version of RoBERTa. It is pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages. XLM-R has been shown to perform particularly well on low-resource languages, such as Swahili and Urdu. We use the XLM-R$_{\text{LARGE}}$ model released by the authors.

3.2 Training Procedures

Standard fine-tuning: This is the standard fine-tuning procedure where we fine-tune BERT and RoBERTa on each training setting-specific data.
Adversarial training (ADV): Adversarial training has proven effective in improving model generalization and robustness in computer vision (Madry et al., 2017; Goodfellow et al., 2014) and more recently in NLP (Zhu et al., 2019; Jiang et al., 2019; Cheng et al., 2019; Liu et al., 2020a; Pereira et al., 2020). It works by augmenting the input with a small perturbation that maximizes the adversarial loss:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D}[\max_{\delta} l(f(x+\delta;\theta), y)] \tag{1}$$

where the inner maximization can be solved by projected gradient descent (Madry et al., 2017). Recently, adversarial training has been successfully applied to NLP as well (Zhu et al., 2019; Jiang et al., 2019; Pereira et al., 2020). In our experiments, we use SMART (Jiang et al., 2019), which instead regularizes the standard training objective using *virtual adversarial training* (Miyato et al., 2018):

$$\begin{aligned}\min_{\theta} \mathbb{E}_{(x,y)\sim D}[l(f(x;\theta), y)+ \\ \alpha \max_{\delta} l(f(x+\delta;\theta), f(x;\theta))]\end{aligned} \tag{2}$$

Effectively, the adversarial term encourages smoothness in the input neighborhood, and α is a hyperparameter that controls the trade-off between standard errors and adversarial errors.

4 Experiments

4.1 Implementation Details

Our model implementation is based on the MT-DNN framework (Liu et al., 2019a, 2020b). We use BERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2019) as the text encoders. We used ADAM (Kingma and Ba, 2015) as our optimizer with a learning rate in the range $\in \{8\times10^{-6}, 9\times10^{-6}, 1\times10^{-5}\}$ and a batch size $\in \{8, 16, 32\}$. The maximum number of epochs was set to 10. A linear learning rate decay schedule with warm-up over 0.1 was used, unless stated otherwise. To avoid gradient exploding, we clipped the gradient norm within 1. All the texts were tokenized using wordpieces and were chopped to spans no longer than 512 tokens. During adversarial training, we follow (Jiang et al., 2019) and set the perturbation size to 1×10^{-5}, the step size to 1×10^{-3}, and to 1×10^{-5} the variance for initializing the perturbation. The number of projected gradient steps and the α parameter (Equation 2) were both set to 1.

We follow (Devlin et al., 2019) and (Liu et al., 2019b), and set the first token as the [CLS] token and the <s> token, respectively, when encoding the input on BERT and RoBERTa, respectively. We separate the input sentence and the target expression with the special token [SEP] and </s> for BERT and RoBERTa, respectively. e.g. [CLS] Ben Salmon is a committed *night owl* with an undying devotion to discovering new music.,"He lives in the great state of Oregon, where he hosts a killerradio show and obsesses about Kentucky basketball from afar. [SEP] *night owl* [SEP].

For both settings (zero-shot and one-shot), we used the dev dataset released by organizers to fine-tune the model's hyperparameters.

4.2 Main Results

Submitted systems were evaluated in terms of F1-score. The systems were ranked from highest F1-score score to lowest. We built several models that use different text encoders and different training methods, as described in Section 3. See Table 3 for the results.

First, we observe that models that use adversarial training obtained better performance overall, without using any additional knowledge source, and without using any additional dataset other than the target task datasets. These results suggest that adversarial training leads to a more robust model and helps generalize better on unseen data. For the zero-shot setting, the model that uses XLM-R as the text encoder and adversarial training performed better than M-BERT on the development set. Thus, we decided to submit this model's results on the test set. It obtained a test set F1-score of 0.7457, and ranked 6th among all participating systems. On the other hand, on the one-shot setting, M-BERT performed better than XLM-R on the development set. Again, M-BERT with adversarial training performed better than vanilla fine-tuning. This model obtained an F1-score of 0.6573 on the test set, and ranked 15th among all participating systems.

5 Conclusion

We proposed a simple and efficient model for multilingual idiomaticity detection. Our experiments demonstrated that it achieves competitive results on both zero-shot and one-shot settings, without relying on any additional resource other than the target task dataset. Although in this paper we focused on the multilingual idiomaticity detection task, our model can be generalized to solve other downstream tasks as well, and we will explore this direction as future work.

Method	**zero-shot** F1	**one-shot** F1
	Dev	
Standard$_{\text{XLM-R}_{\text{LARGE}}}$	0.7076	0.7769
SMART$_{\text{XLM-R}_{\text{LARGE}}}$	0.7378	0.7958
Standard$_{\text{M-BERT}_{\text{BASE}}}$	0.5540	0.8462
SMART$_{\text{M-BERT}_{\text{BASE}}}$	0.6200	0.8568
	Test	
SMART$_{\text{XLM-R}_{\text{LARGE}}}$	0.7457	-
SMART$_{\text{M-BERT}_{\text{BASE}}}$	-	0.6573

Table 3: Comparison of standard and adversarial training in zero-shot evaluation on various natural language inference datasets, where the standard BERT$_{\text{BASE}}$ model is fine-tuned on the MNLI training data.

Acknowledgements

We thank the reviewers for their helpful feedback. This work has been supported by the project KAKENHI ID: 21K17802.

References

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL).

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2019. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR (Poster) 2015.*

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020a. Adversarial training for large neural language models. *arXiv preprint arXiv:2004.08994.*

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Xiaodong Liu, Yu Wang, Jianshu Ji, Hao Cheng, Xueyun Zhu, Emmanuel Awa, Pengcheng He, Weizhu Chen, Hoifung Poon, Guihong Cao, and Jianfeng Gao. 2020b. The microsoft toolkit of multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:2002.07972.*

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083.*

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993.

Lis Pereira, Xiaodong Liu, Fei Cheng, Masayuki Asahara, and Ichiro Kobayashi. 2020. Adversarial training for commonsense inference. *arXiv preprint arXiv:2005.08156.*

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Thomas Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for language understanding. *arXiv preprint arXiv:1909.11764.*

HIT at SemEval-2022 Task 2: Pre-trained Language Model for Idioms Detection

Zheng Chu[†‡], **Ziqing Yang**[‡], **Yiming Cui**[†‡], **Zhigang Chen**[‡], **Ming Liu**[†]
[†]Research Center for SCIR, Harbin Institute of Technology, Harbin, China
[‡]State Key Laboratory of Cognitive Intelligence, iFLYTEK Research, China
[†]`{zchu,ymcui,mliu}@ir.hit.edu.cn`
[‡]`{zhengchu,zqyang5,ymcui,zgchen}@iflytek.com`

Abstract

The same multi-word expressions may have different meanings in different sentences. They can be mainly divided into two categories, which are literal meaning and idiomatic meaning. Non-contextual-based methods perform poorly on this problem, and we need contextual embedding to understand the idiomatic meaning of multi-word expressions correctly. We use a pre-trained language model, which can provide a context-aware sentence embedding, to detect whether multi-word expression in the sentence is idiomatic usage.

1 Introduction

The goal of the SemEval-2022 Task2 (Tayyar Madabushi et al., 2022) SubtaskA is to detect whether a multi-word expression in a sentence is idiomatic in usage. It is a multilingual task and consists of three languages: English, Portuguese and Galician.

Multi-word expressions (MWEs) are expressions that consist of at least two words and are syntactically or semantically specific. The semantics of MWEs are usually divided into two types, (i) the combination of literal meanings of each word in the phrase or (ii) inherent usages (e.g., idiomatic meaning). Understanding the semantic meaning of a sentence requires the correct identification of the MWE in the sentence. Table 1 contains one case for each of the two usages.

Literal	When removing a **big fish** from a net, it should be held in a manner that suports the girth.
Idiomatic	It was still a respectable finish for both Fadol and Nayre, who were ranked outside the top 500 in the world but caught some **big fish** along the way

Table 1: Examples of idiomatic and non-idiomatic usage

Traditional non-contextual word embedding models, such as word2vec (Mikolov et al., 2013), perform poorly at this task. Simple superposition of non-contextually word embeddings does not correctly express the semantics of idiomatic phrases. Therefore, contextual embedding models (Conneau et al., 2020; Devlin et al., 2019) are required to correctly understand the meaning of multi-word expressions in idiomatic usage.

We used large-scale cross-lingual pre-trained language models, multilingual BERT (Devlin et al., 2019) and XLM-RoBERTa (Conneau et al., 2020), with a softmax classifier on top of the pre-trained LM to train a binary classification model. The training data are processed before training, and regularization dropout (Liang et al., 2021), adversarial training (Miyato et al., 2017; Madry et al., 2018) are used in the training process. In addition, we observed the training data and found an interesting phenomenon that we can get better results by post-processing after the training using heuristic rule.

2 Background

2.1 Task Description

Task 2 contains two subtasks, SubtaskA is idiom detection, and SubtaskB is similarity scoring of texts containing idioms. This article focus on SubtaskA. SubtaskA contains two settings, zero-shot and one-shot.

- Zero-shot: Multi-word expressions that appear in test data do not appear in training data.
- One-shot: Every multi-word expressions that appeared in the test data appeared in training data at least once.
- Data Restriction: Under zero-shot setting, we can only use zero-shot training data, and we can use both zero-shot and one-shot training data under one-shot setting. Test data is same for both settings.

2.2 Data Details

In this section, we will describe the characteristics of the training data and test data. The official data includes eight columns, which are DataID, Language, MWE, Setting, Previous, Target, Next, Label.

2.2.1 Language

The test data includes 916, 713, 713 entries in English, Portuguese and Galician, respectively.

There are only English and Portuguese examples in the training data of zero-shot setting, which means that in the testing phase, the model requires zero-shot transfer of Galician with the learned knowledge on other two languages. Any MWE in zero-shot training data will not appear in the test data.

In one-shot training data, there are 73 entries in each of the three languages, which is relatively small compared to zero-shot training data. Any MWE in the one-shot training data will appear in the test data.

2.2.2 Data Length

We counted the average length of the data to facilitate appropriate truncation when using a pre-trained model.

The average, median, max length of target sentences after tokenizer corresponding to the pre-train model are 42.6, 195, 40, respectively. Over 90% of sentences are 64 or less in length.

The average, median, max position that MWE occurs in sentences after tokenizer is 18.9, 89, 15, respectively, and over 90% of sentences are 37 or less in position.

2.3 Related Work

So far, there has been extensive research about idioms detection. (Zeng and Bhat, 2021) propose a multi-stage neural architecture with attention flow. (Garcia et al., 2021a,b) probs idiomaticity in vector space and propose NCTTI dataset. (Do Dinh et al., 2018) propose a multi-task learning method. (Tayyar Madabushi et al., 2021) present a multilingual idiom detection dataset, which will be used as this SemEval-2022 idioms detection track.

Pre-trained word embedding can capture syntactic and semantic information from large amounts of unlabeled data, which has been a standard part of natural language processing task (Mikolov et al., 2013; Pennington et al., 2014). However, each of these methods can only obtain a fixed, non-contextual vector representation for each word which makes it difficult to convey the correct meaning of polysemous words. Due to the disadvantages of non-contextual embedding, recent work has begun to focus on contextual embedding, typical cases are context2vec (Melamud et al., 2016), ELMo (Peters et al., 2018), BERT (Devlin et al., 2019).

The most commonly used in contextual embedding is the pre-trained language model (Devlin et al., 2019; Conneau et al., 2020). These models perform self-supervised training through mask language modeling, next sentence predicting, and other objectives in hundreds of millions of unlabeled datas. Benefiting from multilingual training data, these models have cross-language capabilities. Pre-trained models are gradually taking the place of pre-trained word embeddings as the new paradigm for natural language processing.

3 System Overview

Figure 1 depicts the flow chart of the whole system. We first preprocess the data, tokenizing and then feed into a pre-trained model to get hidden states. Apply some pooling method to get fixed length sentence representation to train a softmax binary classifier. After that, the prediction results of the model are post-processed. In the training process, contrastive learning, adversarial training, regularized dropout, etc. are used. Table 2 is then used as an example to introduce the data preprocessing process of the system.

3.1 Baseline

The baseline method below refers to the method in paper (Tayyar Madabushi et al., 2021).

- In the zero-shot setting, Multilingual BERT is trained on zero-shot data, using the context without idiom as an additional feature.
- In the one-shot setting, Multilingual BERT is trained on combination of the zero-shot and one-shot data, excluding the context and adding the idiom as an additional feature.

3.2 Data Preprocessing

3.2.1 Truncation

The data provides the context of the target sentence. We only use target sentences for training because we found that if we concatenate previous, target,

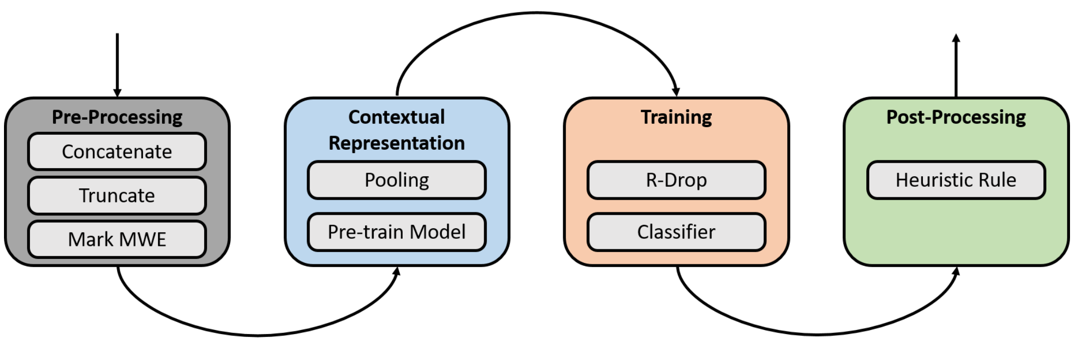


Figure 1: System flow diagram

and next together, the sentence length will be too long, which slows down training and harms performance. We guess that to distinguish whether the MWE is an idiomatic usage, we only need to focus on words near the MWE, too long sentences will introduce unnecessary distractions.

According to the length statistics in the previous chapter, 128 is used as the maximum truncation length of the pre-trained model, which can ensure that most sentences will not be truncated and keep the sentence length as small as possible.

3.2.2 MWE Marking

Following the baseline method, we use the tokenizer's [SEP] token to mark the MWE in the sentence. Unlike the baseline method, we only mark MWEs without deformation. Proper nouns are usually non-idiomatic usage, and are often deformed. Pre-trained models can recognize proper nouns well, so we do not mark the deformed MWEs. The results also show that this gives better performance.

Example in Table 2, MWE 'milk tooth' in sentence "Her latest pamphlet **Milk Tooth**, published by Rough Trade Books, is a collection of thwarted escape plans for a too-heavy world" is capitalized, according to our rules, the MWE in this sentence is not marked. If the MWE in the sentence is not deformed, the [SEP] token will be used to mark the MWE, just like [SEP]**milk tooth**[SEP].

3.3 Model

3.3.1 Pre-trained Model

We tried different multilingual pre-trained models, including mBERT and XLM-RoBERTa, and XLM-RoBERTa consistently outperforms mBERT. In addition, we also try to use different size models, including mBERT-base-cased, XLM-RoBERTa-base, XLM-RoBERTa-large, and bigger models lead to better performance.

3.3.2 Classifier

Different pooling methods are used for hidden states of different layers, including mean pooling, max pooling, [CLS], and token-level pooling. The results show that different pooling methods have little effect on the results. For simplicity, [CLS] is used as the sentence representation.

For token-level pooling, we pool the vectors of MWE positions in hidden states to obtain the final sentence representation, which harms the performance.

After pooling on the pre-trained model, fixed-length sentence representation is obtained. This is followed by a full connection layer with dropout (Srivastava et al., 2014) and a softmax classifier.

3.3.3 Regularized Dropout

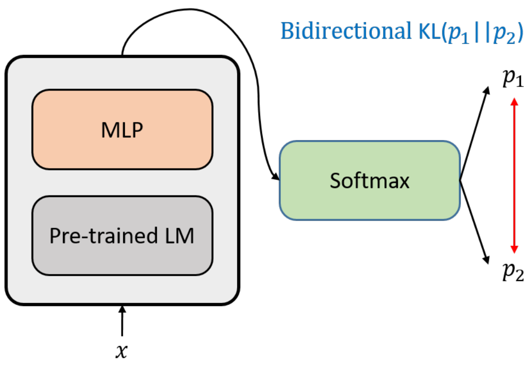


Figure 2: Regularized Dropout

Deep neural networks usually use dropout (Srivastava et al., 2014), but the use of dropout introduces inconsistency between train and inference. R-drop (Liang et al., 2021) is a means of regularization in the training process to mitigate this inconsistency.

A sample output through the pre-trained model, MLP and softmax can be regarded as a probability distribution. R-drop performs two independent

MWE	milk tooth
Previous	A ritual sacrifice from the 19th century is vividly relieved.
Target	Her latest pamphlet **Milk Tooth**, published by Rough Trade Books, is a collection of thwarted escape plans for a too-heavy world.
Next	In these poems of trauma and transformation, the present throbs with unfinished histories.
Label	1

Table 2: Training data example (useless columns have been removed)

forward calculations for each sample, obtaining two outputs probability distribution. Due to the dropout, these two outputs will be slightly different, introducing inconsistency. To mitigate this, the bi-directional KL-divergence is calculated as a penalty between these two outputs probability distributions. In the following equation, y_i represents the label, x_i represents the input, and the superscript represents two independent forward operations. p_i^1, p_i^2 represent probability distribution obtained from two independent forwards.

$$\mathcal{L}_{CE}^{i} = CE(x_i^1, y_i) \ + \ CE(x_i^2, y_i) \tag{1}$$

$$\mathcal{L}_{KL}^{i} = \frac{1}{2}(\mathcal{D}_{KL}(p_i^1||p_i^2) + \mathcal{D}_{KL}(p_i^2||p_i^1)) \tag{2}$$

$$\mathcal{L}^{i} = \mathcal{L}_{CE}^{i} + \alpha \cdot \mathcal{L}_{KL}^{i} \tag{3}$$

3.3.4 Post Processing

We found an interesting phenomenon in the training data. Some MWEs in one-shot training data have only one category label, and most of these MWEs corresponding to entries in the dev data have the same label as in the training data. Some proper nouns are labeled with idiomatic meanings, however some with literal meanings. These labeling inconsistencies may cause problems in the learning of the model, so we design a heuristic rule. On top of the model prediction results, if there is only one label for a certain MWE in the training set, then replace all the predictions for that MWE in the test set with whichever label appears in the training set.

4 Experiment

4.1 Hyperparameters

We use the Huggingface Transformers (Wolf et al., 2019) implementation of mBERT and XLM-RoBERTa. During the training, the learning rate schedule strategy is warmup of first 10% steps with cosine learning rate decay in rest steps. For zero-shot and one-shot, we use learning rates of 1e-5 and 3e-5, respectively, and one-shot is continued training on the well-performing zero-shot model. The mini-batch size is 32. The coefficient of R-drop is chosen from 0, 1, 2, 4. We train a total of 20 epochs and save the best-performing checkpoints on the development set. All models are trained on one single NVIDIA Tesla V100 GPU.

Model	Zero-shot	One-shot
mBERT$_{base}$ w/ C w/ I	74.90	84.78
mBERT$_{base}$ w/ C w/o I	70.59	76.98
mBERT$_{base}$ w/o C w/ I	**75.31**	**85.76**
mBERT$_{base}$ w/o C w/o I	70.76	82.59

Table 3: Context and idiom effects on the development set results. C: context. I: Idiom. (Macro F1 $\times$ 100)

Model	Max Len	Zero-shot	One-shot
mBERT$_{base}$	128	**76.31**	**87.97**
mBERT$_{base}$	192	75.62	87.96
mBERT$_{base}$	256	75.64	86.24

Table 4: Effect of different maximum sentence lengths on the development set results. (Macro F1 $\times$ 100)

4.2 Evaluation Metrics

SubtaskA is evaluated using the Macro F1 score between the gold labels and model predictions.

$$Macro\ F1 = 2 \cdot \frac{precision \times recall}{precision + recall} \tag{4}$$

4.3 Hyperparameters Selection

We compare different pre-trained models, max length, and whether to use MWE and contexts.

4.3.1 Context and Idiom

We compare the performance of the model with and without context in Table 3, the effect of marking idioms on the results. Moreover we conduct experiments on BERT$_{base}$, using [CLS] as pooling, with the max sentence length set to 128. The method of marking MWEs here follows the baseline.

Experiments show that ignoring context and mark idioms gives better results, and this setting will be continued for future experiments.

Model	Zero-shot	One-shot
mBERT$_{base}$	75.31	87.97
XLM-R$_{base}$	76.99	89.15
XLM-R$_{large}$	**78.17**	**91.84**

Table 5: Effect of different pre-trained models on the development set. (Macro F1 $\times$ 100)

4.3.2 Max Sentence Length

In this section, a comparison is made between the cases with different maximum sentence lengths. The model follows the previous setting, with idioms marked and context ignored, using first-last-avg as pooling for training.

Our original hypothesis is that performance and speed are a trade-off as the maximum sentence length increases. In contrast, Table 4 shows that a maximum sentence length of 128 is sufficient in terms of speed and performance. We do not test a smaller maximum sentence length because further reduction might cause parts of the sentence to be truncated, harming the performance.

4.3.3 Pre-trained Model

We compared mBERT$_{base}$, XLM-RoBERTa$_{base}$, and XLM-RoBERTa$_{large}$, and results in Table 5 demonstrated that XLM-RoBERTa outperformed mBERT, and the large model performed better than base model. In addition, our final submission results were obtained using XLM-R$_{large}$.

From the results, we found that the performance improvement is evident as the model size grows, and there is no bottleneck yet. Therefore, increasing the model size may be a simple and effective way.

5 Results

5.1 System Performance

Our final results use model fusion on thirteen models. The zero-shot model finished fourth with an F1 of 77.15, and the one-shot model finished first with an F1 of 93.85.

5.2 Ablation Study

Table 6 provides the results of the ablation study. The baseline of the ablation experiment follows the hyperparameters of the experiment chapter, except that the model is replaced with XLM-R$_{large}$.

Model	Zero-shot	One-shot
XLM-R$_{large}$	78.05	91.02
+mark MWE	78.17	91.84
+contrastive pre-train	-	89.95
+contrastive auxiliary	76.30	88.05
+AEDA	79.09	89.76
+AT	79.78	92.47
+R-drop	**80.34**	**92.91**
+post-processing	-	**93.73**

Table 6: Ablation experiments on the development set. (Macro F1 $\times$ 100)

5.2.1 Mark MWE

First, we change the method of marking MWE in the baseline. We use [SEP] token for tagging only if the MWE in the sentence is the same as the MWE provided in the data. The reason for this is that the organizer's rule is to label proper nouns as literal meaning, and proper nouns are usually deformed with initial capitalization. The pre-trained model can distinguish proper nouns well under the training of a large amount of corpus.

5.2.2 Adversarial Training

Adversarial training (Miyato et al., 2017; Madry et al., 2018) is a way to enhance the robustness of neural networks by adding small perturbations to the samples to interfere with the predictions of the model. In our experiments, the results in Table 6 show that adversarial training significantly improves performance.

5.2.3 R-drop

R-drop is a regularization tool that aims to maintain the consistency of model prediction and training while using dropout by adding bi-directional KL-divergence as a penalty term. After adding R-drop, the performance is significantly improved, exceeding the adversarial training. In Table 6, under zero-shot setting, the relative improvement is 2.17 and 0.56 compared to baseline and adversarial training, respectively, and this improvement is 1.07 and 0.44 under one-shot setting, respectively.

5.2.4 Heuristic Rule

We used the heuristic rule mentioned in 3.3.4 for replacement under the one-shot setting, and as shown in the Table 6, the relative improvement is 0.82 percentage points.

5.2.5 Negative Results

Contrastive Learning: Recently, contrastive learning has been a hot topic in NLP, especially in sentence representation learning. Inspired by SimCSE (Gao et al., 2021), we use contrastive learning before and during training classification, using the data from subtaskA for contrastive pre-train and apply contrastive loss during training as an auxiliary training objectives, respectively. Unfortunately, as shown in Table 6, both of these methods make the results much worse.

Data Augmentation: Due to lack of training data, we use data augmentation for expansion. We used AEDA (Karimi et al., 2021) as a means of data augmentation, which is a straightforward data augmentation, by adding punctuation marks to the sentences. The results showed that a particular improvement was achieved under zero-shot setting, but a decrease was achieved under one-shot setting.

6 Conclusion

We continuously improve the model performance by improving the MWE marking method, using larger pre-trained models, adding regularization terms and heuristic rules. Inspired by pre-trained models, we can find that future work needs to focus more on external data besides training data, especially data with specific idioms, such as idiom dictionaries, because the size of the external data is much larger than the training data, which has not been fully exploited.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Erik-Lân Do Dinh, Steffen Eger, and Iryna Gurevych. 2018. Killing four birds with two stones: Multi-task learning for non-literal language detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1558–1569, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021a. Assessing the representations of idiomaticity in vector models with a noun compound dataset labeled at type and token levels. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2730–2741, Online. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021b. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. AEDA: An easier data augmentation technique for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2748–2754, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. *CoRR*, abs/2106.14448.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference* *on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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

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

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Ziheng Zeng and Suma Bhat. 2021. Idiomatic Expression Identification using Semantic Compatibility. *Transactions of the Association for Computational Linguistics*, 9:1546–1562.

SemEval-2022 Task 3: PreTENS – Evaluating Neural Networks on Presuppositional Semantic Knowledge

Roberto Zamparelli[1], Shammur A Chowdhury[2], Dominique Brunato[3], Cristiano Chesi[4], Felice Dell'Orletta[3], Arid Hasan[5], Giulia Venturi[3]
[1]CIMeC, University of Trento, Rovereto Italy;
[2]Qatar Computing Research Institute, HBKU, Qatar; [3]ILC-CNR, Pisa, Italy;
[4]NETS-IUSS, Pavia, Italy; [5]Daffodil International University, Dhaka, Bangladesh
roberto.zamparelli@unitn.it, shchowdhury@hbku.edu.qa,
cristiano.chesi@iusspavia.it, arid.cse0325.c@diu.edu.bd, [name.surname]@ilc.cnr.it

Abstract

We report the results of the SemEval 2022 Task 3, PreTENS, on evaluation the acceptability of simple sentences containing constructions whose two arguments are presupposed to be or not to be in an ordered taxonomic relation. The task featured two sub-tasks articulated as: *(i)* binary prediction task and *(ii)* regression task, predicting the acceptability in a continuous scale. The sentences were artificially generated in three languages (English, Italian and French). 21 systems, with 8 system papers were submitted for the task, all based on various types of fine-tuned transformer systems, often with ensemble methods and various data augmentation techniques. The best systems reached an F1-macro score of 94.49 (sub-task1) and a Spearman correlation coefficient of 0.80 (sub-task2), with interesting variations in specific constructions and/or languages.

1 Introduction

A growing body of literature on the cognitive side of computational linguistics has tried to probe the ability of language models to recognize deviant linguistic structures. Recognizing linguistic ill-formedness requires some degree of metalinguistic awareness in adult humans (i.e. the ability to say not just that there is 'something wrong' in a sentence, but also where the problem lies or how the sentence could be improved), and it is not clear whether and to what extent artificial systems can induce this type of knowledge purely from exposure to raw linguistic data (Linzen and Baroni, 2020). Most of the previous investigations on this topic have focused on phenomena that are purely syntactic (agreement, Gulordava et al. 2018; dislocated arguments with island effects, Wilcox et al. (2018); Warstadt et al. (2019); Chowdhury and Zamparelli (2018), clause embedding, Futrell et al. 2019, etc.) or at the syntax/semantics interface (negative-polarity items, Jumelet and Hupkes 2018; argument structure, quantifier restrictions, Warstadt et al. 2019), mostly using LSTM architectures (but see Tran et al. 2018; Ettinger 2020). The fact that many purely semantic effects result in the (non) availability of certain readings (e.g. the scope of a quantifier over a higher negation, in "I didn't see some people") makes it of course harder to translated them into computationally testable paradigms.

The task we describe in this paper focuses on an area of purely semantic competence that, to the best of our knowledge, is unexplored in the computational literature, and one which gives rise to a robust intuition of deviance, triggered by the failure of a type of **presupposition**: the requirement for two nominal arguments to be (or not be) in an (ordered) **taxonomic relation**. These presuppositions are introduced by a wide variety of constructions, such as comparatives (1a), coordinations, verbs like *prefer*, modifiers headed by *type* or *except* (1b) etc. (see Table 1 for the full list).

(1) a. I hate guns more than {*weapons / social media}.
b. I like dogs, except {*birds / greyhounds}

Distinguishing the deviant from the acceptable cases requires the ability to (i) detect taxonomic relations and (ii) recognize those constructions which place restrictions on them. The first point is of course crucial for most tasks in Natural Language Inference (NLI) — an active and fast-growing field in the NLP community, with various datasets and benchmarks (e.g. GLUE Wang et al. 2018, SuperGLUE Wang et al. 2019). NLI datasets, however, normally assume that felicity conditions are satisfied. The present dataset, which we call PreTENS, takes a step back and aims to verify if a computational model can detect when this presupposition fails.

The task requires world knowledge, commonsense knowledge (in the sense discussed in Storks et al. 2019), but also linguistic knowledge, to catch

Construction (Variants)	**Example**	**Presup.**
EXCEPT (2)	I like [$_{A1}$ dogs] except [$_{A2}$ {*cats / pugs / *animals}]	**A1>A2**
PARTICULAR (2)	I like [$_{A1}$ dogs], and in particular [$_{A2}$ {*animals / *cats / pugs}].	**A1>A2**
IN GENERAL	I like [$_{A1}$ dogs], and [$_{A2}$ {animals / *cats / *pugs}] in general.	**A1<A2**
GENERALLY	I like [$_{A1}$ dogs], and more generally [$_{A2}$ {animals / *cats / *pugs}].	**A1<A2**
TYPE (2)	I like [$_{A1}$ dogs], an interesting type of [$_{A2}$ {animal / *cat / *pug}].	**A1<A2**
AND-TOO	I like [$_{A1}$ dogs], and [$_{A2}$ {cats / *pugs / *animals}] too.	**$A1 \not\lessgtr A2$**
COMPAR. (3)	I like [$_{A1}$ dogs] more than [$_{A2}$ {cats / *pugs / *animals}]	**$A1 \not\lessgtr A2$**
DRATHER	I would rather have [$_{A1}$ dogs] than [$_{A2}$ {cats / *pugs / *animals}]	**$A1 \not\lessgtr A2$**
PREFER	I don't like [$_{A1}$ dogs], I prefer [$_{A2}$ {cats / *pugs / *animals}]	**$A1 \not\lessgtr A2$**
UNLIKE	Unlike [$_{A1}$ dogs], [$_{A2}$ {*animals / cats / *pugs}] are often mentioned in this text .	**$A1 \not\lessgtr A2$**
BUT-NOT	I like [$_{A1}$ dogs], but not [$_{A2}$ {*animals / cats / pugs}]	**$A1 \not\lessgtr / > A2$**

Table 1: Distribution of taxonomic constructions and their presuppositions. $\not\lessgtr$ indicates no overlap; (*n*) indicates n variants on the construction (e.g. COMPAR. contains samples of majority, minority and equality comparatives). The BUT-NOT case is probably ambiguous, with one meaning close to EXCEPT; the same applies to GENERALLY, which draw uncertain results in the human evaluation and was excluded from sub-task 2 in favour of IN GENERAL.

the requirement of the specific presupposition-inducing constructions. In this respect, the present task is closer in spirit to SemEval-2020 task 4, sub-task A, on the validation of sentences for common-sense (Wang et al., 2020), than to SemEval 2016 task 3, where participants had to extract and identify the taxonomic relationships between two terms (Bordea et al., 2016).

Besides NLI practitioners, the task could be relevant for researchers interested in the potential of NNs as cognitive/linguistic models (see e.g. Warstadt et al. 2019). We believe that it is also a potentially useful addition to the toolbox of probes used to understand the inner working of current language models.

2 Dataset and Task description

2.1 Composition

The PreTENS contains 21,765 artificial sentences with 2-argument relations filled by nominals (the **argument nouns**). The sentences were designed to follow or flout the presuppositions that (i) the argument nouns should or should not be in a taxonomic relation (i.e. one a subset of the other: *dogs* < *animals*) and (ii) when a taxonomic relation was required, the order should be a specific one. (ii) differentiates *I like dogs, and in particular pitbulls* from **I like pitbulls, and in particular dogs*). The list of constructions used is in Table 1.

The data for this task was programmatically generated from a human-verified template, yielding sets of sentences that are extremely similar across constructions. The argument nouns (A1 and A2 in Table 1) are taken from the following semantic categories: *dogs, birds, animals, cars, motorcycles, cutlery, clothes, trees, plastics, furniture, wine, animals, sports, music, vegetables, fruits, pork-based food, desserts, seafood, apartments, movies, jewelry, pets, rain, nature, senses, emotions, books, workers* and *scientists*, and repeat across constructions. The elements not in taxonomic relations were chosen to maximize the plausibility of comparison (e.g. *dogs* if the semantic category was *birds*) and the verbs were chosen to be as semantically neutral as possible (often *like* or *have*, but e.g. *trust* in the semantic category of *senses*). The English template file was created and revised (using dictionaries and Wordnet) by the task proponents, all expert linguists, and double-checked by a native speaker.

PreTENS is a simplified, no-repetiton subset of a larger dataset, DuckRabbit, which also contains 5 semantic categories (*countries, cities, painters, politicians, actors*) examplified by well-known proper names (e.g. *Paris, Picasso, Obama*), which we decided not to use for the PreTENS task. The full DuckRabbit dataset (55,296 items) is arranged in a way that systematically tests all the possible orders of pairs of argument nouns taken from a super-category, a subcategory in the same taxonomic domain and a distractor (non taxonomically ordered with either, e.g. <*birds, parrots, dogs*>). This arrangement, however, creates a large number of repeated entries.

The fixed nature of the patterns used allowed us to propose the dataset in three languages (English, Italian and French), where the French and Italian versions are slightly adapted translations of the English dataset.[1] Adding more languages would

[1] A key difference was that the English bare plurals used in generic sentences were replaced by NPs introduced by definite

be relatively straightforward. The template and the scripts used to generate the data are publicly available under a CC BY 3.0 "Attribution" license.[2] As far as we can tell, the contents do not raise any issue w.r.t. ethics or privacy.

2.2 Definition of the Task

The task was articulated into the two sub-tasks:

- a **binary classification task** (hereafter referred to as *sub-task1*), which consisted in predicting the acceptability label assigned to each sentence of the test set on the basis of a theoretical linguistic model;
- a **regression task** (hereafter *sub-task2*), which consisted in predicting the average score on a 7 point Likert-scale assigned by human annotators to a subset of data evaluated via crowdsourcing (see Section 2.3).

For each task and each language, the dataset was split into training and test sets. The classification task was composed of 5,838 training samples and 14,560 testing samples; the regression task, of 524 sentences in training and 1,009 in test. Table 2 reports the internal composition of the training and test dataset of each sub-task. As it can be seen, not all the constructions contained in the test were provided in the training set. This choice was deliberate, to test the generalization abilities of the systems across unseen constructions. The sentences in training data were independently randomly ordered in the three languages, to discourage mapping the results obtained in one language to sentences with the same ID in the other languages.

2.3 Annotation with human judgments

The dataset released for *Sub-task2* is composed by a subset of 1,533 sentences taken from the whole dataset, corresponding to about 5% of the total and representative of the patterns considered, which were judged by human annotators via a crowdsourcing campaign.

The purpose of this evaluation was two-fold: (i) to provide a bottom-up assessment of the quality of

Constructions	sub-task1		sub-task2	
	Training	**Test**	**Training**	**Test**
and-too	835	525	131	88
but-not	831	526	131	88
comparatives	835	3,245	131	88
drather	–	1,360	–	–
except	831	1,887	–	–
in general	–	–	–	219
generally	–	1,360	–	–
particular	835	1,885	–	219
prefer	835	525	–	–
type	835	1,887	131	88
unlike	–	1,360	–	219
TOTAL	**5,838**	**14,560**	**524**	**1,009**

Table 2: Distribution of taxonomic constructions in terms of number of sentences in the dataset.

the linguistic categories that informed the creation of the dataset templates; (ii) to obtain more fine-grained judgments of semantic acceptability in the form of gradual, rather than categorical, scores.

The annotation was performed through the Prolific[3] platform. Specifically, for each language the annotation process was split into different tasks, each one consisting in the annotation of about 150 randomly mixed sentences for the typologies reported in Table 1. For all tasks, we recruited 12 native speakers, who were asked to read each sentence and answer the following question:

> *How acceptable is this sentence from 1 (completely unacceptable) to 7 (completely acceptable)?'*

As an example, we report below two sentences (with corresponding average score) from the English portion of the annotated dataset, which were rated as very poorly and very highly acceptable:

> *I like politicians, an interesting type of farmer* (1.42)
>
> *I like governors, an interesting type of politician* (6.16)

Table 3 provides the average value (μ) and standard deviation (σ) of acceptability labels for the whole dataset (first row) and for sentences classified according to the various constructions. As it can be noted, French sentences were evaluated on average as more acceptable than Italian and English ones but with a slightly higher standard deviation. While for all languages the maximum average score on the Likert scale was obtained by very few sentences (i.e. only one sentence for English and French and four for Italian), the number

determiners in Italian and French. This makes the latter sub-datasets systematically longer. In addition, certain English nouns required compounds or N+PPs to be rendered in the other languages.

[2]Github Repository: https://github.com/shammur/SemEval2022Task3
Task Website: https://sites.google.com/view/semeval2022-pretens/

[3]www.prolific.co

	ENG		ITA		FRE	
	μ	σ	μ	σ	μ	σ
All_sents	3.89	1.61	3.75	1.73	4.05	1.85
and-too	4.83	0.92	4.98	0.97	5.29	0.95
but-not	4.59	1.08	3.94	1.06	5.07	0.94
comparatives	4.91	1.93	5.05	1.42	5.18	1.41
in general	3.64	1.17	3.28	1.29	3.78	1.06
particular	2.54	1.48	2.36	1.46	2.28	1.50
type	2.13	1.45	1.89	1.44	1.80	1.40
unlike	4.56	0.84	4.74	1.15	4.93	1.93

Table 3: Statistics about the distribution of human judgments in the dataset collected for sub-task2. μ = average judgment; σ = standard deviation.

of sentences rated with the lowest score is higher for Italian and French (i.e. 42 and 45 respectively) than for English (i.e. 7). If we focus on the distribution of judgments across the distinct constructions, we observe that examples containing the TYPE construction were perceived on average as the less acceptable ones for all languages. Conversely, sentences belonging to the AND TOO and COMPARATIVES categories obtained the highest acceptability scores.

In order to see how consistent was the human perception of semantic acceptability across languages, we computed the Pearson's r between the average scores assigned to the whole set of sentences for each pair of languages. The correlation scores were very high, with the highest scores obtained between sentences in French and Italian (i.e. 0.86), followed by English and French (i.e. 0.80), and, lastly, by English and Italian (i.e. 0.77)[4].

Finally, an additional outcome that we want to highlight here is the strong connection between the theoretically-driven and the human-based semantic acceptability label, which was assessed by calculating the Spearman's rank correlation coefficient between the average human scores and the binary acceptability labels attributed to the same set of sentences, for all languages. In this case, too, we found a very high correlation, although weaker in English (ρ=.73) than in Italian and, especially, French (ρ =.78 and .83, respectively).

3 Shared Task Organisation

Shared Task Phases We ran the shared Task 3 in two phases. In the first phase, we released the baseline pipeline, along with the cross-validation results on the official training set and introduced the participants to the aforementioned task evaluation measure.

The second phase – the main Evaluation Phase – was conducted using codalab platforms for both sub-task1[5] and sub-task2.[6] During this phase, the participants were provided with the test sets and were allowed to submit their predictions to the system. The number of submissions of each participant was limited to three, but the participant could choose among them which runs/submission they want to display in the leader-board. During the evaluation phase, the leader-board was visible to all the participants.

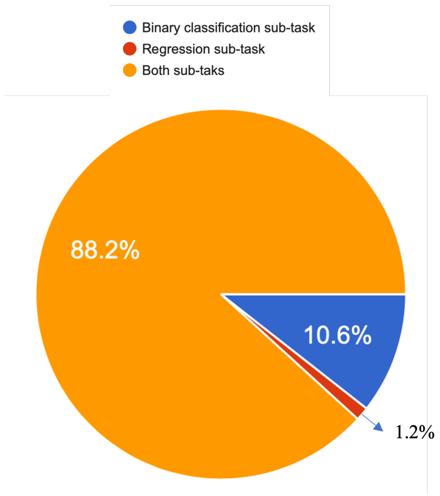


Figure 1: Statistics of participants' interest on Tasks based on initial registration.

Task	sub-task1	sub-task2
Team Participated	21	17
Total System Submissions	134	110
Total Accepted Submissions	108	84

Table 4: Statistics on participation

Baselines For each sub-task a separate baseline were defined: *i)* for Sub-task1 – the binary classification sub-task, a Linear Support Vector classifier using n-grams (up to three) as input features was used, and for the *ii)* Sub-task2 – the regression sub-task, a baseline using a Linear Support Vector regressor with the same n-grams features was provided.

We provided the starter code to all the participants, along with different cross-validation configurations that we encouraged participants to use to validate their methodology. Moreover, we provided

[4]All correlations are significant with p value< 0.01.

[5]https://codalab.lisn.upsaclay.fr/competitions/1292

[6]https://codalab.lisn.upsaclay.fr/competitions/1290

information on how the performance in the validation task translated in the official test split (applying the baseline methods to the official test-set yielded results 10-20% lower than with the training set). We highlighted the importance of achieving maximal syntactic generality on this task and test different cross-validation configurations on the training set.

Official Evaluation Metrics Given the differences in the nature of the sub-tasks output, we defined two different sets of evaluation metrics. For sub-task1, systems are evaluated with respect to binary and macro F-measure. These measures were evaluated per languages, and the final ranking was based on the global ranking of each participant, calculated by averaging the macro F-measure score from all the three languages (this provided an incentive to give results for all languages). In addition to the official measure, we also gave the participants their precision and recall scores, per language.

As for sub-task2, a Spearman's rank correlation coefficient (ρ) between the task participants' scores and the test set scores was computed. To be consistent with sub-task1, the global ranking of this task was calculated by averaging the position of the participant's ρ per language.

At the end of the competition, we provided the participants with packages containing the results for each of their submissions, and publicly updated the leader-board with ranks listing all teams who competed in each sub-task.

Participation The task attracted nearly 83 teams. Among them, 43 teams actually registered for the evaluation phase; 21 teams (sub-task1) and 17 teams (sub-task2) submitted their system's predictions. The detailed statistics are shown in Table 4.

4 Participating Systems

We received six system description papers for both sub-tasks, plus two papers by teams that participated only in sub-task1, for a total of eight papers. As it can be observed in the following summaries of the approaches proposed, there were several points of methodological similarity, but also interesting differences. Many teams experimented data augmentation techniques devised to overcome the limited amount of training data, in particular for the solution of sub-task2. These techniques ranged from the use of external resources to the generation of new sentences (Zhou et al. (2022), Sarhan et al. (2022) and van den Berg et al. (2022)), to the automatic translations across the three languages considered (Sarhan et al. (2022) and Zhou et al. (2022)), to mapping the Likert scale results to the binary values. This was the strategy used by the first two teams classified (Xia et al. (2022) and Li et al. (2022)) according to the global scores.

As we can see by the description of the participating systems (see Section 4.1), the majority of teams chose monolingual instead of multilingual models, especially in the resolution of sub-task1. The exceptions are represented by Li et al. (2022), who obtained the second position in the global ranking of both sub-tasks, by Aziz et al. (2022), and by Sarhan et al. (2022) who (in the resolution of sub-task 2 only), used the multilingual version of the Universal Sentence Encoder, since it yielded better performance than the monolingual one. Interestingly enough, the top-ranked team in both sub-tasks (Xia et al. (2022)) found that, for all languages, the monolingual DeBERTa-v3 models always outperformed the multilingual version.

A further approach shared by the participating teams is represented by the adoption of ensemble methods. Two main ensemble strategies were suggested. In the first one, the training data used to fine-tune the adopted model was split, obtaining different models, each with its training and validation sets. This was the case with the second-ranked system (Li et al. (2022) and with Vetter et al. (2022), but only in sub-task2. A second main approach used the fusion of the acceptability scores predicted by two different models. As described in the following subsection, Aziz et al. (2022) combined the scores predicted by XLM-RoBERTa (Conneau et al., 2019) and mBERT (Devlin et al., 2019), while Zhou et al. (2022) merged the predictions made by ERNIE-M and DeBERTa-v3 (only in sub-task1).

4.1 Individual System Descriptions

Xia et al. (2022) (model LingJing), tackling sub-task1, experimented with different strategies to fine-tune DeBERTa-v3 (He et al., 2021), which ended up outperforming both the PreTENS baseline and three new baselines introduced by the authors, including a multilingual version of DeBERTa, i.e. mDeBERTa model. These strategies included the augmentation of the original training set with trans-

Team/user name	Global score	Rank	ENG	Rank	FRE	Rank	ITA	Rank
LingJing⋆	94.49	1	97.17	1	93.24	1	93.05	1
HW-TSC⋆	92.80	2	93.04	5	93.01	2	92.34	2
UU-TAX⋆	91.57	3	93.08	4	89.53	4	92.12	3
CSECU-DSG⋆	91.12	4	91.51	7	90.73	3	91.11	4
piano	90.74	5	97.12	2	86.14	11	88.95	6
SPDB Innovation Lab⋆	89.58	6	94.55	3	87.28	7	86.90	8
bpc	89.09	7	91.36	8	88.32	6	87.59	7
weijiyao	88.78	8	92.15	6	87.28	8	86.90	9
ddd7788	86.68	9	80.44	13	88.86	5	90.75	5
cnxupupup	86.68	10	86.88	9	86.39	10	86.76	10
MaChAmp	86.42	11	86.58	10	86.52	9	86.17	12
aidenqiu	86.30	12	86.29	11	86.09	12	86.51	11
UoR-NCL⋆	80.32	13	77.23	16	80.08	14	83.65	13
RUG-1-pegasussers⋆	79.56	14	80.31	14	79.71	15	78.64	14
KaMiKla⋆	77.99	15	77.21	17	82.34	13	74.40	15
Huawei-zhangmin	71.80	16	78.54	15	65.77	18	71.08	16
RCLN	70.54	17	73.02	18	75.73	16	62.86	17
BASELINE	67.39	18	70.47	19	72.13	17	59.59	18
Jan/Jasper/Boris	27.26	19	81.76	12	–	–	–	–
folkertleistra	22.64	20	67.92	20	–	–	–	–
RUG-3	19.95	21	59.85	21	–	–	–	–

Table 5: Sub-task 1 results for each team/user ordered by overall F1-Macro along with micro-averages for each language. Team/user names marked with ⋆ have submitted their system description.

lations from the three languages, adversarial training and Child-Tuning (Xu et al., 2021). In addition, the authors performed experiments with different compositions of the original training, i.e. mixing the data for the three languages, and fine-tuning in one language, then expanding to the others. Each strategy achieved different results for each language. Due to the small size of the training set of sub-task2, the team transferred the knowledge of the classification model to the regression task, in terms of the model's parameters and in the idea of mapping the Likert scale results to the binary values.

Li et al. (2022) (HW-TSC), addressing both subtasks1 and 2, developed an ensemble classification and regression model by fine-tuning the multilingual XLM-RoBERTa model (Conneau et al., 2019) on different splits of the training data. To this end, they added a language tag to each training sentence with the same id across the three languages and divided the data in different folds to prevent the model from learning the translation information. To address the small size of the sub-task2 training data, they devised a data augmentation strategy to transform the binary values into the scalar human judgments.

Sarhan et al. (2022) (UU-TAX), experimented with different Neural Language Models and diverse training data compositions to test their generalization abilities against the PreTENS tasks. For sub-task1, their best performing model is represented by ELECTRA (Clark et al., 2020), which was fine-tuned using a two-stage strategy to augment the original training data. Firstly, the authors generated new sentences by making modifications to the original sentences using BERT-base (Devlin et al., 2019) to obtain the embeddings of the modified words. Secondly, for each language *l*, the original sentences of the other two languages were translated into *l* using the Google Translate API. For sub-task2, the best model uses the multilingual version of the Universal Sentence Encoder (Yang et al., 2020) followed by a different type of classifier for each language.

Aziz et al. (2022) (CSECU-DSG), for both subtasks, exploited an ensemble method of two multilingual Transformers, i.e. XLM-RoBERTa (Conneau et al., 2019) and mBERT (Devlin et al., 2019), which were fine-tuned with the PreTENS datasets. To enhance the performance of each individual model, the authors fused the predicted probability scores of the two models by computing their weighted arithmetic mean.

Vetter et al. (2022) (KaMiKla), for both subtasks, used monolingual versions of the BERT model (Devlin et al., 2019), i.e. BERT base for English, AlBERTo (Polignano et al., 2019) for Italian and CamemBERT (Martin et al., 2020) for French. For sub-task1, the authors fine-tuned the models on the distributed training data, while for sub-task2,

Team/user name	Global score	Rank	ENG	Rank	FRE	Rank	ITA	Rank
LingJing⋆	0.802	1	0.758	1	0.841	1	0.807	1
HW-TSC⋆	0.757	2	0.706	2	0.805	2	0.759	2
Huawei-zhangmin	0.669	3	0.636	3	0.74	3	0.631	3
BASELINE	0.309	4	0.265	6	0.317	4	0.344	4
UU-TAX⋆	0.221	5	0.478	4	-0.062	15	0.246	5
daydayemo	0.206	6	0.212	8	0.284	5	0.121	9
aidenqiu	0.205	7	0.211	9	0.284	6	0.121	10
CSECU-DSG⋆	0.16	8	0.191	10	0.081	8	0.207	6
RCLN	0.139	9	0.418	5	-0.005	9	0.006	14
folkertleistra	0.123	10	0.232	7	0.102	7	0.036	13
KaMiKla⋆	0.078	11	0.059	15	-0.01	10	0.186	7
xxxyyyxxx	0.074	12	0.094	14	-0.013	11	0.14	8
UoR-NCL⋆	0.056	13	0.122	13	-0.043	13	0.089	12
RUG-3	0.046	14	0.137	12	–	–	–	–
suzuki	0.042	15	0.14	11	-0.018	12	0.003	15
akkhan1871	0.008	16	-0.006	16	-0.06	14	0.09	11
MaChAmp	-0.164	17	-0.131	17	-0.195	16	-0.167	16

Table 6: Sub-task 2 results for each team/user ordered by overall ρ along with results for each language. Team/user names marked with $\star$ have submitted their system description.

they first normalized the scores to be between zero and one, then performed an inverse transformation to get the final output. In addition, for this second task, they trained 10 models per language (each with its one training split) and used the median result as their final prediction.

Zhou et al. (2022) (SPDB) participated only in sub-task1, using a different ensemble system for each language. For Italian and French, the system combines the results of 10 ERNIE-M models (Ouyang et al., 2021) obtained by applying a cross-validation process; for English, the authors combined the predictions made by ERNIE-M and DeBERTa-v3 (He et al., 2021). They also enlarged the distributed PreTENS training set using *i)* two different translators (Google and Baidu) to translate the English sentences into French and Italian, thus increasing the diversity of data, and *ii)* the English and French version of the XNLI dataset (Conneau et al., 2018).

van den Berg et al. (2022) (RUG-1-pegasussers) participated only in sub-task1 using English BERT base (Devlin et al., 2019) which they fine-tuned, experimenting with multiple approaches to expand the training data. In particular, they augmented the data by adding new English sentences that contained new category templates, new words instantiating the templates, new words previously used exclusively as hyponyms, inversions of the arguments involved in the taxonomical relation, paraphrases automatically generated. The acceptability labels of Italian and French sentences were predicted by translating the sentences into English, in order to process them with the English BERT.

Markchom et al. (2022) (UoR-NCL), for both sub-tasks, experimented with fine-tuning different monolingual versions of the BERT-based model (Devlin et al., 2019), i.e. DistilBERT-Base-Uncased for English (Sanh et al., 2019), BERT-Base-Italian-XXL-Uncased for Italian, FlauBERT-Base-Uncased for French. The authors relied on the distributed training data to fine-tune the models using specific loss functions, binary cross-entropy loss for sub-task1 and mean squared error loss for sub-task2.

5 Results and Discussion

Almost all teams submitted their runs for the three languages considered. Tables 5 and 6 show, for each sub-task, the top submissions received from each team, along with the baseline scores. Team names marked with $\star$ represent teams that have submitted system description papers. The *Rank* column reports the position of the team in the ranking for global and language-specific scores.

Task 1 evaluation: To better understand the performances per construction of the models submitted, we report the average F1-macro ($\pm std$) of the top-3 submissions per language, in Table 7.

Our results show that English — the most resource-rich language in terms of computational models and data — outperforms the Italian and French models for correctly predicting presuppositions in these constructions. However, the English model performed below French in the UNLIKE construction (e.g. "Unlike trees, {*oaks /

CONSTRUCTION	EN	FR	IT	Avg_lang
DRATHER	**94.9** (±0.04)	90.9 (±0.03)	89.3 (±0.03)	91.70
COMPARATIVES	**94.2** (±0.05)	87.4 (±0.03)	88.2 (±0.02)	89.93
EXCEPT	**88.8** (±0.09)	88.3 (±0.05)	84.1 (±0.05)	87.07
UNLIKE	87.4 (±0.07)	**88.4** (±0.05)	84.6 (±0.05)	86.80
BUTNOT	**89.3** (±0.07)	78.0 (±0.15)	81.5 (±0.03)	82.93
PREFER	**86.5** (±0.1)	83.5 (±0.05)	78.1 (±0.01)	82.70
ANDTOO	**84.4** (±0.13)	74.6 (±0.14)	77.5 (±0.0)	78.83
PARTICULAR	**94.3** (±0.04)	45.3 (±0.0)	86.7 (±0.04)	75.43
TYPE	**75.8** (±0.14)	66.8 (±0.08)	72.2 (±0.12)	71.60
GENERALLY	45.5 (±0.0)	**75.2** (±0.12)	46.7 (±0.01)	55.80

Table 7: Average macro F-measure of the top 3 participants per construction in sub-task 1 (binary classification). The standard deviation between the top 3 submission are in (.). Best results per construction are in bold.

animals} are often mentioned in this text", presupp. A1$\not\leq$A2), and does quite poorly in GENERALLY ("I like oaks, and more generally {trees / *animals}", presupp. A1<A2). Note that neither constructions were present in the training set. Italian is aligned with English, while interestingly the French model seems to be capable of more accurate generalizations in both cases.

Task 2 evaluation: To gain a better understanding of the models generalization abilities in this task, we computed the Root Mean Squared Error (RMSE) between the gold value and the average predicted value by the first three teams classified. This data is shown in Figure 2 for each construction and for each language, along with the average value across languages. As it can be seen, the resulting picture contrasts substantially with that of sub-task1. The TYPE and PARTICULAR constructions, among the worst in the first sub-task, have the lowest error in sub-task2. The second task sees a substantial drop of ANDTOO (now the worst case) and COMPARATIVES (one of the best constructions in sub-task1).

We also observe distinctions between languages across the two tasks. In particular, while the average performance for English across constructions is the highest in sub-task1, the French models obtained on average the best results in sub-task2. The success at predicting the presuppositional knowledge triggered by the same construction changes in the two sub-tasks (which are often demanded to different models in the various teams). For example, the French models are the best at classifying the acceptability label for the UNLIKE construction but are the worst in predicting the human score for the same construction. Conversely, COMPARATIVES turn out to be among the easiest constructions for the English models in sub-task1 but are the most mispredicted English type in sub-task2.

Quite interestingly, the presence of a construction in training doesn't always guarantee better performances. Two notable examples are represented by DRATHER and PARTICULAR. Despite being absent in the training set of the corresponding tasks, the first obtains the highest F1 score (for English and on average) and the second is among the top-predicted constructions in sub-task2. We leave a more thorough analysis of the systematicity of these trends to future work, where we will also consider the linguistic variants for each construction and the semantic categories of the nominal arguments involved.

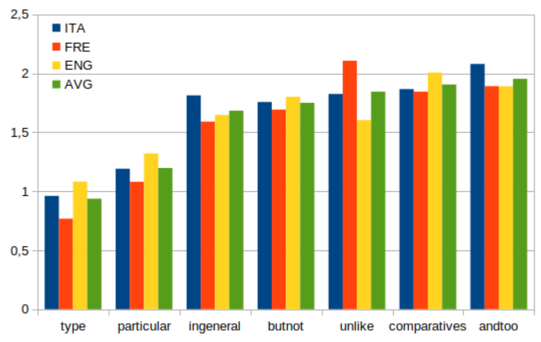


Figure 2: sub-task2: Root Mean Squared Error (RMSE) averaged across the top 3 participants per construction (lower is better). Constructions are ordered per average RMSE values across languages.

6 Conclusion

The SemEval2022 task3 – PreTENS offered two sub-tasks aiming to investigate the effectiveness of computational models to detect a certain type of presuppositional failures induced by specific constructions. The task attracted a total of 21 teams, from both academia and industry. The findings showcases the power and ubiquity of large self-supervised pre-trained models in mono- or multilingual settings. Despite this apparent uniformity, the participants chose to use and combine the models in very different and creative ways, giving rise to a range of scores from 70.54% to 94.49%.

The outcomes of the task highlights the ability of these transformer models to generalize to new/unknown construction in the test sets, but also the presence of intriguing differences in specific constructions and languages (e.g. in the binary task *A1 and more generally A2* reaches a 75.2 F-measure in French but a 46.7 in Italian). Also worth further investigation is the lower correlation between the binary judgments and the human ratings in English — probably reflected in the .05 drop seen in the sub-task2 global score for this language, compared to French.

The success of the task indicates a growing interest towards research on prediction models that can incorporate world knowledge and common sense, along with an understanding of the linguistic properties that condition the outcomes. We hope that this trend will continue and the PreTENS data will help researchers to probe future models for this ability. With this spirit, we make the dataset public.

References

Abdul Aziz, Md. Akram Hossain, and Abu Nowshed Chy. 2022. CSECU-DSG at SemEval-2022 Task 3: Investigating the taxonomic relationship between two arguments using fusion of multilingual transformer models. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Frank van den Berg, Gijs Danoe, Esther Ploeger, Wessel Poelman, Lukas Edman, and Tommaso Caselli. 2022. RUG-1-pegasussers at SemEval-2022 Task 3: Data generation methods to improve recognizing appropriate taxonomic word relations. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. Semeval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *Proceedings of the 10th international workshop on semantic evaluation (*SEMEVAL-2016*)*, pages 1081–1091.

Shammur Absar Chowdhury and Roberto Zamparelli. 2018. RNN simulations of grammaticality judgments on long-distance dependencies. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 133–144. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceedings of International Conference on Learning Representations*, page OpenReview.ne.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

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

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

R. Futrell, E. Wilcox, T. Morita, P. Qian, M. Ballesteros, and R. Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. In *Proceedings of the 2019 Conference of the North American Chapter of the ACL: Human Language Technologies*, volume 1, Minneapolis, Minnesota. Association for Computational Linguistics.

K. Gulordava, P. Bojanowski, E. Grave, T. Linzen, and M. Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of NAACL, HLT 2018 (16th Annual Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies)*, pages 1195–1205, East Stroudsburg, PA. ACL.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. arXiv:2111.09543. Version 2.

Jaap Jumelet and Dieuwke Hupkes. 2018. Do language models understand anything? On the ability of LSTMs to understand negative polarity items. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 222–231, Brussels, Belgium.

Yinglu Li, Min Zhang, Xiaosong Qiao, Minghan Wang, Hao Yang, Shimin Tao, and Ying Qin. 2022. HW-TSC at SemEval-2022 Task 3: A unified approach fine-tuned on multilingual pretrained model for PreTENS. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Tal Linzen and Marco Baroni. 2020. Syntactic structure from deep learning. arXiv: 2004.10827.

Thanet Markchom, Huizhi Liang, and Jiaoyan Chen. 2022. UoR-NCL at SemEval-2022 Task 3: Fine-tuning the BERT-Based models for validating taxonomic relations. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE-M: Enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 27–38, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

M. Polignano, P. Basile, M. de Gemmis, G. Semeraro, and V. Basile. 2019. AlBERTo: Italian BERT language understanding model for nlp challenging tasks based on tweets. In *roceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it)*, pages volume 2481, CEUR.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Injy Sarhan, Pablo Mosteiro, and Marco Spruit. 2022. UU-TAX at SemEval-2022 Task 3: Improving the generalizability of language models for taxonomy classification through data augmentation. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Shane Storks, Qiaozi Gao, and J. Chai. 2019. Recent advances in natural language inference: A survey of benchmarks, resources, and approaches. *arXiv: Computation and Language*.

Ke Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of being recurrent for modeling hierarchical structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4731–4736, Brussels, Belgium. Association for Computational Linguistics.

Karl Vetter, Miriam Segiet, and Klara Lennermann. 2022. KaMiKla at SemEval-2022 Task 3: AlBERTo, BERT, and CamemBERT—Be(r)tween taxonomy detection and prediction. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP@EMNLP*.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020. Semeval-2020 task 4: Commonsense validation and explanation. ArXiv preprint arXiv:2007.00236.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2019. Blimp: A benchmark of linguistic minimal pairs for english. *CoRR*, abs/1912.00582.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do RNN language models learn about filler-gap dependencies? In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221, Brussels. ACL.

Fei Xia, Bin Li, Yixuan Weng, Shizhu He, Bin Sun, Shutao Li, Kang Liu, and Jun Zhao. 2022. LingJing at SemEval-2022 Task 3: Applying DeBERTa to lexical-level presupposed relation taxonomy with knowledge transfer. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9514–9528, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Yue Zhou, Bowei Wei, Jianyu Liu, and Yang Yang. 2022. SPDB Innovation Lab at SemEval-2022 Task 3: Recognize appropriate taxonomic relations between two nominal arguments with ERNIE-M model. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

LingJing at SemEval-2022 Task 3: Applying DeBERTa to Lexical-level Presupposed Relation Taxonomy with Knowledge Transfer

Fei Xia[1,2]*, Bin Li[3]*, Yixuan Weng[1]*, Shizhu He[1,2],
Bin Sun[3], Shutao Li[3], Kang Liu[1,2], Jun Zhao[1,2]
[1] National Laboratory of Pattern Recognition Institute of Automation, CAS
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences
[3] College of Electrical and Information Engineering, Hunan University
{libincn, shutao_li, sunbin611}@hnu.edu.cn, wengsyx@gmail.com,
xiafei2020@ia.ac.cn, {shizhu.he, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

This paper presents the results and main findings of our system on SemEval-2022 Task 3 Presupposed Taxonomies: Evaluating Neural Network Semantics (PreTENS)[1]. This task aims at semantic competence with specific attention on the evaluation of language models, which is a task with respect to the recognition of appropriate taxonomic relations between two nominal arguments. Two sub-tasks including binary classification and regression are designed for the evaluation. For the classification sub-task, we adopt the DeBERTa-v3 pre-trained model for fine-tuning datasets of different languages. Due to the small size of the training datasets of the regression sub-task, we transfer the knowledge of classification model (i.e., model parameters) to the regression task. The experimental results show that the proposed method achieves the best results on both sub-tasks. Meanwhile, we also report negative results of multiple training strategies for further discussion. All the experimental codes are open-sourced at https://github.com/WENGSYX/Semeval.

1 Introduction

In order to dive into the capability of the current language models (Bengio et al., 2000; Howard and Ruder, 2018), we take part in and apply the pre-training language model on the SemEval-2022 Tasks 3: the Presupposed Taxonomies: Evaluating Neural Network Semantics (PreTENS) (Zamparelli et al., 2022). To evaluate the model performance comprehensively, two sub-tasks are designed in this PreTENS task:

Sub-task 1) Binary classification task[2], which consists in predicting the acceptability label assigned to each sentence of the test set. For example, "I like trees, and in particular birches" is acceptable while " I like oaks, and in particular trees" is unacceptable, so they are labeled 1 and 0, respectively.

Sub-task 2) Regression sub-task[3], which consists in predicting the average score assigned by human annotators on a seven-point Likert scale (Joshi et al., 2015) with respect to the subset of data evaluated via crowdsourcing. For example,"I like governors, an interesting type of politician" is more acceptable than "I like politicians, an interesting type of farmer", so the former will also have a higher score (6.16) than the latter (1.42).

It is noted that both sub-tasks comprise datasets in 3 languages: English, Italian, French, where French and Italian are slightly adapted translations of the English dataset. For each sub-task, every sample is formed as the arguments A and B, e.g., comparatives (I like A more than B), exemplifications (I like A, and in particular B), generalizations (I like A, and B in general), and others, where the argument nouns are taken from various semantic categories.

The most similar tasks are the Natural Language Inference (MacCartney, 2009; Bowman et al., 2015; Conneau et al., 2017) and Taxonomy Expansion & Enrichment (Zhang et al., 2018; Shen et al., 2018; Yu et al., 2020), where the former requires the model to differentiate the relationship between a premise sentence and a hypothetical sentence, while the latter shall identify the relationship between different concepts. There are many powerful language models for accomplishing these tasks (Devlin et al., 2018; He et al., 2020), and well-formed semantic representations can be obtained for the downstream tasks. However, it is a challenge for the model to decide the acceptance of the given sentence, as the semantic meaning is hard to be distinguished. Moreover, the performance of downstream tasks when fine-tuning is limited by the size of the training dataset (Xie et al., 2020).

To solve the above problems, we propose a

*These authors contribute equally to this work.
[1]https://sites.google.com/view/semeval2022-pretens/
[2]https://codalab.lisn.upsaclay.fr/competitions/1292
[3]https://codalab.lisn.upsaclay.fr/competitions/1290

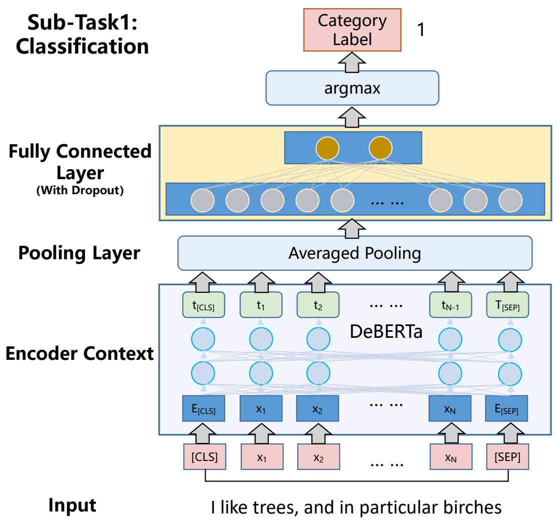
Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 239 - 246
July 14-15, 2022 ©2022 Association for Computational Linguistics

method that applies DeBERTa to lexical-level presupposed relation taxonomy with knowledge transfer. Specifically, the powerful DeBERTa-v3 pre-trained model (He et al., 2021) is fine-tuned with datasets of different languages in the classification task. For the regression task, due to the limited size of the training datasets, we fine-tune the trained model of the classification task in the regression task. As a result, the proposed method achieves a global top score of 94.173 in sub-task 1 and a global top score of 0.802 in sub-task 2. Our method wins on two sub-tasks. In addition, we present the negative results of multiple training strategies when fine-tuning and provide further discussions.

2 Main method

In this section, we will elaborate on the main methods for the two sub-tasks of the PreTENS task. The training strategies are included at the end of this section.

2.1 Sub-task 1 - Binary classification using DeBERTa

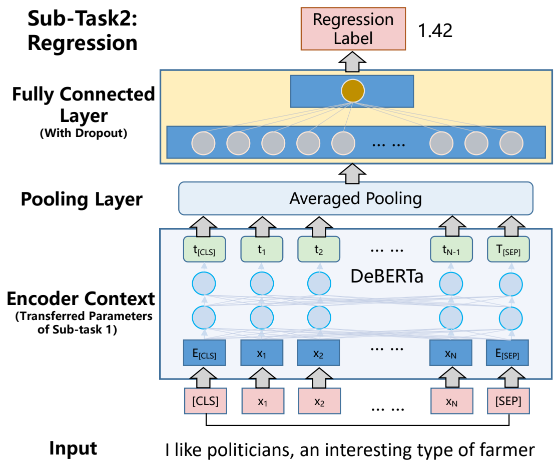


Sub-Task1: Classification
Category Label
1
argmax
Fully Connected Layer (With Dropout)
Pooling Layer
Averaged Pooling
Encoder Context
DeBERTa
Input
I like trees, and in particular birches

Figure 1: Main structure of the method in sub-task 1.

Sub-task 1 is a classic classification task, where two acceptability labels are required to be classified. We adopt the DeBERTa-v3 (He et al., 2021) model for processing this binary classification, where the main method structure is shown in Figure 1. The given sentence is separated into tokens and then sent to the pre-trained model as the input. To obtain the complete meaning of the whole sentence, we take the output embedding of each token to be averaged by the averaged pooling layer. The binary classification task is designed by sending the averaged encoding into the fully connected layer with dropout.

2.2 Sub-task 2 - Regression with knowledge transferring



Sub-Task2: Regression
Regression Label
1.42
Fully Connected Layer (With Dropout)
Pooling Layer
Averaged Pooling
Encoder Context (Transferred Parameters of Sub-task 1)
DeBERTa
Input
I like politicians, an interesting type of farmer

Figure 2: Main structure of the method in sub-task 2.

Sub-task 2 is a regression task, where a seven-point Likert-scale which ranges from 1 (not at all acceptable) to 7 (completely acceptable) is used to perform the regression. There is a subset of 1,533 sentences of the entire dataset for this regression task, where the total number is relatively small. It is a wise choice to transfer the knowledge from the pre-trained model of sub-task 1 into the regression task. The reason is that the fine-tuned model of sub-task 1 learns quite a few patterns of sentence acceptance that come from the extra knowledge. As shown in Figure 2, the DeBERTa-v3 model trained in sub-task 1 is designed to perform the regression task. The fully connected layer with dropout for obtaining the output logits is used for the regression task.

2.3 Multiple training strategies

In this section, we will introduce some training strategies used in the competition, which includes data augmentation with translation, adversarial training and child-tuning training.

2.3.1 Data augmentation with translation

When fine-tuning the English datasets, we translate the training sample of the Italian and the French to English one by one based on the M2M-1.2B model (Ott et al., 2019; Fan et al., 2020). It is a process that provides more useful datasets from the same

Example	Sample	Label
Classification	I like trees, and in particular birches	1
	I like oaks, and in particular trees	0
Regression	I like politicians, an interesting type of farmer	1.42
	I like governors, an interesting type of politician	6.16

Table 1: The dataset sample example.

resources. As a result, the DeBERTa model fine-tuned the datasets in English may achieve better results.

2.3.2 Adversarial training

The common method in adversarial training is the Fast Gradient Method (FGM). The idea of the FGM (Miyato et al., 2016) is straightforward[4]. The loss is to increase the gradient so that we can take

$$\Delta x = \epsilon \nabla_x L(x, y; \theta) \tag{1}$$

where x represents the input, y represents the label, θ is the model parameter, $L(x, y; \theta)$ is the loss of a single sample, Δx is the anti-disturbance. To prevent Δx from being too large, it is usually necessary to standardize $\nabla_x L(x, y; \theta)$. The more common way is

$$\Delta x = \epsilon \frac{\nabla_x L(x, y; \theta)}{\|\nabla_x L(x, y; \theta)\|}. \tag{2}$$

2.3.3 Child-tuning training

We use the Child-tuning (Xu et al., 2021) for fine-tuning the pre-trained model and only update the parameters of the Child network through gradients mask. For the two sub-tasks, the task-independent algorithm is used. In the process of fine-tuning, the gradients mask is obtained by sampling from the Bernoulli distribution (Chen and Liu, 1997) in each step of iterative update, which is equivalent to randomly dividing a part of the network parameters when updating. The equation of the above steps is shown as follows

$$\begin{aligned} \mathbf{w}_{t+1} &= \mathbf{w}_t - \eta \frac{\partial \mathcal{L}(\mathbf{w}_t)}{\partial \mathbf{w}_t} \odot M_t \\ \boldsymbol{M}_t &\sim \text{ Bernoulli } (p_F). \end{aligned} \tag{3}$$

3 Experimental setup

3.1 Data description

The PreTENS dataset not only needs to judge the classification relationship between two nouns

[4]https://spaces.ac.cn/archives/72

Datasets	Classification Task		Regression Task	
Language	Train	Test	Train	Test
English	5837	14560	524	1009
French	5837	14560	524	1009
Italian	5837	14560	524	1009

Table 2: Number statistics of task dataset samples.

(Wang et al., 2017), but also needs to identify whether the two nouns are in line with the actual situation in the artificially constructed natural sentences. The argument nouns are taken from 30 semantic categories (e.g., dogs, birds, mammals, cars, motorcycles...).

Specifically, PreTENS is articulated into the two following sub-tasks. The classification task requires judging the acceptability of the samples. The training set and test set contain 5838 and 14556 samples. Regression Sub-task obtains scores from 1 (not at all acceptable) to 7 (completely acceptable) through human crowdsourcing, which could be affected by usability considerations, argument order, and other factors. The data set of the regression sub-task is a small amount, 524 sentences will be provided for the training set and 1,009 for the test set. The example of two sub-task datasets is shown in Table 1, and the number statistics of each sub-task is shown in Table 2.

3.2 Evaluation metrics

For classification tasks, the official evaluation indicators include precision, recall, macro F1, and global score. The global score is the average value of macro F1.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 Precision Recall}{Precision + Recall}$$

$$MacroF1 = \frac{\sum_{i=1}^{n} F1_i}{n}$$

$$Global = \frac{F1(En) + F1(Fr) + F1(It))}{3}$$

where n means that the higher the total number of categories, accuracy, recall rate, and macro F1. The higher F1 the method, the better the performance.[5]

For regression, it is mainly evaluated by MSE, RMSE and Spearman correlation (rho) (Wissler, 1905).

$$MSE = \frac{1}{n}\Sigma_{i=1}^{n}\left(\frac{R_i - Q_i}{\sigma_i}\right)^2$$

$$RMSE = \sqrt{MSE}$$

$$RHO = 1 - \frac{6\sum_{i=1}^{n}(R_i - Q_i)^2}{n(n^2-1)}$$

where the paired values of two variables are ranked from small to large (or from large to small). R_i represents the rank of x_i, Q_i represents the rank of y_i, and R_i - Q_i is the difference between the ranks of x_i and y_i.

3.3 Baselines introduction

3.3.1 Binary classification sub-task

N-gram method The original baseline provided by the organizers is based on the n-grams (Broder et al., 1997), where n=3. A Linear Support Vector (Tang, 2013) classifier using n-grams (set to 3) as input features is used for the binary classification.

mDeBERTa model The mDeBERTa (He et al., 2021) is a multilingual version of DeBERTa (He et al., 2020) which uses the same structure as DeBERTa and was trained with CC100 multilingual data (Wenzek et al., 2020; Conneau et al., 2020). The mDeBERTa model comes with 12 layers and a hidden size of 768. It has 86M backbone parameters with a vocabulary containing 250K tokens which introduce 190M parameters in the Embedding layer. This model was trained using the 2.5T CC100 data as XLM-R (Conneau et al., 2019).

3.3.2 Regression sub-task

N-gram method This method is provided by the organizers, where it uses a Linear Support Vector regressor with the 3-grams features is provided for the regression sub-task.

mDeBERTa model We adopt the mDeBERTa (He et al., 2021) for processing the sub-task 2. The architecture of this method is the same as Figure 2, where the linear layer over in the mDeBERTa is initialized before fine-tuning. A clamped step is performed for obtaining the final results of the regression.

3.4 Implementation details

We train the model based on the PyTorch (Paszke et al., 2019) and use the hugging-face (Wolf et al., 2020) framework. During training, we employ the AdamW optimizer (Loshchilov and Hutter, 2017). The default learning rate is set to 1e-5 with the warm-up (He et al., 2016). Four RTX3090 GPUs are implemented for all experiments. There are some variants of the DeBERTa-v3 model, i.e., base and large model. We adopt DeBERTa-v3-large models as our backbone, where the batch size is set to 24, and the max length of input is set to 64. We train our backbone for 6 epochs, and save the model parameters at the end of each round. We test the saved checkpoints in the evaluation phase, and select the highest score as the experimental result.

For the sub-task 1, we will implement the overall training (by mixing the datasets in different languages for training), and separate training (to fine-tune in one language and expand to the others), and conduct experiments on the training strategies such as FGM, data augmentation with translation and Child-tuning.

For the sub-task 2, we map the result of [1,7] in the label space to the minimum value of [0,1]. The results of the regression model are clamped so that the minimum value is 0 and the maximum value is 1. Moreover, we will compare the performance of models which use knowledge transfer or not.

4 Results and discussions

In this section, we studied the experimental results of the two sub-tasks and the impact of different strategies on the results, and further discussed the comparison with other participants to prove the effectiveness of the method. Finally, Studies analyze the deviation of language model and the future research direction.

4.1 Experimental results

4.1.1 Classification sub-task

Model Selection We first carry out experiments on three different models when fine-tuning, namely

[5]Below is the specific meaning of the formula. TP: The prediction is correct and the sample is correct. FP: The prediction is wrong and the sample is correct. FN: The prediction is correct and the sample is wrong.

Experimental Items		English			French			Italian		
Method	Global	Precision	Recall	macro F1	Precision	Recall	macro F1	Precision	Recall	macro F1
N-gram	72.34	64.19	85.64	73.38	65.07	89.92	75.50	55.71	87.73	68.15
mDeBERTa	88.58	83.33	95.30	88.92	82.72	93.33	87.71	83.38	95.67	89.11
DeBERTa-v3-base	72.88	75.66	69.91	75.03	83.60	75.18	81.16	58.60	69.02	62.45
DeBERTa-v3-large	92.90	88.10	95.83	91.94	92.31	93.89	93.42	93.11	93.48	93.35
DeBERTa-v3-large in English	92.24	96.18	**98.57**	**97.48**	84.38	**97.14**	90.19	82.36	**97.11**	89.06
DeBERTa-v3-large in French	89.33	85.12	96.35	90.36	85.23	86.79	86.67	88.03	93.58	90.98
DeBERTa-v3-large in Italian	91.78	**96.88**	89.20	93.50	92.80	82.75	88.72	**94.22**	91.06	93.12
DeBERTa-v3-large in En. and Fr.	93.21	92.18	96.73	94.59	**95.34**	92.26	94.21	92.31	87.95	90.82
DeBERTa-v3-large + FGM	88.32	83.72	97.61	89.93	81.11	96.09	87.62	80.85	96.05	87.42
DeBERTa-v3-large + Translation	92.75	92.93	96.66	94.96	88.25	94.06	91.30	89.04	94.67	92.00
DeBERTa-v3-large + Child-tuning	94.41	91.57	97.74	94.70	93.26	94.93	**94.37**	92.84	94.99	**94.18**
Ours	**95.34**	96.18	**98.57**	**97.48**	93.26	94.93	**94.37**	92.84	94.99	**94.18**

Table 3: Main experimental results of the sub-task1. From top to bottom, the first four lines are the comparison between different pre-training models, and then the best model DeBERTa-large is selected as the subsequent fine-tuning model. The four lines in the middle represent the way to use separate or overall datasets for training, not all data sets. The last three lines are some training strategies used for fine-tuning. We chose the highest score of all experiments under different data sets as "Ours" "Global Score".

Experimental Items	Original				Fine-Tuned			
Method	Global Score	RHO(EN)	RHO(FR)	RHO(IT)	Global Score	RHO(EN)	RHO(FR)	RHO(IT)
N-gram	0.309	0.265	0.317	0.344	/	/	/	/
mDeBERTa	0.430	0.412	0.350	0.529	0.720(+0.290)	0.670(+0.258)	0.783(+0.433)	0.708(+0.179)
DeBERTa-v3-base	0.108	0.216	-0.018	0.124	0.275(+0.167)	0.266(+0.050)	0.232(+0.250)	0.326(+0.202)
DeBERTa-v3-large	0.429	0.426	0.344	0.516	0.815(**+0.386**)	0.759(**+0.333**)	0.849(**+0.505**)	0.837(**+0.321**)

Table 4: Main experimental results of the sub-task2.

mDeBERTa, DeBERTa-v3-base, DeBERTa-v3-large. It can be found in the Figure 3 that the DeBERTa-v3-large model beats the N-gram method and surpass mDeBERTa in French and Italian. It shows that the larger model can bring improvements in the classification task.

Datasets Choosing We are more curious about the generalization ability of the DeBERTa model to fine-tune one language and then transfer to other languages. As a result, we fine-tune the datasets of separate and overall training in different languages. Although the performance by fine-tuning a single language is not satisfactory, it can be better than overall training. It is because there will be large gaps between different languages, and forced dataset mixing will reduce the final performance.

Training strategies After that, we compare some commonly used training strategies and the experimental results in this task, including FGM, translation, and Child-tuning. The performance of the FGM is not good, which indicates that the task pays more attention to the semantic features at the lexical level than the semantic features of sentences. All languages are translated into English, even better than using the original language. We believe that this is because the error accumulation caused by inaccurate translation will interfere with the semantic representation information. The result of Child-tuning is positive, which shows that the catastrophic forgetting problem of the model can be alleviated by eliminating some unimportant weights in the large model.

4.1.2 Regression sub-task

In our experiments of sub-task 2 shown in the Figure 4, we select three different methods for the experiments and compare them with the same experimental settings. The "Original" is the original pre-training model, the "Fine-Tuned" represents the model that is fine-tuned by sub-task1 and re-initialized the linear layer. The experimental results of three different methods show that the performance of the model can be greatly improved through knowledge transfer.

4.2 Official results

As shown in Table 5 and Table 6, our method achieved first place in subtask 1and subtask 2 and

Experimental Items		English			French			Italian		
System	Global	Precision	Recall	macro F1	Precision	Recall	macro F1	Precision	Recall	macro F1
Ours(LingJing)	**94.173**	97.65	96.34	**96.988**	93.15	92.48	**92.817**	91.68	93.77	**92.714**
YingluLi	92.310	93.29	91.89	92.582	93.34	91.77	92.546	92.94	90.69	91.802
injySarhanUU	91.247	90.54	95.26	92.839	85.83	93.14	89.335	92.69	90.47	91.567
piano	90.842	97.72	96.15	96.926	78.85	96.51	86.792	84.86	93.14	88.807
csecudsg	90.714	89.21	93.26	91.189	88.87	91.84	90.334	90.32	90.92	90.620
holdon	89.686	92.66	96.05	94.325	81.25	94.89	87.541	80.81	94.67	87.193

Table 5: System comparison on the three datasets of binary classification sub-task.

System	Global Score	RHO(EN)	RHO(FR)	RHO(IT)
Ours(LingJing)	**0.802 (1)**	**0.758 (1)**	**0.841 (1)**	**0.807 (1)**
qiaoxiaosong	0.757 (2)	0.706 (2)	0.805 (2)	0.759 (2)
huawei_zhangmin	0.669 (3)	0.636 (3)	0.740 (3)	0.631 (3)
injySarhanUU	0.221 (5)	0.478 (4)	-0.062 (16)	0.246 (5)
daydayemo	0.206 (6)	0.212 (8)	0.284 (5)	0.121 (9)
aidenqiu	0.205 (7)	0.211 (9)	0.284 (6)	0.121 (9)
Baseline	0.309 (4)	0.265 (6)	0.317 (4)	0.344 (4)

Table 6: System comparison on the three datasets of regression sub-task.

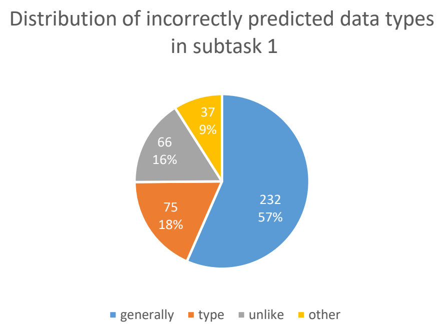


Figure 3: Case study in the sub-task1.

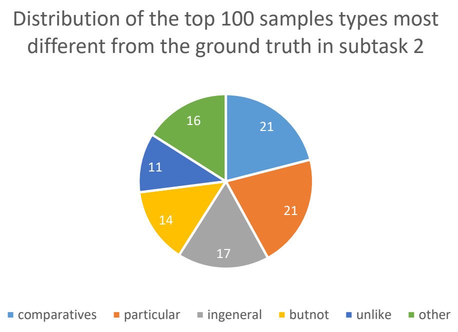


Figure 4: Case study in the sub-task2.

was substantially ahead of second place. Specifically, we earned first place in English, French and Italian in the subtask one classification task, with macroF1 values of 96.988%, 92.817% and 92.714%, respectively. Our global score of 94.173% was 1.863% above second place. For the Subtask 2 regression task, we also came first in English, French and Italian, with RHO scores of 0.758, 0.841 and 0.807, respectively. Our global rank score of Subtask 2 was 0.802, 0.045 above the second-place score.

4.3 Case studies

We counted and analyzed the mispredicted samples, and the distribution of error types is shown in Figures 3 and 4. For subtask 1, we select all the predicted error data for statistics. For subtask 2, we chose the top 100 samples with the most significant difference from the ground truth as the analysis object.

As we can see from Figure 3, the most mispredicted type in the classification task was “generally", with 57%, followed by“type" with 18% and “unlike" with 16%. Our analysis suggests that the reason for the incorrect predictions may be that “generally" sentences are less frequent in common usage and that our model did not have a large enough corpus of similar samples in the previous pre-training phase, thus leading to incorrect predictions.

From Figure 4, we can see that the top 100 data types with the greatest difference from the ground truth on the regression task are more evenly distributed, which means that the model migration is effective for subtask 2.

5 Conclusion

In this paper, we introduce the submitted system to the Semeval-22 task3 PreTENS. Based on the pre-training DeBERTa-v3, we carry out a simple and effective classification method in sub-task 1 and apply the method of knowledge transferring to sub-task 2. The proposed systems have won first place on both sub-tasks. The experimental results show that our proposed method has better performance than other methods. In addition, we also conducted a number of comparative experiments to further explore the difficulties of the PreTENS task. In the future, we will try to explore more effective methods to perform better semantic taxonomies.

6 Acknowledgement

This work is supported by the Natural Key R&D Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No. 61922085, No.61976211) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDBS-SSW-JSC006). This research work was supported by the independent research project of National Laboratory of Pattern Recognition, the Youth Innovation Promotion Association CAS and Yunnan Provincial Major Science and Technology Special Plan Projects (No. 202103AA080015).

References

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13.

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

Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig. 1997. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8):1157–1166. Papers from the Sixth International World Wide Web Conference.

Sean X Chen and Jun S Liu. 1997. Statistical applications of the poisson-binomial and conditional bernoulli distributions. *Statistica Sinica*, pages 875–892.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation.

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

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Ankur Joshi, Saket Kale, Satish Chandel, and D Kumar Pal. 2015. Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396.

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

Bill MacCartney. 2009. *Natural language inference*. Stanford University.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jiaming Shen, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle Vanni, Brian M. Sadler, and Jiawei Han. 2018. Hiexpan: Task-guided taxonomy construction by hierarchical tree expansion. In *Knowledge Discovery and Data Mining*.

Yichuan Tang. 2013. Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239*.

Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2017. A short survey on taxonomy learning from text corpora: Issues, resources and recent advances. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1190–1203, Copenhagen, Denmark. Association for Computational Linguistics.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Clark Wissler. 1905. The spearman correlation formula. *Science*, 22(558):309–311.

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

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*.

Yue Yu, Yinghao Li, Jiaming Shen, Hao Feng, Jimeng Sun, and Chao Zhang. 2020. Steam: Self-supervised taxonomy expansion with mini-paths. *arXiv: Computation and Language*.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. Semeval-2022 task3 (pretens): Evaluating neural networks on presuppositional semantic knowledge. In *Proceeding of SEMEVAL 2022*.

Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian M. Sadler, Michelle Vanni, and Jiawei Han. 2018. Taxogen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering. In *Knowledge Discovery and Data Mining*.

RUG-1-Pegasussers at SemEval-2022 Task 3: Data Generation Methods to Improve Recognizing Appropriate Taxonomic Word Relations

Frank van den Berg[*], **Gijs Danoe**[*], **Esther Ploeger**[*], **Wessel Poelman**[*]
Lukas Edman, Tommaso Caselli
Department of Information Science
University of Groningen
`{f.l.van.den.berg, g.danoe, e.ploeger.1, w.g.poelman}@student.rug.nl`
`{j.l.edman, t.caselli}@rug.nl`

Abstract

This paper describes our system created for the SemEval 2022 Task 3: Presupposed Taxonomies - Evaluating Neural-network Semantics. This task is focused on correctly recognizing taxonomic word relations in English, French and Italian. We develop various data generation techniques that expand the originally provided train set and show that all methods increase the performance of models trained on these expanded datasets. Our final system outperforms the baseline from the task organizers by achieving an average macro F1 score of 79.6 on all languages, compared to the baseline's 67.4.

1 Introduction

In this paper, we describe our system and approach for the SemEval 2022 PreTENS (Presupposed Taxonomies: Evaluating Neural Network Semantics) shared task (Zamparelli et al., 2022).[1] The aim of this task is to gain a better understanding of the ability of language models to recognize taxonomic relations between two words.

We focus on subtask 1, which is a binary classification task in which a system should predict whether a sentence is valid or not, depending on the taxonomic word relation in a given sentence. We formulate the following research question:

> What are effective data generation approaches in order to improve a language model's ability to recognize appropriate taxonomic word relations?

In our attempt to answer this question, we experiment with multiple approaches: adding new templates, adding new nouns from similar word lists, adding additional hyponyms, inverting templates and using a paraphrasing model to create sentence variations. We use the expanded training data to fine-tune a base English BERT (Devlin et al., 2019) model for the final classification task.

In our approach, we incorporate all three languages for this task: English, Italian and French. Instead of generating additional data for each language and training separate models, we opt to train an English model and translate the Italian and French sentences to English, before predicting the validity labels. We choose this approach in part because several of our data generation methods are not available for French or Italian. We make use of Google Translate, as this is a widely used state-of-the-art general-domain translation system. Our model, trained on the expanded dataset, scores an average F1 score across all languages of 79.6, which is an improvement over the 67.4 baseline score. We find that the best data expansion technique is to combine multiple approaches, where the output of one method is the input for the next. Our ablation experiments show that our paraphrasing method improves scores the most. All code, data and other related files can be found in our GitHub repository.[2]

2 Task description

For the binary classification subtask, the challenge was to predict the acceptability label assigned to each sentence of the test set. The participants were provided with a training set consisting of 5,838 sample sentences and their validity labels, while the test set contained 14,556 sentences. The splits were provided in English, Italian and French, the latter two being slightly adapted translations of the English dataset. These sentences exemplify constructions enforcing presuppositions on the taxonomic status of their arguments A and B, as can be seen in the following examples:

[*]Contributed equally

[1]`https://sites.google.com/view/semeval2022-pretens/home-page`

[2]`https://github.com/WPoelman/shared-task`

I like trees, and in particular birches.	1
I like oaks, and in particular trees.	0

A sentence will only get a validity label of 1 when the taxonomic relations are compatible with the sentence construction.

For this task, participants were free to use external resources, with the exception of lexical resources where semantic relationships (including taxonomic ones) are manually marked, such as WordNet (Miller, 1995) or BabelNet (Navigli and Ponzetto, 2012). However, using these lexical resources was allowed for the generation of data, which is part of our approach.

The task of detecting taxonomic word relations has been tried via various approaches. From purely rule-based (Hearst, 1992), to using semantic tree-like resources (Navigli et al., 2011), to adapting pre-trained language models (Atzori and Balloccu, 2020; Chen et al., 2021) or creating hybrid systems (Shwartz et al., 2016; Ravichander et al., 2020). Since this SemEval task is focused on *neural* (language) models, we opt to use BERT and focus mainly on different data generation techniques.

3 System overview

Our main research focus is to explore the effects of different data generation approaches to expand the English training data. This data is then used to train an English BERT model. The unseen Italian and French sentences (to predict) from the test set are first translated into English before we feed them to the model to get their final prediction. We describe these different stages in more detail below.

3.1 Development data split

In order to evaluate our experiments during the development of our system, we created a test set from the original training data. The aim of this set is to replicate the expected official test data characteristics as well as possible. The original training data, as published by the task organizers, consists of seven different templates, distinguished by the words used to describe the relation between two nouns (thus disregarding pronouns). We categorize these in three types of relations:

1. No hypernym relations possible: *I do not like pigs, I prefer animals*
2. Word A is a hypernym of word B: *I like animals, except pigs*
3. Word B cannot be a hypernym of word A: *I like animals, but not pigs*

The first two categories are seen in two templates, while the last category is only relevant to one template. In order to evaluate our techniques, we created our own test set which consists of all sentences with three templates: one for each defined relation. These templates are not present in the training set. Additionally, we filtered nouns that were used with the verbs 'use' and 'met', i.e. verbs that relate to the noun categories 'materials' and 'people'. These nouns were only present in our test set, to ensure that we also evaluate the system on unseen nouns. Additionally, adding the sentences with these nouns means that the test set contains all seven templates, so that there are four overlapping templates with the training set, as this also seems similar to the official test set, where we expected some overlap with training sentences.

3.2 Data generation methods

3.2.1 New templates

First, new templates were added. As described in the previous section, our initial training split consists of only four templates. However, since sentences occur in many different variations, we manually wrote templates in the three previously distinguished relation categories.[3] In total, we added 58 new templates: 22 for the first relation, 23 for the second and 13 for the third.

3.2.2 New words

Our next step was to add new words to the templates. In the task description it was mentioned that the nouns were divided into 30 semantic categories, such as dogs, mammals, motorcycles etc. Not all categories were given, so we tried to infer this from the training data. We extracted all unique nouns from the training data and divided these into lists of different categories. With these categories, we tried to approximate and expand the semantic categories given by the organizers. Certain changes were made, for example, splitting the 'entertainment' category, consisting of books, movies, games and music into separate categories, or combining mammals and dogs into an 'animals' category. Words that occurred together but did not fit into a category, which includes emotions and buildings among others, got assigned to a 'miscellaneous' category. Finally, 14 categories were identified. These categories were also tied to certain verbs in the provided training data. The verb

[3]The full criteria of writing these can be found in our data split description.

'like' was used in all categories, but 'use' was only paired with materials, for example.

The word lists were then enriched. This was mainly done manually, with the help of searching for all hyponyms for a word in WordNet. More verbs were added as well. All categories and their verbs can be found in Table 1. Moreover, we added additional pronouns, with their corresponding possessive pronouns. In addition to *I/my* and *he/his*, we added *she/her* and *they/their*. These lists were used to fill in the existing and new templates. The sentences were checked on their validity using WordNet in order to generate their labels. The generated data is balanced, meaning that each category gets an even amount of relation types, which, in turn, get an even amount of 1's and 0's.

Category	**Verbs**
Animals	like, **love**
People and professions	like, **love**, met
Materials	like, **love**, use
Games and sports	like, **love**, **enjoy**, **play**
Clothing and jewelry	like, **love**, **wear**
Drinks	like, **love**, **enjoy**, **drink**
Food	like, **love**, **enjoy**, **eat**
Transport	like, **love**, **enjoy**
Movies	like, **love**, **enjoy**, **watch**
Music	like, **love**, **enjoy**, **listen to**
Books	like, **love**, **enjoy**, **read**
Plants	like, **love**
Furniture and household items	like, **love**
Miscellaneous	like, **love**, **enjoy**, **feel**, trust

Table 1: Categories and their corresponding verbs. The bold verbs have been added to the data.

3.2.3 Hyponyms of hyponyms

We noticed that in the valid sentences containing appropriate taxonomic relations, the used arguments always have the same role throughout the data set. As an example, in the valid sentence *I like seafood, except salmon* the argument 'seafood' is a hypernym and the argument 'salmon' is a hyponym. In all other sentences of the training data, the word 'salmon' is also exclusively used as a hyponym, even though it can be a hypernym in a sentence such as *I like salmon, except redfish*.

In line with this example, we created additional sentences where words previously used exclusively as a hyponym, were now used as a hypernym. To do so, we extracted all the hyponyms occurring in the training data and searched WordNet for their direct hyponyms. We then used the first five returned results to generate both valid and invalid new sentences. In creating these additional sentences, we aim to challenge the language model to recognize appropriate taxonomic relations even when the role of an argument alternated between being a subcategory and a supercategory.

3.2.4 Inverting

As mentioned before, we categorized the training data into three possible template relations. For a sentence with the relation 'X is a hypernym of Y', e.g. *I like animals, except pigs*, we know that the sentence is valid because 'animal' is a hypernym of 'pig'. Following this logic, we also know that swapping the arguments to create *I like pigs, except animals*, invalidates the sentence. This process of swapping the arguments in a sentence and (possibly) changing the validity is what we call 'inverting'. Note that not every sentence's validity will change when inverted. When swapping the arguments in the valid sentence *I like jazz more than jewelry* to *I like jewelry more than jazz*, the sentence remains valid. Therefore, we carefully looked at the conditions that make a sentence valid or invalid.[4] We then swapped the two arguments of each sentence and changed the validity label when applicable to create additional data. The language model should learn that the validity also depends on the order of the arguments.

3.2.5 Paraphrasing

In addition to data generation, we also wanted to look at synthetic data. This led us to experiment with a neural paraphrasing model. Specifically we used a fine-tuned 'Pegasus' model from Google (Zhang et al., 2020), originally trained on the task of summarizing and fine-tuned on paraphrasing. We used this model as the final step in our data generation pipeline in order to generate paraphrases of all method combinations. The following is an example the paraphrasing outputs:

I do not like dogs, I prefer blackbirds.

- I prefer blackbirds, I don't like dogs.
- I don't like dogs and I like blackbirds.
- I don't like dogs and I prefer blackbirds.

As the example shows, the differences are not drastic, but do introduce some variation. In order to prevent the paraphrasing model from outputting unrelated sentences, we applied some filtering steps to restrain the output of the model. For instance,

[4] Valid and invalid relations for each template are described in our data split description.

we added length constraints, which ensures that the newly generated sentences were not too short, but neither too long. We allowed at most ten paraphrases per sentence.

3.3 Lemmatization

Since the generated sentences were not all grammatically correct (e.g. ‘He like’ instead of ‘He likes’) and because of the fact that the nouns in the original data were plural and the nouns extracted from WordNet were singular, we experimented with lemmatizing the nouns and verbs in the generated sentences. This might remedy the incorrect sentences by equally pre-processing all sentences.

3.4 Translation

By combining the output from our different generation methods, we created an English training set. We used this to fine-tune a pre-trained English language model, namely BERT base provided by Hugging Face.[5] Then, in order to predict the validity labels for Italian and French sentences, we translated these into English in order to process them with the English model. We opted for this approach since several of the data generation methods were not available for French or Italian. For instance, there were no paraphrasing models or easily accessible WordNet-like resources available.

For the translation system, we use Google Translate, as this general-domain transformer-based system is the state-of-the-art. Manual inspection revealed that translation quality seemed sufficient for our purposes.

3.5 System

We experiment with various combinations of the previously mentioned data generation techniques. The detailed results of these experiments are described in the Results section. Our final training dataset was created as follows:

1. Take the existing and new templates (not the ones included our test set)
2. Fill these in with new words from our word lists
3. From the resulting sentences, create additional sentences using the noun hyponyms

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

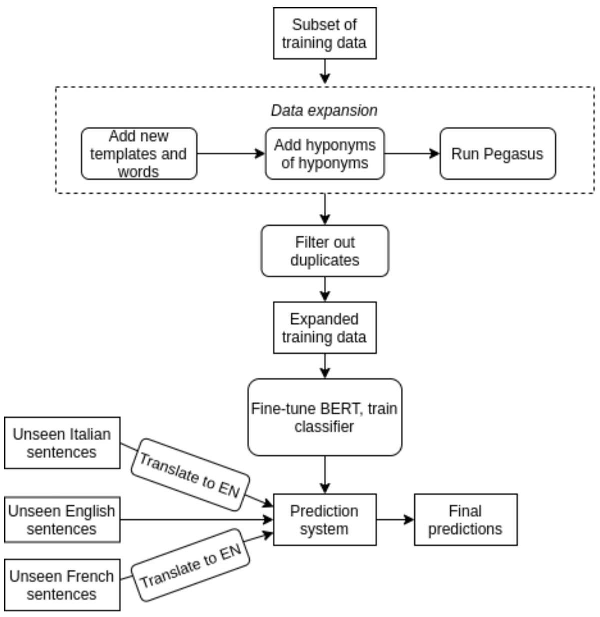


Figure 1: Overview of system and prediction pipeline.

4. Add paraphrases of all generated sentences
5. Finally, filter out duplicate sentences

This resulted in dataset of 211,354 sentences, which was used to train our final model. Figure 1 shows an overview of our entire system to get the final predictions.

4 Experimental setup

To run our experiments, we created various combinations of our data generation techniques. The full list of experiments can be seen in Table 3. The paraphrasing model we used comes from the Hugging Face model hub.[6]

We trained each system using the Hugging Face transformers library.[7] The exact hyperparameters and other settings can be found in Appendix A. The final model we used for generating our submission can be found in our GitHub repository.

As mentioned, all models were tested using our custom test set with special challenging characteristics.

5 Results

In this section, we discuss the results on both our custom test set and the official test set. We also provide an error analysis in order to gain insight into

[6] https://huggingface.co/tuner007/pegasus_paraphrase
[7] https://huggingface.co/docs/transformers

the contributions of the different data generation methods.

5.1 Custom test set

The full results of our experiments on our custom test set can be found in Appendix B. All data generation methods improve performance over the originally provided training set, while there are some notable differences between the methods. For instance, when looking at the individual methods, we observe that adding the hyponyms has a rather small effect, especially considering that the size of the dataset is almost doubled. The size of the dataset also does not seem be a direct indicator of increased performance. Inverting applicable templates, for example, increases the training set by about 120 sentences, but increases the F1 score from 53.1 to 61.7.

Another surprising effect can be observed in the lemmatized and normal versions of the same datasets. With the individual methods we can see a clear increase in performance, but this effect is not visible with the fully combined methods. The same applies to the inverting method, which resulted in worse performance in some cases in the fully combined methods, which was not the case when it was used on its own.

5.2 Official test set

The official evaluation for the first subtask was twofold:

1. Ranking per language.
2. Global ranking - the average score across all three languages.

While the systems were evaluated using the metrics of precision, recall, micro F1 and macro F1 scores, the official rankings were based only on macro F1 scores.

With our submission, we achieve an average macro F1 score of 79.6, which ranks our system 14th out of 21 participants. We achieve macro F1 scores of 80.3, 78.6 and 79.7 for English, Italian and French respectively. Our system improves over the baseline system, which obtains a macro F1 score of 67.4.

5.3 Error analysis

Once the official test set with labels was available, we conducted ablation experiments to see what effects the different data generation methods had on our submission. Of all methods, the sentences generated with the Pegasus model seem to contribute the most to our final model. The full results of these experiments can be seen in Table 2. To our surprise, the split of our best dataset *without* the new words scores better on the official test set. An explanation could be that using the extensive word lists, which contain rare words, can lead to the generation of vague or uncommon sentences. However, this effect was not apparent in our custom test set, both as an individual method as well as combined with others. Apart from this, the scores are all quite close and again show that the increase in dataset size does not directly lead to better performance.

	Sentences	Acc	Pre	Rec	F1
Templates, new words, hyponyms, pegasus (used for final submission)	211,354	80.7	86.0	70.4	77.4
- pegasus	33,571	77.2	80.0	68.9	74.0
- new words	108,819	81.9	84.1	75.9	79.8
- hyponyms	116,234	79.0	79.9	74.1	76.9

Table 2: Ablation test with final expanded training dataset on official test set.

For each ablation test, we analyze incorrectly predicted examples from a particular model, which were predicted correctly by the others. We look at both the templates and words of the official test set. Additionally, the official set provides information about the structure that was used by the task organizers to generate a particular sentence, which we also include in the analysis.

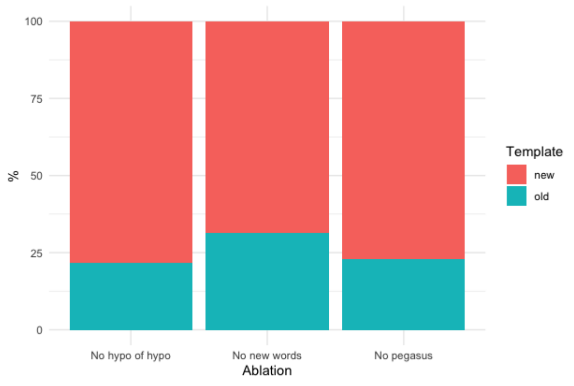


Figure 2: The ratio of incorrectly predicted instances with old and new templates per ablation test.

In Figure 2 we can see the effects of the ablation methods on the templates in the test set. Not adding new words results in a lower error rate for new templates. This is unexpected, as generating sentences with new words was also done using new manual templates, however the test set still contains completely unseen templates. Another surprise is that not including paraphrasing did not result in relatively more incorrect predictions on new templates, since paraphrasing does introduce some variation in templates to an extent.

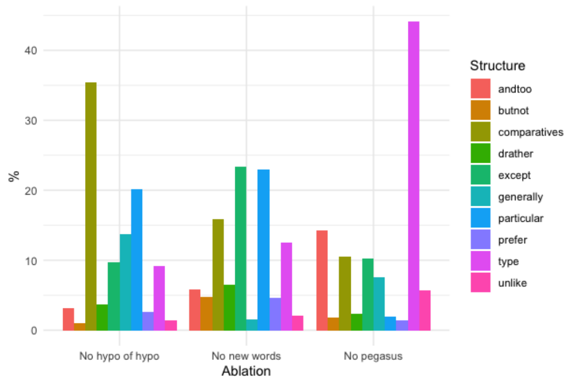


Figure 3: The ratio of structure types in the incorrectly predicted instances per ablation test.

Figure 3 displays the effect of the ablation tests on the different structures. Without the hyponyms of hyponyms method, the error rate of the *comparatives* is high. Excluding the paraphrases results in a higher error rate for the *type*-structure and a lower error rate for the *particular*-structure. This could be because the *type*-sentences are mostly paraphrased correctly which is not the case for *particular*-sentences.

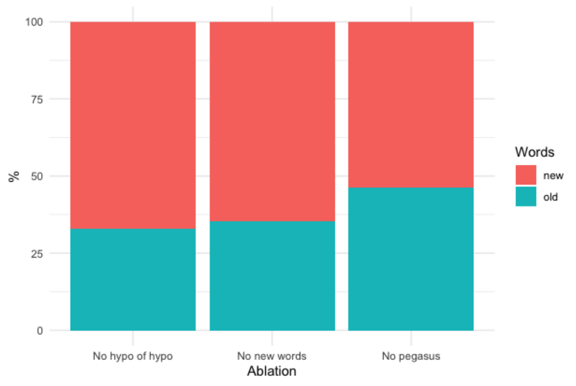


Figure 4: The ratio of incorrectly predicted instances with old and new words per ablation test.

Figure 4 shows the effect of the ablation tests on old and new words. We expected the error rate for ablating new words to be higher. However, new words in the test set were mostly from completely new categories, as opposed to more words from the same categories, which our approach was based on.

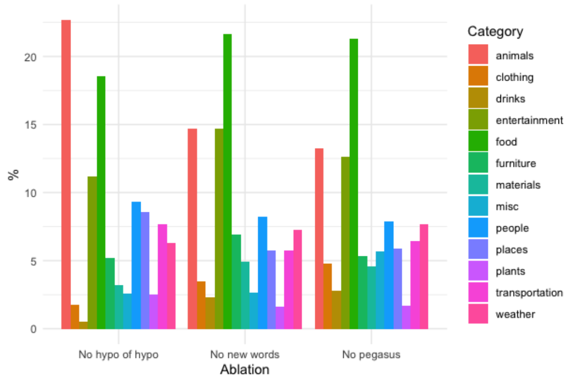


Figure 5: The ratio of word categories in the incorrectly predicted instances per ablation test.

Figure 5 shows the effect on the different categories of words for the ablation tests. These are categorized in the same manner as described in the 'new words' section. There are two new categories in the official test set: *places* and *weather*. Both did not see an increase in error rate by ablating any method. The error rate of *animals* increased by ablating hyponyms of hyponyms. The extensive manner in which the taxonomy of animals is represented in WordNet might be the reason behind this.

6 Conclusion

We have outlined several data generation methods and show that all methods improve the performance of a model trained on the expanded datasets compared to the original training data. Especially new words, new templates and paraphrasing are effective, even by themselves. Surprisingly, adding hyponyms is not that effective by itself, but combined with the previously mentioned methods it scores quite well. The final system we used to participate in the task was trained on a dataset created from those four methods. With this system, we improve the baseline set by the task organizers, placing us 14th out of 21 participants.

As we have shown, some unexpected effects occur when looking at how different data generation techniques perform when combined. In future research, it might be interesting to explore this in a broader context: what can cause such differences? Furthermore, since the provided dataset consisted of short and specific template-based sentences, it could be interesting to experiment with longer sentences that contain more complex constructions.

7 Acknowledgments

We are grateful to Tommaso Caselli and Lukas Edman for their supervision and helpful comments. We also want to thank the Center for Information Technology of the University of Groningen for providing access to the Peregrine high performance computing cluster.

References

Maurizio Atzori and Simone Balloccu. 2020. Fully-unsupervised embeddings-based hypernym discovery. *Information*, 11(5).

Catherine Chen, Kevin Lin, and Dan Klein. 2021. Constructing taxonomies from pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4687–4700, Online. Association for Computational Linguistics.

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

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial intelligence*, 193:217–250.

Roberto Navigli, Paola Velardi, and Stefano Faralli. 2011. A graph-based algorithm for inducing lexical taxonomies from scratch. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102, Barcelona, Spain (Online). Association for Computational Linguistics.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. Semeval-2022 task3 (pretens): Evaluating neural networks on presuppositional semantic knowledge. In *Proceeding of SEMEVAL 2022*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

A Hyperparameters

We used the Trainer class from Hugging Face, which was mostly left at the default settings. The log of the trainer is included in the folder with our final model, the link to which can be found in the readme of our repository. We put in a time limit of 6 hours for all models. Parameters:

```
optimizer: AdamW
learning rate: 5e-05
batch: 8
scheduler: linear
max epochs: 4
```

B Experiment results on custom test set

	Sentences	F1
Individual methods		
Train	2,737	53.1
Train, hyponyms	4,957	55.5
Train, inverted	2,868	61.7
Train, new words	21,456	65.4
Train, templates (new words)	9,000	71.3
Train, templates (only original train set words)	9,000	73.1
Train, new words lemmatized*	21,456	73.1
Train, pegasus	19,484	77.6
Combined methods		
Train, hyponyms, templates	18,831	70.0
Train, new words, templates	21,456	74.7
Full pipeline combinations		
Templates, new words, inverted, pegasus	138,572	57.9
Templates, new words, hyponyms, pegasus, lemmatized*	211,354	59.1
Templates, new words, hyponyms, inverted	40,820	62.2
Templates, new words, hyponyms, inverted, pegasus	282,834	81.5
Templates, new words, hyponyms, inverted, pegasus, lemmatized*	282,796	83.6
Templates, hyponyms, inverted, pegasus	147,008	85.6
Templates, new words, hyponyms, pegasus	211,354	**88.1**

Table 3: Data experiment results on our own test set. ‘Train’ refers to the original train set. *Models trained on lemmatized input data were also evaluated on a lemmatized test set. The model trained on the best scoring dataset, in bold, was used for the final submission.

CSECU-DSG at SemEval-2022 Task 3: Investigating the Taxonomic Relationship Between Two Arguments using Fusion of Multilingual Transformer Models

Abdul Aziz, Md. Akram Hossain, and Abu Nowshed Chy
Department of Computer Science and Engineering
University of Chittagong, Chattogram-4331, Bangladesh
{aziz.abdul.cu, akram.hossain.cse.cu}@gmail.com,
and nowshed@cu.ac.bd

Abstract

Recognizing lexical relationships between words is one of the formidable tasks in computational linguistics. It plays a vital role in the improvement of various NLP tasks. However, the diversity of word semantics, sentence structure as well as word order information make it challenging to distill the relationship effectively. To address these challenges, SemEval-2022 Task 3 introduced a shared task PreTENS focusing on semantic competence to determine the taxonomic relations between two nominal arguments. This paper presents our participation in this task where we proposed an approach through exploiting an ensemble of multilingual transformer models. We employed two fine-tuned multilingual transformer models including XLM-RoBERTa and mBERT to train our model. To enhance the performance of individual models, we fuse the predicted probability score of these two models using weighted arithmetic mean to generate a unified probability score. The experimental results showed that our proposed method achieved competitive performance among the participants' methods.

1 Introduction

Lexical relations are regarded as the most important semantic relations to infer the meanings of words effectively (Khoo and Na, 2006). Therefore, identifying such relations is beneficial to understand the semantic competence and distill the underlying context of the textual expression. If we consider the NLP applications, it has a significant impact on several tasks including semantic search, automatic question/answering, story generation, relation extraction, and machine translation (Barkan et al., 2020). However, most of the prior works focused mainly on syntactic (Luu et al., 2014) and contextual relation (Maksimov et al., 2018) to infer the lexical relation in between words.

Sentence	Language	Label / Score
Sub-task 1		
Apprezzo il vino , ma non il Chianti.	It	1
J' aime les chats, sauf les beagles.	Fr	0
I like movies, but not comedies.	En	1
I like earrings, except socks.	En	0
Sub-task 2		
Amo i merli piÃ¹ degli uccelli.	It	1.9
J'aime les chats, et aussi les canards.	Fr	6.17
I like cats, but not sparrows.	En	4.77

Table 1: Examples of sub-task 1 and sub-task 2.

Considering this PreTENS shared task at SemEval 2022 (Zamparelli et al., 2022) introduce a new task that focuses mainly on semantic competence with specific attention on the estimating taxonomic relations between two nominal arguments. The taxonomic relation here means one argument is a supercategory of the other, or in extensional terms, one argument is a superset of the other. The task is divided into two subtasks. The first one is a binary classification subtask, where a system needs to predict the acceptability label for given text considering the taxonomic relation. The second one is a regression subtask, where a system needs to predict the percentage of acceptability label on a seven-point Likert-scale for a given text considering the same scenario. Besides, the task addresses the challenges of multilingual expression and comprises a dataset of three different languages including Italian (It), French (Fr), and English (En). To illustrate a clear view of the task definition and research goal, we articulate a few examples from different languages and corresponding labels for each subtask in Table 1.

We articulate the rest of the contents as follows: Section 2 describes our proposed approach

**The first two authors have equal contributions.

whereas, in Section 3, we present our experimental setup and conduct performance analysis against the various settings and participants' methods. Finally, we conclude our work with some future directions in Section 4.

2 Proposed Framework

In this section, we describe our proposed approach for the PreTENS shared task. Our goal is to determine the semantic competence between two arguments focusing on the taxonomic relations. The task is articulated into both the binary classification subtask and the regression subtask. We depict the overview of our proposed framework in Figure 1.

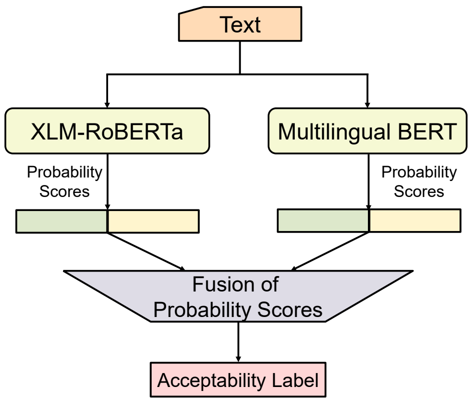


Figure 1: Overview of our proposed framework.

Given an input sentence containing two nominal arguments, we employ two transformer models including XLM-RoBERTa and multilingual BERT (mBERT) to extract the diverse contextual features. Later, a feed-forward linear layer is employed in each model to estimate the probability score of each class. In Figure 2, we illustrate an overview of the setup of mBERT transformer model. Finally, we fuse these models' predictions by taking the weighted arithmetic mean of these scores to determine the final label.

2.1 XLM-RoBERTa

The Facebook AI launched the XLM-RoBERTa as an upgrade to their initial XLM-100 model (Conneau et al., 2020). It is a scaled cross-lingual sentence encoder. Using self-supervised training approaches, it offers state-of-the-art performances in cross-lingual understanding where a model is taught in one language and then applied to multiple languages with no additional training data. This model showed increased performance on numerous NLP applications. XLM-RoBERTa creates the

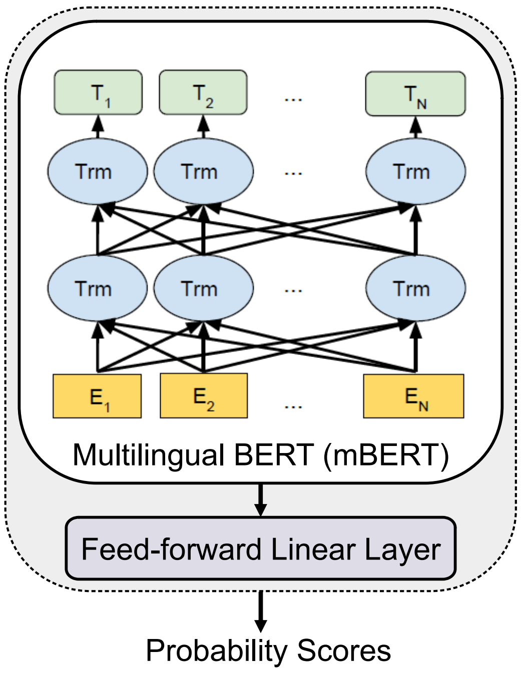


Figure 2: Multilingual BERT (mBERT) model.

possibility for a one-model-for-many-languages approach rather than a single model per language.

However, XLM-R maps the same sentence in different languages to similar representations which is crucial for this PreTENS task to learn semantic competence in cross-lingual form. Here, we use the HuggingFace's implementation of the XLM-RoBERTa-base model (Conneau et al., 2020). It is composed of 12-layers (i.e. transformer block), the dimension of hidden size is 768, the number of the self-attention head is 12, the size of vocabulary is 250K, and containing 270M parameters.

2.2 Multilingual BERT

Multilingual-BERT (mBERT) (Devlin et al., 2019) is a version of BERT that is gaining popularity for effective contextual representation of textual contents in various multilingual tasks including natural language inference in cross-lingual characteristics, named entity recognition in multilingual corpora, and dependency parsing (Chi et al., 2020). It is pretrained on 104 different languages in Wikipedia. It provides an effective path to zero-shot cross-lingual model transfer.

Multilingual-BERT representations are influenced by high-level grammatical features that are not manifested in any one input sentence which is critical to learn taxonomic relations in PreTENS task. In our approach, we employ the huggingface

https://huggingface.co/xlm-roberta-base

implementation of the bert-base-multilingual-cased model and perform finetuning the model with the task-specific data.

2.3 Fusion of Transformer Models

To enhance the performance of individual models, we fuse the predicted probability score of two models to generate a unified probability score for each class. We use the weighted arithmetic mean to average both model's probability scores to determine the each class confidence of the fused model. After calculate the both class probability score of fused model, our system predict the best confidence class, which is our final label. The estimation is computed as follows:

$$f(x_i, y_i) = \begin{cases} 0, & \text{if } W_0 > W_1 \\ 1, & \text{otherwise} \end{cases}$$

$$W_i = \frac{x_i * M + y_i * R}{M + R}$$

x_i and y_i correspond to the mBERT and XLM-RoBERTa probability score, where M and R represents their weight respectively. W_i (i.e. i = {0, 1}) is the unified probability score for each class.

3 Experiment and Evaluation

3.1 Dataset Description

The organizers of the SemEval-2022 PreTENS shared task 3 (Zamparelli et al., 2022) provided a benchmark dataset to evaluate the performance of the participants' systems. The dataset comprises in 3 languages including English, Italian, and French. The French and Italian are slightly adapted translations of the English dataset. The dataset statistics is summarized in Table 2.

Language	Sub-task 1		Sub-task 2	
	Train	Test	Train	Test
It	5837	14560	524	1009
Fr	5837	14560	524	1009
En	5837	14560	524	1009
Total	17511	43680	1572	3027

Table 2: The statistics of the datasets used in PreTens shared task.

Dataset of each languages contained about 20K artificially generated text samples that enforces presuppositions on the taxonomic status of their arguments A and B, e.g. comparatives (I like A more than B), exemplifications (I like A, and in particular B), generalizations (I like A, and B in general), and others. In Subtask 1, all samples are provided with an acceptability label either 1 or 0 where 1 stands for acceptable (i.e. the taxonomical relations is compatible with the construction at issue) and 0 stands for not acceptable (not compatible). Besides, a subset of 1533 samples of the whole dataset i.e. 5% of the total and representative of the patterns considered, was used for the subtask 2. It was annotated via a crowdsourcing campaign on a seven point Likert-scale, ranging from 1 (not at all acceptable) to 7 (completely acceptable). The average judgment is considered as the final label for each sample.

https://huggingface.co/bert-base-multilingual-cased

3.2 Experimental Settings

We now describe the details of our experiments and set of parameters that we have used to design our proposed CSECU-DSG system for each subtask.

Parameter	Optimal Value
Subtask 1: Parameters used in both models	
Learning rate	3e-5
Max-len	100
Epoch	3
Batch size	16
Manual seed	4
Subtask 2: Parameters of XLM-RoBERTa	
Learning rate	3e-5
Max-len	100
Epoch	5
Batch size	8
Manual seed	4

Table 3: Model settings for each subtask.

In Subtask 1, we utilize the Huggingface (Wolf et al., 2019) implementation of the two state-of-the-art multilingual transformer models with finetuning, including XLM-RoBERTa and mBERT. We finetune these models with the provided training data. To generate the unified prediction, we fuse the probability score of each model as described in Section 2.3. Since XLM-R typically perform better than the mBERT, so we don't count both model confidence weight as equal. However, from some sets of experimental result we choose the weight R = 0.6 for XLM-RoBERTa and M = 0.4 for mBERT in equation 2.3. However, in Subtask 2, we only employed the XLM-RoBERTa model with finetuning strategy. The optimal parameter settings used in both subtasks are articulated in Table 3 and we used the default settings for the other parameters.

3.3 Evaluation Measures

To assess the performance of the participants' systems, PreTENS task organizers (Zamparelli et al., 2022) used different strategies and metrics for subtask 1 and subtask 2. Since the evaluation file contains instances from all three languages, the average of the F1-macro score from all the three languages is used as the global ranking score (GRS) to rank the participants' systems. We can write this as follows:

$$\text{Global Rank Score, } GRS = \frac{AS}{3}$$

where AS=(F1-macro (English) + F1-macro (French) + F1-macro (Italian)).

In subtask 2, the average of the Spearman correlation (Rho) scores from all the three languages is used as the global ranking score (GRS) to rank the participants' systems. We can write this as follows:

$$\text{Global Rank Score, } GRS = \frac{AS}{3}$$

where AS=(Rho (English) + Rho (French) + Rho (Italian)).

3.4 Results and Analysis

In this section, we analyze the performance of our proposed approaches in the PreTENS shared subtasks. The dataset comprises of 3 different languages including English, Italian, and French and the overall performance of the system is estimated considering the average score obtains in each languages dataset. Considering this, we analyze the performance of our CSECU-DSG system, based on each language. The corresponding results for subtask 1 and subtask 2 are reported in Table 4 and Table 5, respectively.

Results showed that in Subtask 1 our CSECU-DSG system achieved a pretty good score in all sets of datasets considering three languages. It demonstrates the generalizability of our approach in diverse types of languages. However, our method limited to obtain a good score in Subtask 2 for all the datasets. One plausible reason behind this is to use only single transformer models instead of the fusion approach and failed to address and analyze the challenges of subtask 2 properly.

Language	F1-macro	F1
It	91.113	90.620
Fr	90.732	90.334
En	91.506	91.189
All (CSECU-DSG)	91.117	-

Table 4: Results of our proposed CSECU-DSG system on individual monolingual tracks (subtask 1).

Language	Spearman Cor. (Rho)
It	0.207
Fr	0.081
En	0.191
All (CSECU-DSG)	0.160

Table 5: Results of our proposed CSECU-DSG system on individual monolingual tracks (subtask 2).

Next, the obtained results of our proposed CSECU-DSG system in the PreTENS shared task along with other top performing and competitive participants systems in subtask 1 and subtask 2 are articulated in Table 6 and Table 7, respectively. Following the benchmark of the PreTENS shared task, participants' systems are ranked based on the primary evaluation measures of each subtask.

Team (Rank)	F1-macro			
	Global	It	Fr	En
CSECU-DSG (4th)	91.117	91.113	90.732	91.506
Performance of team based on F1-macro score				
LingJing (1st)	94.485	93.047	93.236	97.172
injySarhanUU (3rd)	91.574	92.118	89.529	93.076
holdon (6th)	89.579	86.903	87.281	94.551
cnxupupup (10th)	86.676	86.755	86.390	86.883
breaklikeafish (15th)	77.985	74.398	82.345	77.213
Baseline (18th)	67.394	59.588	72.126	70.468

Table 6: Comparative results with other selected participants (Sub-task 1).

Results showed that our system ranked 4th among the participants' systems in subtask 1. This deduces the efficacy of our approach in addressing

Team (Rank)	Spearman Correlation (Rho)			
	Global	It	Fr	En
CSECU-DSG (8th)	0.160	0.207	0.081	0.191
Comparative performance of team based on Rho score				
LingJing (1st)	0.802	0.807	0.841	0.758
huawei_zhangmin (3rd)	0.669	0.631	0.740	0.636
daydayemo (6th)	0.206	0.121	0.284	0.212
breaklikeafish (11th)	0.078	0.186	-0.010	0.059
thanet.markchom (13th)	0.056	0.089	-0.043	0.122
Baseline (4th)	0.309	0.344	0.317	0.265

Table 7: Comparative results with other selected participants (Sub-task 2).

the challenges of the PreTENS shared task. However, in subtask 2, our system obtained a poor score though we ranked 8th in this task.

To further analyze the effectiveness of the components used in our approach, we estimate the performance of each model used in the fusion approach in subtask 1. The results are reported in Table 8. It shows that our fusion strategy improves the overall performance of the model compared to the performance of the mBERT and XLM-RoBERTa. However, considering the individual model's performances XLM-RoBERTa performed better compared to the mBERT.

Method	F1-macro			
	Global	It	Fr	En
CSECU-DSG	**91.117**	**91.113**	**90.732**	**91.506**
Performance of individual model				
XLM-RoBERTa	90.217	90.322	89.853	90.475
mBERT	88.076	86.477	88.288	89.463

Table 8: Performance analysis of individual model using subtask 1 test dataset.

4 Conclusion and Future Directions

In this paper, we presented our proposed system to address the challenges of the PreTENS shared task. We employed the weighted fusion of two state-of-the-art multilingual transformer models predictions. Experimental results demonstrate the competency of our approach in Subtask 1.

In the future, we have a plan to incorporate the task specific features and technologies to address the challenges properly. We also have a plan to explore the causal inference techniques to distill the taxonomic relation.

References

Oren Barkan, Avi Caciularu, and Ido Dagan. 2020. Within-between lexical relation classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3521–3527.

Ethan A Chi, John Hewitt, and Christopher D Manning. 2020. Finding universal grammatical relations in multilingual bert. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

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

Christopher SG Khoo and Jin-Cheon Na. 2006. Semantic relations in information science. *Annual review of information science and technology*, 40(1):157–228.

Anh Tuan Luu, Jung-jae Kim, and See Kiong Ng. 2014. Taxonomy construction using syntactic contextual evidence. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 810–819.

NV Maksimov, AS Gavrilkina, VV Andronova, and IA Tazieva. 2018. Systematization and identification of semantic relations in ontologies for scientific and technical subject areas. *Automatic Documentation and Mathematical Linguistics*, 52(6):306–317.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. HuggingFace's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771.*

Robertoa Zamparelli, Shammur Absar Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. SemEval-2022 Task3 (PreTENS): Evaluating neural networks on presuppositional semantic knowledge. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

UoR-NCL at SemEval-2022 Task 3: Fine-Tuning the BERT-Based Models for Validating Taxonomic Relations

Thanet Markchom
University of Reading, UK
t.markchom@pgr.reading.ac.uk

Huizhi Liang
Newcastle University, UK
huizhi.liang@newcastle.ac.uk

Jiaoyan Chen
University of Oxford, UK
jiaoyan.chen@cs.ox.ac.uk

Abstract

In human languages, there are many presuppositional constructions that impose a constrain on the taxonomic relations between two nouns depending on their order. These constructions create a challenge in validating taxonomic relations in real-world contexts. In SemEval2022-Task3 Presupposed Taxonomies: Evaluating Neural Network Semantics (PreTENS), the organizers introduced a task regarding validating the taxonomic relations within a variety of presuppositional constructions. This task is divided into two subtasks: classification and regression. Each subtask contains three datasets in multiple languages, i.e., English, Italian and French. To tackle this task, this work proposes to fine-tune different BERT-based models pre-trained on different languages. According to the experimental results, the fine-tuned BERT-based models are effective compared to the baselines for classification. For regression, the fine-tuned models show promising performances with the possibility of improvement.

1 Introduction

Taxonomic relations are one of the significant lexical relationships that have been used in many applications such as question answering (Yih et al., 2013), sentiment analysis (Araque et al., 2019) and biomedical ontologies (Bodenreider, 2004). In natural languages, there are many constructions that constrain the taxonomic relation between two nouns based on the order of these two nouns. For instance, given a sentence "I have a dog, not a pet". The construction "I have a ..., not a ..." implies that the taxonomic relation does not hold between "dog" and "pet". This can be seen as a presupposition imposed by the construction. However, this is not true since dogs are pets. Thus, with various presuppositional constructions, validating taxonomic relations becomes more complicated in the real world.

To address this issue, SemEval2022-Task3 Presupposed Taxonomies: Evaluating Neural Network Semantics (PreTENS) (Zamparelli et al., 2022) introduces the task where taxonomic relations have to be validated in different presuppositional constructions. This task proposes novel datasets in multiple languages (i.e., English, Italian and French) containing sentences with different two-noun constructions. Each sentence is labeled by an acceptability label for classification and an acceptability score for regression. Two challenges have been raised in this task: (1) a taxonomic relation between two nouns in the sentence must be detected, and (2) the construction which embeds the two nouns must also be validated.

To effectively validate taxonomic relations in such constructions, understanding the contexts or semantic meanings of these constructions are the key. Many previous studies have shown that pre-trained models comprise the prior knowledge of context comprehension (Yang et al., 2019). Recently, the language mode called BERT has been widely used in several tasks. The BERT model can be pre-trained with the self-supervised method to generate word/token or sentence representations enriched with prior knowledge. Then, they can be fine-tuned specifically for many downstream tasks including validating taxonomic relations. Therefore, in this work, we adopt the pre-trained BERT-based models in different languages to utilize the prior knowledge from the resources that they were pre-trained with. Then, elaborating on the pre-training, we fine-tune these pre-trained models to predict the acceptability of each sentence.

2 Related Work

Pre-trained language models such as GloVe (Pennington et al., 2014) and ELMo (Peters et al., 2018) have been widely used to generate word/sentence representations for many NLP applications. These representations have been proven to be effective since they are enriched with knowledge from the pre-training resources. Recently, the transformer architecture (Vaswani et al., 2017), a multi-layer multi-head self-attention, has made the major success in NLP. With this architecture as a fundamental, a language model called BERT (Devlin et al., 2019) was proposed. Such model can be first pre-trained with two self-supervising tasks, masked LM and next sentence prediction. After that, they can be fine-tuned with additional output layers to create new models for various downstream tasks. Due to the huge success of the BERT model, many language models stemming from it have been proposed. These include RoBERTa (Liu et al., 2019), the more robust BERT model, and DistilBERT (Sanh et al., 2020), the modified BERT model using knowledge distillation methods.

All of these BERT-based models can be pre-trained on different corpora/resources depending on various purposes. Previously, there are variations of the BERT-based models pre-trained in different languages. MDZ Digital Library team at the Bavarian State Library introduced the variations of the BERT-based models pre-trained on Italian and German corpora[1]. Le et al. proposed FlauBERT, pre-trained on a large French corpus consists of texts in diverse topics and writing styles. Although they have been used to solve several tasks in many languages, how to use these models in validating taxonomic relations in real-world contexts still remains an open issue.

3 System Overview

This task consists of two sub-tasks (1) a binary classification sub-task and (2) a regression sub-task. For sub-task 1, each sentence in the datasets is labeled with 1 if it is acceptable and 0 otherwise. For sub-task 2, each sentence is labeled with the average score assigned by human annotators. The score is on a seven point Likert-scale ranging from 1 that means “not at all acceptable” to 7 that means “completely acceptable”.

This paper proposes two similar approaches of fine-tuning the BERT-based models for taxonomic relation classification and regression. For both sub-tasks, we select three pre-trained BERT-based models that were pre-trained on three corpora with different languages. For English, we choose *DistilBERT-Base-Uncased* [2], pre-trained on Toronto Book Corpus and full English Wikipedia. The model has 6 layers, 12 heads, and 768 embedding dimension and has 66M parameters in total. For Italian, we select *BERT-Base-Italian-XXL-Uncased* [3], the Italian BERT model pre-trained on texts from a recent Wikipedia dump and the OPUS corpora collection. This model consists of 12 layers, 12 attention heads and 768 embedding dimension. The total number of parameters is 110M parameters. Lastly, for French, we select *FlauBERT-Base-Uncased* [4], pre-trained on text corpus consists of 24 sub-corpora gathered from different sources such as Project Gutenberg [5] and Common Crawl [6]. This model has 12 layers, 12 attention heads and 768 embedding dimension. The total number of parameters is 137M parameters. Based on these pre-trained models, an additional layer is added in each model to fine-tune these models. Each sub-task has different settings for fine-tuning.

Sub-task 1: Binary Classification To fine-tune the models for sub-task 1, a fully-connected layer is added on top of the pooled output (the sequence embedding, i.e., the “[CLS]” token embedding from the pre-trained model). This layer has an output size 2 and the softmax activation function. It outputs the probability of each class (1 and 0). The loss function for fine-tuning is the binary cross-entropy loss. The final prediction of each sentence is made by selecting the class with the maximum probability.

Sub-task 2: Regression Similarly to sub-task 1, we add a fully-connected layer on top of the pooled output for model fine-tuning. This layer has an output size 1. This output is the predicted score for a regression task. The mean squared error loss function is used for fine-tuning this model.

4 Experiments

We conducted experiments on the training and test sets provided by the task organizers. For each sub-task, there are three training sets and three test sets for three different languages. The training set of each language has 5,837 samples while the test set has 14,560 samples. Both training and test sets consist of sentences with different presuppositional

[1]https://github.com/dbmdz/berts

[2]https://github.com/huggingface/transformers

[3]https://github.com/dbmdz/berts

[4]https://github.com/getalp/Flaubert

[5]https://www.gutenberg.org/

[6]https://data.statmt.org/ngrams/deduped2017/

Construction	Sentence
andtoo	I like forests, and cities too.
butnot	I like sports, but not football.
comparatives	I like movies less than videogames.
drather	I would rather have beagles than rabbits.
except	I like pets, with the only exception of hamsters.
generally	I like bracelets, and more generally jewelry.
particular	I like fruits, and more specifically lemons.
prefer	I do not like cauliflower, I prefer apples.
type	I can stand rainstorms, an interesting type of rain.
unlike	Unlike cats, ducks are often mentioned in this text.
ingeneral	I like mountains, and nature in general.

Table 1: Examples of sentences with different constructions

Test set	Sub-task 1					Sub-task 2			
	Model	Precision	Recall	F1	F1-macro	Model	MSE	RMSE	ρ
English	Baseline	0.642	**0.866**	0.734	0.734	Baseline	4.45	2.11	0.23
	En-C	**0.900**	0.824	**0.860**	**0.873**	En-R	**2.71**	**1.65**	**0.24**
Italian	Baseline	0.557	0.877	0.682	0.682	Baseline	4.18	2.05	**0.40**
	It-C	**0.820**	**0.938**	**0.875**	**0.874**	It-R	**3.86**	**1.97**	0.04
French	Baseline	0.651	0.899	0.755	0.755	Baseline	4.66	2.16	**0.30**
	Fr-C	**0.763**	**0.905**	**0.828**	**0.823**	Fr-R	**3.65**	**1.91**	0.23

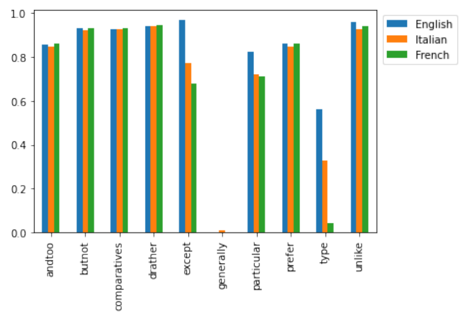
Table 2: Results of sub-task 1 and 2

constructions. For sub-task 1, there are 10 constructions, i.e., “andtoo”, “butnot”, “comparatives”, “drather”, “except”, “generally”, “particular”, “prefer”, “type” and “unlike”. For sub-task 2, there are 7 constructions, i.e., “andtoo”, “butnot”, “comparatives”, “ingeneral”, “particular”, “type” and “unlike”. Table 1 shows examples of sentences with different constructions. All the models were fine-tuned by using the Adam optimizer for 3 epochs with the batch size 16 and the learning rate 5e-5. For sub-task 1, the models were evaluated by Precision, Recall, F1 and F1-macro. For sub-task 2, they were evaluated by MSE, RMSE and Spearman Correlation (ρ). It is worth noting that ρ is used to measure the rank correlation between actual labels and predictions. It ranges between -1 to 1 where the higher value means the labels and predictions have a similar rank (or identical when it is 1) and the lower value means they have a dissimilar rank. (or fully opposed when it is -1) For each sub-task, a simple classification model using n-grams as features was used as a baseline.

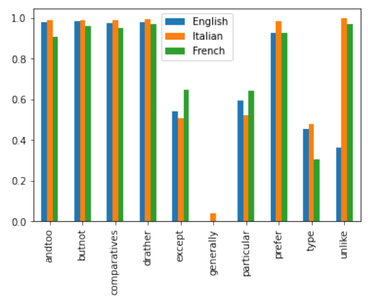
4.1 Sub-Task 1 Results and Discussion

For this sub-task, we named the fine-tuned DistilBERT-Base-Uncased for classification as **En-C**, the fine-tuned BERT-Base-Italian-XXL-Uncased for classification as **It-C** and the fine-tuned FlauBERT-Base-Uncased for classification as **Fr-C**. Table 2 shows the results of sub-task 1. From this table, It-C and Fr-C performed better than the baseline in every evaluation metric. Meanwhile, En-C outperformed the baselines in terms of Precision, F1 and F1-macro.

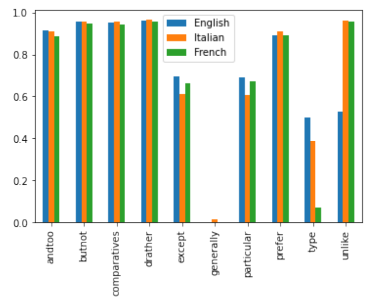
We further investigated the performance of our approaches by comparing the results of En-C, It-C and Fr-C on each construction. The results are shown in Figure 1. From this figure, we can see that En-C, It-C and Fr-C performed particularly poorly on “generally” construction. To identify the mistake, we examined the confusion matrices of their performance on “generally” construction as shown in Figure 2. This figure shows that all of them failed in predicting the true positive cases of this construction. To answer why they failed to predict the true positive cases, we further examined the attention weights at the last layer of these



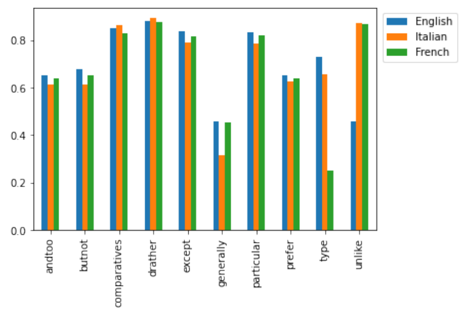
(a) Precision



(b) Recall

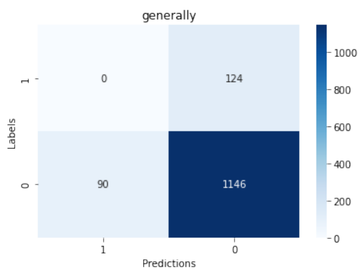


(c) F1

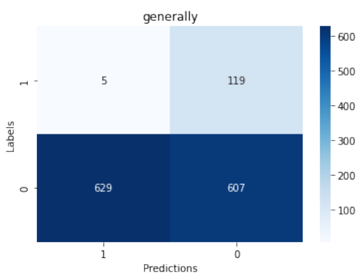


(d) F1-macro

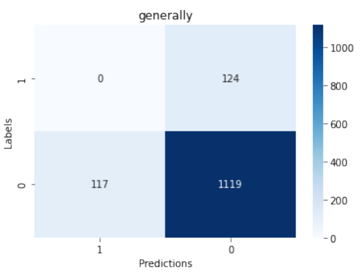
Figure 1: Comparison of the proposed approach performance on each construction in sub-task 1



(a) En-C



(b) It-C



(c) Fr-C

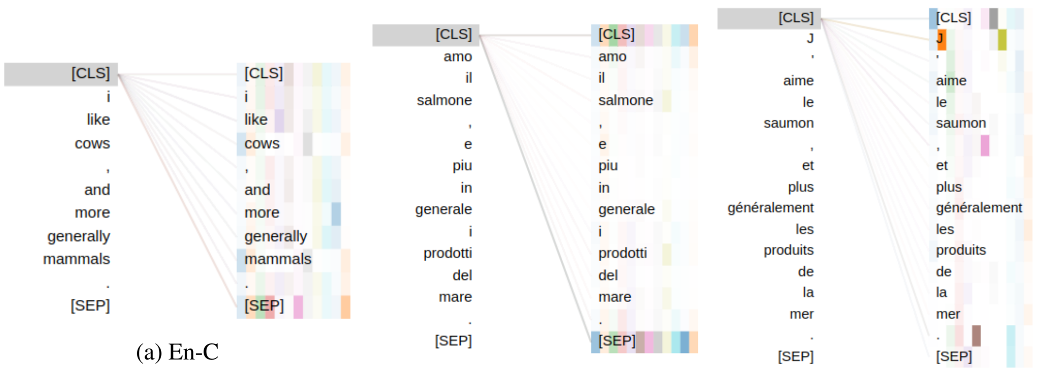
Figure 2: Confusion matrix of the results from (a) En-C, (b) It-C and (c) Fr-C on the construction "generally"

models using *bertviz*[7] library (Vig, 2019). Figure 3 illustrated the attention on the last layer of En-C, It-C and Fr-C given the acceptable sentence (labeled with 1) with "generally" construction as an input. In this figure, the attention is represented with lines connecting between the word being updated (on the left) and the word being attended to (on the right). The thickness of the lines indicates the weight. The thicker it is, the higher the weight will be. Since we use the embedding of "[CLS]" as the pooled output for fine-tuning, we only consider this token's attention. From Figure 3a, the attention weights of "and", "more" and "generally" tokens are relatively low compared to the other tokens. Similarly, the attention weights of "e", "più" "in" and "generale" in It-C and the attention weights of "et", "plus" and "généralement" in Fr-C are also low as shown in Figure 3b and 3c respectively. This suggests that these models ignored these tokens when they were fine-tuned. However, these tokens are important, since they act like keywords indicating the presuppositional "generally". Therefore, ignoring them may result in mistakenly predicting the acceptability labels of this construction.

4.2 Sub-Task 2 Results and Discussion

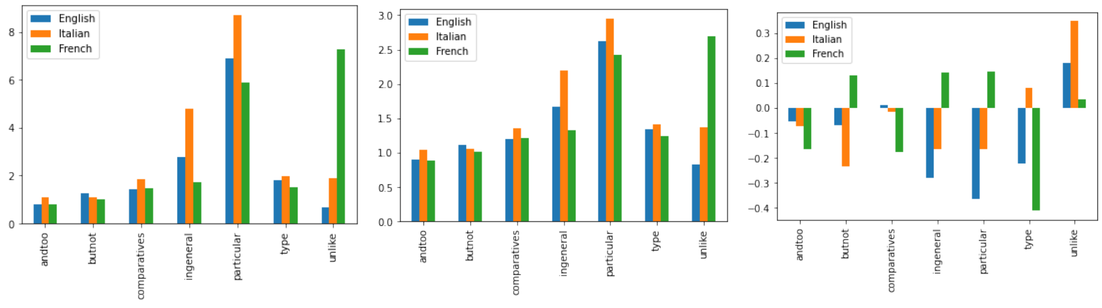
For sub-task 2, we named the fine-tuned DistilBERT-Base-Uncased for regression as **En-R**, the fine-tuned BERT-Base-Italian-XXL-Uncased for regression as **It-R** and the fine-tuned FlauBERT-Base-Uncased for regression as **Fr-R**. The overall results are shown in Table 2. From this table, our approaches outperformed the baselines in terms

[7]https://github.com/jessevig/bertviz



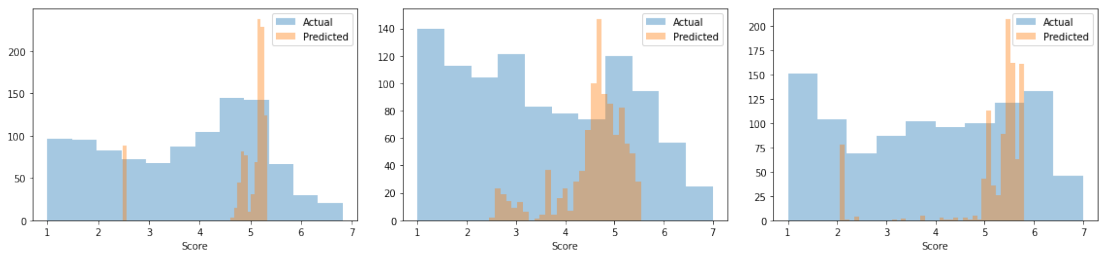
(a) En-C (b) It-C (c) Fr-C

Figure 3: Attention weights connecting with "[CLS]" token from the last layer of (a) En-C, (b) It-C and (c) Fr-C when the sentence with "generally" construction was given as an input



(a) MSE (b) RMSE (c) ρ

Figure 4: Comparison of the proposed approach performance on each construction in sub-task 2



(a) English (b) Italian (c) French

Figure 5: Distributions of the actual acceptability scores and the predicted acceptability scores of our approaches En-R, It-R and Fr-R on each test set (a) English, (b) Italian and (c) French respectively

of MSE and RMSE. Nonetheless, only En-R produced the results with the higher ρ than the baseline while the others failed to compete with their baselines. This suggests that the proposed approaches predicted the acceptability scores close to their actual scores but their ranks are dissimilar. Figure 5 shows the distributions of the actual acceptability scores and the predicted acceptability scores of our approaches, En-R, It-R and Fr-R on each test set, English, Italian and French. From this figure, we can see that all En-R, It-R and Fr-R tended to predict scores with low variances. This is possibly caused by using the mean squared error loss for fine-tuning these models.

As in sub-task 1, we also compared the results of them on each construction as shown in Figure 4. From Figure 4a and 4b, in terms of MSE and RMSE, En-R and It-R performed well on most constructions except "ingerneral" and "particular". Fr-R performed also well on almost every construction except "particular" and "unlike". On the other hand, in terms of ρ, the proposed models failed on most of the constructions as shown in 4c. En-R produced positive ρ on only "comparative" and "unlike". It-R only produced positive ρ on "type" and "unlike". Fr-R produced positive ρ on "butnot", "ingeneral", "particular" and "unlike". Overall, our models produced negative ρ in most of the constructions. This indicates that they failed to predict the acceptability scores with the same tendency as the actual scores. One possible reason is that the added regression layers are not suitable for fine-tuning these models.

5 Conclusion

This work proposes to fine-tune the pre-trained BERT-based models to validate taxonomic relations in different presuppositional constructions. Three different pre-trained BERT-based models are selected and fine-tuned to perform classification and regression on three different languages, English, Italian and French. According to the results, the fine-tuned models using the binary cross-entropy loss for classification are effective compared to the baseline. As for the regression subtask, the fine-tuned models using the mean squared error loss for regression performed less effectively than the baseline when evaluated with Spearman Correlation. This might be the result of using the mean squared error loss for fine-tuning. This leaves room for improvement in fine-tuning the BERT-based models for taxonomic relation regression.

References

Óscar Araque, Ganggao Zhu, and Carlos Angel Iglesias. 2019. A semantic similarity-based perspective of affect lexicons for sentiment analysis. *Knowl. Based Syst.*, 165:346–359.

Olivier Bodenreider. 2004. The unified medical language system (umls): Integrating biomedical terminology. *Nucleic acids research*, 32:D267–70.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2020. Flaubert: Unsupervised language model pre-training for french. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach.

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

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

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

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

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1744–1753, Sofia, Bulgaria. Association for Computational Linguistics.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. Semeval-2022 task3 (pretens): Evaluating neural networks on presuppositional semantic knowledge. In *Proceeding of SEMEVAL 2022*.

SPDB Innovation Lab at SemEval-2022 Task 3: Recognize Appropriate Taxonomic Relations Between Two Nominal Arguments with ERNIE-M Model

Yue Zhou, Bowei Wei, Jianyu Liu, Yang Yang
Shanghai Pudong Development Bank
{zhouy93, weibw1, liujy27, yangy103}@spdb.com.cn

Abstract

Synonyms and antonym practices are the most common practices in our early childhood. It correlated our known words to a better place deep in our intuition. At the beginning of life for a machine, we would like to treat the machine as a baby and build a similar training for it as well to present a qualified performance. In this paper, we present an ensemble model for sentence logistics classification, which outperforms the state-of-art methods. Our approach essentially builds on two models including ERNIE-M and DeBERTaV3. With cross-validation and random seed tuning, we select the top performance models for the last soft ensemble and make them vote for the final answer, achieving the top 6 performance.

1 Introduction

Synonym, antonym and their relations from unstructured text are fundamental problems in information classification field. These problems can be decomposed into three subtasks: word extraction using regrex, relation extraction (Zelenko et al., 2003), (Bunescu and Mooney, 2005), and classifying the logistics between them. However, an end-to-end model, i.e. ERNIE-M model (Ouyang et al., 2020), is proposed to solve the three tasks.

Presupposed Taxonomies - Evaluating Neural-network Semantics (PreTENS) (Zamparelli et al., 2022) is a task to predict the acceptability of simple sentences containing constructions whose two arguments are presupposed to be or not to be in an ordered taxonomic relation. In this paper, we first present a simple approach with the ERNIE-M model to solve the task. Although the ERNIE-M model performs unexpectedly impressive, the model has poor robustness. Hence, the additional pre-trained model is introduced to solve the robustness problem. The latest model DeBERTaV3 (He et al., 2021) has outstanding performance on cross-linguistic tasks, which outperforms BERT and DeBERTa on many tasks. The proposed model consists of two parts: the basic ERNIE-M model and the pre-trained model DeBERTaV3. The DeBERTaV3 model shares the same pre-trained data with ERNIE-M called XNLI (Conneau et al., 2018), which can improve the performance and robustness as well. The DeBERTaV3 model is trained independently, which has significant improvement for English but somehow brought no improvement for other languages. Based on the above conclusion, we employ the DeBERTaV3 model for English-task only.

To better understand the effectiveness of the proposed model, we started a bunch of analyses. The first problem is the data-set limitation. Two additional datasets were imported, i.e., the translated dataset from Google translation which is translated from three languages, and the XNLI dataset. However, larger datasets don't lead to better performance. We compared the performance of the ERNIE-M model on four sets of data: the given data, the given data with translated data, the given data with XNLI augmentation, and the given data with both the translated data and XNLI data. We do the same experiments with the DeBERTaV3 model as well. The results show that the combination of ERNIE-M with all the three datasets and DeBERTaV3 with the given English data perform the best.

2 Related Work

Multilingual model ERNIE-M proposes a new training method that encourages the model to align the representation of multiple languages with monolingual corpora, to overcome the constraint that the parallel corpus size places on the model performance. There are two models in ERNIE-M which are Cross-Attention masked language modeling (CAMLM) and Back-Translation masked language modeling (BTMLM).

Cross-Attention masked language modeling (CAMLM) is to align cross-language semantic representations on parallel corpora. Then, the multilingual representation is enhanced with transferability learned from parallel corpora.

Back-Translation masked language modeling (BTMLM) is trained to generate pseudo-parallel sentences from monolingual sentences. The generated pairs are then used as the input of the model to further align the cross-lingual semantics, thus enhancing the multilingual representation.

DeBERTaV3 presents a new pre-trained language model, which improves the original DeBERTa model by replacing mask language modeling (MLM) with replaced token detection (RTD), a more sample-efficient pre-training task. They all come from an important field, multilingual models.

Since the related paper was published at the end of 2021, there are no similar tasks have been done and published.

3 Our Approach

In this section, we first introduce the methods to solving the multi-language problem and then present our work about improving the performance on uni-language. To extenuate over-fitting for a specific language, our team uses a multi-language ensemble learning strategy that includes a pre-trained language model and a multilingual language model. Based on the approach above, it makes the learned representation generalizable across languages and improves the performance in finding the suitable taxonomic relations in two nominal arguments.

3.1 Multilingual Language Model Training

Our key idea of solving multilingual language tasks is to learn the language invariant feature space shared among multiple languages. We tried multilingual masked language modeling (MMLM), translation language modeling (TLM), and cross-attention masked language modeling (CAMLM) have been tried. However, the scale of the parallel corpus is quite limited, which limits the performance of the model.

However, we found that using the transferability learned from parallel corpora to enhance the model's learning of large-scale monolingual corpora to enhance multilingual semantic representation can achieve a good effect. ERNIE-M does this by making the predictions of tokens depending on tokens in another language, but not on other tokens in this language. Therefore, we choose ERNIE-M as the baseline model for this task and explore on this basis to improve the prediction effect.

In the process of using multilingual language models, we mainly adopt random search to fine-tune the ERNIE-M model and data augmentation methods are used for model training. Cross-lingual natural language inference (XNLI) dataset is used and the English training set is translated to Italian (E2I set). Firstly, the English training set is combined with the French and E2I set. Then, the model is fine-tuned with the combined training set. Finally, the augmented task training set in three languages is adopted for fine-tune process.

3.2 Cross-validation

To improve the robustness of our model, our team apply cross-validation for training. Firstly, by using different random seeds, we divided the training set which included all three languages ten times. Through this process, we obtained 10 folds of data, which contain 15768 training samples and 1751 validation samples in each fold. During the fine-tuning process, we used random search to optimize hyper-parameters like epochs, learning rate, and batch size. By using F1-Score as our evaluation metric, the best model at all the ten-fold of training is saved. Finally, by making predictions on the test set, we save the mean of the probability of all ten best-saved models. This result is our final output of ERNIE-M. Cross-validation process is shown in Figure 1.

3.3 Pre-trained Language Model

To enhance the effect in a single language subtask, we consider using an enhanced mask decoder and a disentangled attention mechanism to improve the effect. DeBERTaV3 meets our needs by using Electra-style pre-training and gradient unwrapping embedding sharing. We have tried to use DeBERTaV3 for training in each single language subtask respectively.

3.4 Ensemble

By using the multilingual language model and pre-trained language model respectively, we have two groups of validation set results for each language. We adopt the mean of the best-saved models from ERNIE-M and DeBERTaV3 after making predictions on the validation set. After comparing the

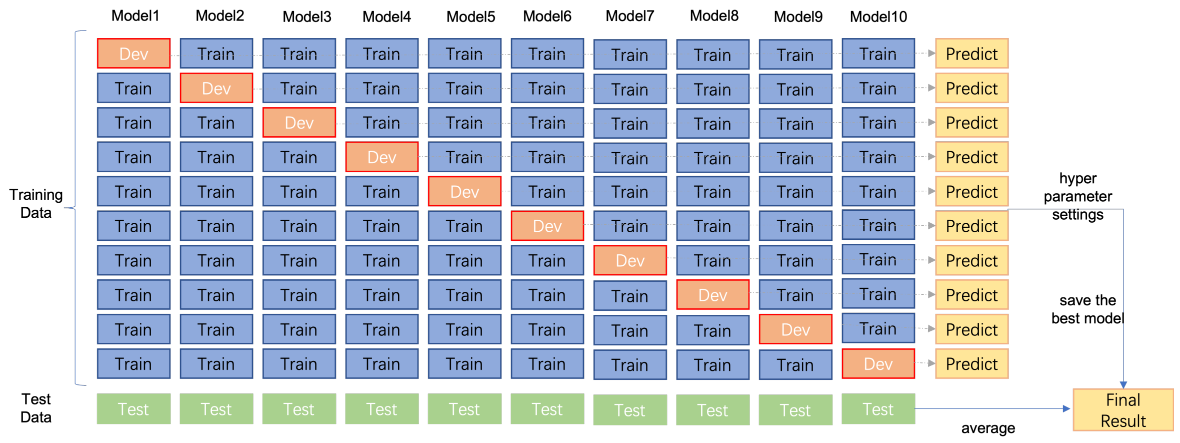


Figure 1: The process of 10-fold cross-validation and ensemble. The training set which includes all three languages is divided randomly 10 times by setting different random seeds. In each division, the training set is divided into 10 parts, of which 9 parts are respectively used as the training set and the remaining 1 part is used as the validation set. And finally, the average of all saved best models predicted on the test set is the final results.

combination result, we finally used different strategies in different languages. For the English subtask, we retain the strategy of merging the two types of models. For French and Italian subtasks, the result from cross-validation of the multilingual language model is used directly.

3.5 Data Augmentations

As the total number of labeled data in each language is only 5840, it's liable to overfit the training data even with pre-trained models. The overfitting phenomenon may be more significant than expected because the data is generated programmatically through manually verified templates. To increase the size of training data, we use the following data augmentation methods: 1) translate English data into French and Italian by using Baidu translate 2) translate English data into French and Italian by using Google translate 3) translate French and Italian data into English by using Google translate. We find that the augmentation can help delay the overfitting occurrence slightly, especially for large models.

4 Experiments

In this section, we first describe the dataset and our data preprocessing steps, and then we present the details of the experimental setup for subtask1.

4.1 Dataset

Our dataset comes from two parts.

The first part is the trial dataset released by organizers, which is composed of English, French and Italian. Each language contains 5838 sentences. Because the trail dataset provided by organizers is only 5838 in each language, to increase the amount of data and make the model better, we use Google translator and Baidu translator to translate the English dataset into French and Italian again. The use of two different translators also increases the diversity of data.

The other part is that we use the public dataset – XNLI. We use XNLI dataset because it is often used in similar cross-language tasks. The XNLI dataset contains a total of 15 languages, and each language contains 7500 pairs of data. We used the English and French datasets in this competition. Because the XNLI dataset itself does not contain Italian datasets, we translated the English dataset into Italian and then used the three languages in ERNIE-M model training.

4.2 Experiment Settings

In this task, we mainly use the ERNIE-M model and DeBERTaV3 model. The ERNIE-M model is composed of 24 layers, 1024 hidden, and 16 heads. In terms of parameter selection, we set a set of parameters, as Table 2 shows.

We set up 10000 times of ERNIE-M model training, in which the specific values of the above parameters are randomly selected according to the table1 at each training. And in each training process, the training method of 10 folds cross-validation is used.

The DeBERTaV3 model is composed of 12 layers and a hidden size of 768. It has only 86M

Parameter	Value1	Value2	Value3	Value4
batch size	8	16	32	
lr decay	0.8	0.85	0.9	0.95
rdrop	1.0	3.0	5.0	7.0
epoch	2	4		
learning rate	2e-5	3e-5	4e-5	5e-5
dropout	0.1	0.2		

Table 1: Parameter Setting. We set different specific values of different parameters according to some previous experience. Because the appropriate value is not fixed in different tasks, we choose to use the random combination of various values of the above parameters for model training, so as to find the most appropriate parameter value and obtain the optimal model result.

Training Methods	Language	Precesion	Recall	F1
ERNIE-M	English	0.8240	0.9547	0.8846
ERNIE-M	French	0.8185	0.9402	0.8751
ERNIE-M	Italian	0.8163	0.9307	0.8698
Ensemble Model	English	0.9266	0.9605	0.9432
Ensemble Model	French	0.8125	0.9489	0.8754
Ensemble Model	Italian	0.8081	0.9467	0.8719

Table 2: Results of different models. We select the best model from a large number of randomly generated parameter training models and compare it with the final best ensemble result. And we can see that the performance of the three languages has been improved.

backbone parameters with a vocabulary containing 128K tokens which introduces 98M parameters in the Embedding layer. And we set batch size to 8, learning rate to 2e-5, and epoch to 3.

4.3 Main Results

The best single model on the development set is the ERNIE-M LARGE. And the model that uses DeBERTaV3 doesn't perform well in French and Italian, so we just use the results on English data. The best ensemble model on the test set is trained on both the XNLI dataset and the trial dataset. The ensemble model obtained English test set F1 scores of 94.325, French test set F1 scores of 86.792, and Italian test set F1 scores of 88.807. The ensemble model achieves the F1 score of 94.325 in English data, the F1 score of 86.792 in French data, and the F1 score of 88.807 in the Italian data. The results are shown in Table 2.

For the comparative analysis of the results of using only ERNIE-M as the baseline model and the ensemble model, we can see that the improvement of the ensemble model in English is relatively obvious, but the improvement in Italian and French is very weak. We think this is due to the following reasons: Firstly, Italian is not included in the original XNLI dataset. In this task, we translate English into Italian. So to a certain extent, the understanding of English by the ERNIE-M model is increased. Secondly, because DeBERTaV3 performs well in English, we only use its results in English, So the results for Italian and French did not get a big boost. This also shows that using the ensemble model can indeed improve the prediction. In the future, we will explore ensemble models that can improve predictions in Italian and French.

5 Conclusion

To solve the problem of judging whether the meaning of a sentence is self-consistent in multilingual language tasks, that is, the problem raised in task 3, we propose an ensemble model using ERNIE-M and DeBERTaV3, and regard this problem as a binary classification problem. Furthermore, to solve the issue of the small dataset, we use various strategies, such as K-ford cross-validation, translating the dataset using different translators, and introducing an external dataset - XNLI, a dataset commonly used in multilingual problems. In future efforts, we plan to further improve our model from these aspects. The first is to enrich the data, especially Italian and French, to help the model learn better. The second is that we could train more models on standard fine-tuning, multi-step fine-tuning, multi-task learning, or adversarial training. Then try to ensemble different models to gain a better performance.

References

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, page 724–731, USA. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

P. He, J. Gao, and W. Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv e-prints*.

X. Ouyang, S. Wang, C. Pang, Y. Sun, and H. Wang. 2020. Ernie-m: Enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell' Orletta, Arid Hasan, and Giulia Venturi. 2022. Semeval-2022 task3 (pretens): Evaluating neural networks on presuppositional semantic knowledge. In *Proceeding of SEMEVAL 2022*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3(3):1083–1106.

UU-Tax at SemEval-2022 Task 3: Improving the generalizability of language models for taxonomy classification through data augmentation

Injy Sarhan[1,2], Pablo Mosteiro[1], and Marco Spruit[1,3,4]

[1]Department of Information and Computing Sciences, Utrecht University, The Netherlands.
[2]Arab Academy for Science, Technology, and Maritime Transport, Egypt.
[3]Department of Public Health and Primary Care, Leiden University Medical Center, The Netherlands.
[4]Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands.
i.a.a.sarhan@uu.nl ,p.mosteiro@uu.nl m.r.spruit@lumc.nl

Abstract

This paper presents our strategy to address the SemEval-2022 Task 3 PreTENS: Presupposed Taxonomies Evaluating Neural Network Semantics. The goal of the task is to identify if a sentence is deemed acceptable or not, depending on the taxonomic relationship that holds between a noun pair contained in the sentence. For sub-task 1—binary classification—we propose an effective way to enhance the robustness and the generalizability of language models for better classification on this downstream task. We design a two-stage fine-tuning procedure on the ELECTRA language model using data augmentation techniques. Rigorous experiments are carried out using multi-task learning and data-enriched fine-tuning. Experimental results demonstrate that our proposed model, UU-Tax, is indeed able to generalize well for our downstream task. For sub-task 2—regression—we propose a simple classifier that trains on features obtained from Universal Sentence Encoder (USE). In addition to describing the submitted systems, we discuss other experiments that employ pre-trained language models and data augmentation techniques. For both sub-tasks, we perform error analysis to further understand the behaviour of the proposed models. We achieved a global $\text{F1}_{\text{Binary}}$ score of 91.25% in sub-task 1 and a rho score of 0.221 in sub-task 2.[1]

1 Introduction

Predicting the semantic relationship between words in a sentence is essential for Natural Language Processing (NLP) tasks. Deep neural language models accomplish outstanding results in multiple tasks involving semantics evaluation. The question posed by the shared task Presupposed Taxonomies: Evaluating Neural Network Semantics (PreTENS) is whether neural models can detect the taxonomic relationship between nouns, especially in scenarios where the pattern and/or the set of nouns in the sentence is previously unseen (Zamparelli et al., 2022). Sub-task 1 is a simpler classification task, while sub-task 2 is a more complex regression task. Both sub-tasks involve datasets in English, French and Italian. For each sub-task, teams are permitted three submissions. For each submission, the score is averaged over the three languages. The highest score from the three submissions is reported.

We propose a series of models based on pre-trained language models. We enhance the provided datasets using state-of-the-art data augmentation tools, and further increase the dataset size by employing translations. The aim of both steps is to create slightly modified versions of the sentences, such that the model can learn alternative forms of nouns and patterns.

For the classification task (sub-task 1), we obtained the 3rd place, with an $\text{F1}_{\text{Binary}}$ score of 91.25% averaged over the three languages. For the regression task (sub-task 2), we obtained the 5th place, with a Spearman's correlation coefficient ρ of 0.221 averaged over the three languages. Sub-task 2 is markedly more difficult than sub-task 1 due to sentences that can be ambiguous, such as *I like dogs, but not chihuahuas*; some humans will judge this sentence as acceptable, while some will not. We attempt to solve both tasks by employing data augmentation techniques in order to help the models understand variations in text. Our main contributions are: *(i)* we devise a special development-validation split to emulate the real situation in which the model must face new words and patterns, and *(ii)* we combine various data augmentation tools to allow the models to learn from various versions of the training dataset.

In Section 2 we present the task details and some of the related work that was done previously. In Section 3 we motivate our choice of models. The experiments we performed are in Section 4. Results and conclusions are presented in Sections 5 and 6.

[1]Our implementation of UU-Tax is publicly available at https://github.com/IS5882/UU-TAX.

2 Background

For the present task, we are provided with a list of sentences following a set of *patterns*, all of which have two slots for noun phrases. One such sentence might be: *I don't like beer, a special kind of drink*. The pattern corresponding to this sentence would be: *I don't like* [blank], *a special kind of* [blank]. Sentences are labeled according to whether the taxonomic relation between the two nouns makes sense. In sub-task 1, labels are binary; a sentence such as that shown above has a label of 1, while this sentence would have a label of 0: *I like huskies, and dogs too*. In sub-task 2, labels are continuous, ranging from 1 to 7; these scores are based on a seven-point Likert scale, judged by humans via crowdsourcing. The same dataset is presented in English, Italian and French. For sub-task 1, the training and test sets consist of 5 838 and 14 556 sentences, respectively; for sub-task 2, the training and test sets consist of 524 and 1 009 sentences, respectively.

There are two challenges to this dataset: *(i)* The test dataset is much bigger than the training dataset, and *(ii)* There are unseen patterns and noun pairs in the test set. The combination of these hampers the ability of machine learning (ML) models trained on the training set to generalize well to the test set. Indeed, that is the aim of this task: to evaluate the ability of language models to generalize to new data when it comes to inferring taxonomies.

One way to conceptualize the PreTENS task is to reformulate it as a taxonomy extraction task with pattern classification and distributed word representations. For a given sentence, extract the noun pair and the pattern from the sentence, and then determine if the taxonomic relation between the nouns matches the relations allowed by the pattern. This formulation is motivated by previous work in taxonomy construction that relied on various approaches ranging from pattern-based methods and syntactic features to word embeddings (Huang et al., 2019; Luu et al., 2016; Roller et al., 2018). As promising as this approach sounds for PreTENS, it involves manual labeling of the noun-pair taxonomic relations in the training set, as we are not allowed to use resources such as WordNet (Fellbaum, 1998) or BabelNet (Navigli and Ponzetto, 2012).

A different approach is to tackle PreTENS as a cross-over task between extraction of lexico-semantic relations and commonsense validation. There have been SemEval tasks to extract and identify taxonomic relationships between given terms (SemEval-2016 task 13) (Bordea et al., 2016), and to validate sentences for commonsense (SemEval-2020 task 4, sub-task A) (Wang et al., 2020). The aim of the common-sense validation task is to identify which of two natural language statements with similar wordings makes sense.

In the SemEval-2016 task 13, approaches related to extracting hypernym-hyponym relations to construct a taxonomy involved both pattern-based methods and distributional methods. TAXI relied on extracting Hearst-style lexico-syntactic patterns by first crawling domain-specific corpora based on the terminology of the target domain and later using substring matching to extract candidate hypernym-hyponym relations (Panchenko et al., 2016). Another team designed a semi-supervised model based on the hypothesis that hypernyms may be induced by adding a vector offset to the corresponding hyponym word embedding (Pocostales, 2016).

Participants in the SemEval 2020 commonsense validation task had an advantage over PreTENS participants: they were allowed to integrate taxonomic information from external resources such as ConceptNet (Wang et al., 2020), which eased the process of fine-tuning the language models on the down-stream task. As an example, the CN-HIT-IT.NLP team (Zhang et al., 2020) and ECNU-SenseMaker (Zhao et al., 2020) both used a variant of K-BERT (Liu et al., 2020a) with additional data; the former injects relevant triples from ConceptNet to the language model, while the later also uses ConceptNet's unstructured text to pre-train the language model. Other systems relied on ensemble models consisting of different language models such as RoBERTa and XLNet (Liu, 2020; Altiti et al., 2020).

In Section 3 we outline the architectures chosen to tackle the two sub-tasks of PreTENS. We draw on previous work, as outlined above, and provide novel combinations of datasets and algorithms to improve the performance of out-of-the box language models.

3 System Description

The systems we propose for both PreTENS sub-tasks are based on language models. In sub-task 1 we use the ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) transformer (Clark et al., 2020), while in sub-task 2 we employ USE (Universal Sentence Encoder) (Yang et al., 2020).

3.1 Sub-task 1: Classification

In the first sub-task—binary classification—we were required to assign an acceptability label for each sentence in the three languages English, French and Italian. Of the 20 394 sentences that were provided for sub-task 1, only 5 838 sentences (28.61%) were available for training. This split causes the model to be likely to encounter unknown data formats at testing time. This is a pivotal challenge in PreTENS, as the robustness and generalization of language models is an open challenge and cannot be guaranteed (Tu et al., 2020; Ramesh Kashyap et al., 2021). In our experiments we found that every language model we used (BERT, RoBERTa, XLNet, and ELECTRA) failed to generalize well to unseen datasets, even though all of them are pre-trained on large amounts of data. To address this challenge, we built our models based on data augmentation.

While designing our model, we split the provided training data into a development set (30%) and a validation set (70%), to emulate the train-test split sizes. We deliberately leave several patterns out of the development set, including, for example: *I like* [blank], *and more specifically* [blank]. We choose these so-called *complex patterns* because, during exploratory experiments, we found that pre-trained models had trouble with them. For example, out of the 820 instances of the aforementioned pattern in the training dataset, 750 instances were misclassified by one of the early instances of our model; this includes sentences where the noun pair was included in other sentences in the training data. We thus remove complex patterns from the training data, to simulate a situation in which new unseen and difficult patterns are found in the test set.

Transformer language models like BERT (Devlin et al., 2019) are pre-trained on two tasks: Masked Language Modelling (MLM) and Next Sentence Prediction (NSP). However, in subsequent models such as RoBERTa, training on NSP was proven to be unnecessary; these models are thus pre-trained solely on MLM. ELECTRA further enhanced MLM performance while utilizing notably less computing resources for the pre-training stage. The pre-training task in ELECTRA is built on discovering replaced tokens in the input sequence; to achieve this, ELECTRA deploys two transformer models: a generator and a discriminator, where the generator is trained to substitute input tokens with credible alternatives and a discriminator to predict the presence or absence of substitution. This setting is similar to Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), with a key difference that the generator does not attempt to trick the discriminator, making ELECTRA non-adversarial. In ELECTRA, the generator parameters are only adjusted during the pre-training phase. Fine tuning on downstream tasks only modifies the discriminator parameters (Clark et al., 2020).

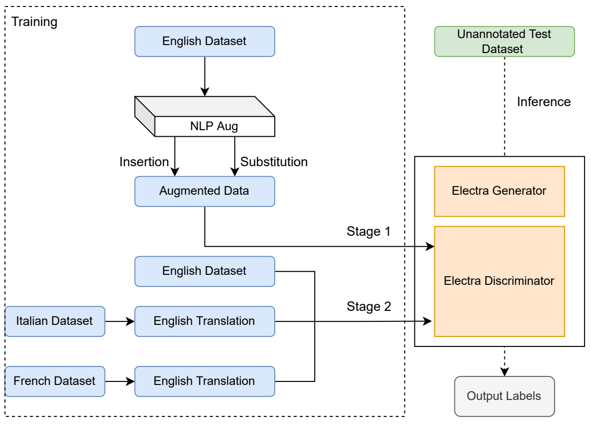


Figure 1: Sub-task 1: The English version of the proposed two-stage fine-tuning model (UU-Tax). In the French version, the Italian and English data are translated to French, and the NLPAug tool is employed on the provided French training set. Likewise in the Italian version.

Multi-stage fine-tuning has proven its effectiveness on the robustness and generalization of models (Kocijan et al., 2019; Li and Rudzicz, 2021). We perform a 2-stage fine-tuning; Figure 1 portrays our model work-flow. In the first stage, we use the NLPAug tool (Ma, 2019) to generate new sentences by making modifications to existing sentences based on contextualized word embeddings. There are several actions for the NLPAug tool; we utilize the 'Insertion' and 'Substitution' operations. The 'Insertion' operation picks a random position in the sentence, and then inserts at that position the word that best fits the local context. Meanwhile, the 'Substitution' operation replaces a word in a given sentence by the most appropriate alternative for that word. In both operations, the word choice is given by contextualized word embeddings, as will be explained in Section 4.1. To avoid drifting away from the original sentence, in both operations we limit the number of insertions and substitutions to two. Because 'Substitution' in NLPAug might turn an incorrect sentence into a correct one, we only carry out 'Substitution' on sentences labeled 1. An example of the output of the NLPAug tool is shown in Figure 4 in Appendix A.

The second stage of fine-tuning also involves data augmentation, using translation. For each language l, we translate the datasets of the other two languages into l. For example, as seen in Figure 1, when working on the English model, we translate the Italian and French datasets to English, and perform the second fine-tuning stage on the translated data along with the original data. We use the Google Translate API for all translations [2].

3.2 Sub-task 2: Regression

In sub-task 2—regression—we are required to determine the level of acceptability of sentences on a seven-point Likert scale. Our initial attempt in sub-task 2 resembles the efforts made in the first sub-task by relying on pre-trained language models. However, our first submission, which relies on fine-tuning multi-lingual BERT (Devlin et al., 2019) with translation as data augmentation, did not perform well; more elaboration on this in Section 5.2. As a result, we opt for a simpler yet more effective model using Universal Sentence Encoder (USE) (Yang et al., 2020) followed by a regressor. USE is based on two encoder models and deep averaging networks; both are equipped to generate a 512-dimension sentence embedding from a given textual input, where embeddings for words and bigrams are averaged together and then passed as input to a deep neural network that processes and outputs the sentence embeddings.

4 Experimental Set-up

4.1 Sub-task 1: Classification

We implement our submitted models using SimpleTransformers[3]. All models are trained for 4 epochs with a batch size of 8; these values were determined by validation, as we explain below. The model is optimized using AdamW (Loshchilov and Hutter, 2019) and a linear decay learning rate schedule. The learning rate is a key aspect of the performance of a trained model. A large learning rate results in quick model convergence; however, if the learning rate is too large, it will lead to drastic updates that will trigger divergent behaviour, while training a model with a too-small learning rate might lead to an under-fitted model that gets stuck in local minima (Bengio, 2012). In our two-stage model, the first stage has a lower learning rate of 3×10^{-5} as opposed to the 4×10^{-5} assigned in the second stage, which contains the PreTENS training data; this is because we want the model to learn more from the real training data than from the NLPAug-edited data. A summary of the model hyper-parameters is given in Table 1. All the hyper-parameters are tuned based on the F1 score on the validation set. The same hyper-parameters are utilized for all three languages—English, French and Italian.

For data augmentation with NLPAug, $\text{BERT}_{\text{base}}$ is employed to obtain the contextual word embeddings for both 'Insertion' and 'Substitution' operations.

Hyper-parameter	**Value**
Epochs	4
Batch Size	8
Stage 1 Learning Rate	3×10^{-5}
Stage 2 Learning Rate	4×10^{-5}
Optimizer	AdamW

Table 1: Sub-task 1: Hyper-parameters values for training the ELECTRA model. The number of epochs and the batch size were determined by validation.

4.2 Sub-task 2: Regression

For the three languages English, French and Italian we deploy multi-lingual $\text{USE}_{\text{Large}}$ as it yields better performance than mono-lingual USE for the three languages. USE is employed through its TensorFlow hub module[4]. We experiment with four different regressors: Linear Regression (LR) (Montgomery et al., 2021), K-Nearest Neighbors Regressor (KNR) (Kramer, 2013), Decision Tree (DT) (Myles et al., 2004), and Support Vector Regressor (SVR) (Awad and Khanna, 2015). We use the Scikit-Learn (Pedregosa et al., 2011) library for the implementation of the regressors. All regressors are utilized with their default parameters except for SVR epsilon ε . To define a higher margin of tolerance where no penalty is given to errors we set ε to 0.2 rather than the default value of 0.1.

[2]Only 15% of the translated sentences using Google Translate API were duplicates of the original sentence.

[3]https://github.com/ThilinaRajapakse/simpletransformers

[4]https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3

4.3 Evaluation measures

Sub-task 1 is evaluated using the Binary-averaged F1 score ($F1_{\text{Binary}}$) for each language, while the global rank score is calculated as the average of the $F1_{\text{Binary}}$ for all three languages. Sub-task 2 is evaluated using Spearman's rank correlation coefficient (ρ) for each language, with the global rank given by the average of the coefficients for all languages.

5 Results and Evaluation

In this section, we analyze the performance of our submitted models in both sub-tasks. We further discuss other notable experiments that were carried out.

5.1 Sub-task 1: Classification

Language	**Results**		
	Recall	**Precision**	**$F1_{\text{Binary}}$**
English	95.26 %	90.54 %	92.84 %
French	93.14 %	85.83 %	89.34 %
Italian	90.47 %	92.69 %	91.57 %
Average			91.25%

Table 2: Sub-task 1: UU-Tax submission results using a two-stage fine-tuned ELECTRA model.

Results of the submitted models for English, French, and Italian are shown in Table 2. Out of 21 teams, we were officially ranked 3rd in sub-task 1, achieving a global score of 91.25%, only 1.06, and 2.92 percentage points short of the 2nd and 1st places, respectively. In the next few sections, we explain how our experimentation led us to the model we chose: the two-stage fine-tuning using ELECTRA with data augmentation (UU-Tax).

5.1.1 Experiments

Baseline. The PreTENS organizers proposed a baseline algorithm that trains an SVM classifier with features generated by TF-IDF with n-grams ($n = 3$). Results of the baseline model are reported in Table 3.

Multi-task fine-tuning. We experimented with several models on the English dataset. We tried a multi-task approach that involves further fine-tuning on related data-rich supervised tasks. In our case, it was the 'common sense validation' task, as it is highly correlated to PreTENS as previously mentioned in Section 2. We used the dataset from SemEval-2020 Common Sense Validation sub-task A (Wang et al., 2020) and modified the sentence label to 1 if it is a valid sentence and 0 otherwise. We then fine-tuned our ELECTRA model in the first stage using this data; the second stage of fine-tuning was carried out using the augmented data from NLPAug and the provided training data. Multi-task fine-tuning has proven its effectiveness across a variety of tasks (Mahabadi et al., 2021). This model achieved an $F1_{\text{Binary}}$ of 89.09%, which demonstrates the effect of information sharing between the different tasks, particularly in cases when the downstream task is of a limited size. Nevertheless, multi-task fine-tuning suffers from several shortcomings including catastrophic forgetting, over-fitting in low-resource tasks and under-fitting in high-resource tasks (Mahabadi et al., 2021). For this reason, we did not move forward with this approach.

Data-enriched fine-tuning. As an alternative, we developed a data-enriched fine-tuning model that employed a pre-trained BERT model with an additional Bidirectional Long Short Term Memory (Bi-LSTM) layer on top. In addition to the input sentence, we concatenated the two nominal arguments to the given input. To extract the two nouns from the sentences, we leveraged the fact that nouns in this dataset tend to have very low document frequencies (DF), and classified any word with DF less than 5% as a noun. The final prompt of the input was as follows: $[\text{CLS}]Sentence[\text{SEP}]Noun\,1[\text{SEP}]Noun\,2[\text{SEP}]$ Similar to the aforementioned models, we also input to the model the augmented data generated from NLPAug. This model was implemented with PyTorch using the Hugging Face[5] Transformers library (Wolf et al., 2019). Figure 2 depicts the data-enriched fine-tuning model. The model's performance resembles that of the multi-task fine-tuning model by achieving an $F1_{\text{Binary}}$ of 89.04%.

As shown in Table 3, our submitted two-stage fine-tuning ELECTRA model (UU-Tax) achieved the highest results amongst all models, by a margin of 3.63% and 4.62% between both multi-task learning model and data-enriched fine-tuning model, respectively. We have almost 20% improvement compared to the baseline.

Model	**Results**		
	Recall	**Precision**	**F1$_{\text{Binary}}$**
Baseline (TF-IDF + SVR)	85.64 %	64.19 %	73.38 %
Multi-task fine-tuning	**95.82** %	83.45 %	89.21 %
Data-enriched fine-tuning (BERT + Bi-LSTM)	86.70 %	89.79 %	88.22 %
UU-Tax (two-stage ELECTRA)	95.26 %	**90.54 %**	**92.84 %**

Table 3: Sub-task 1: Comparison of the different experiments carried out on the English Language.

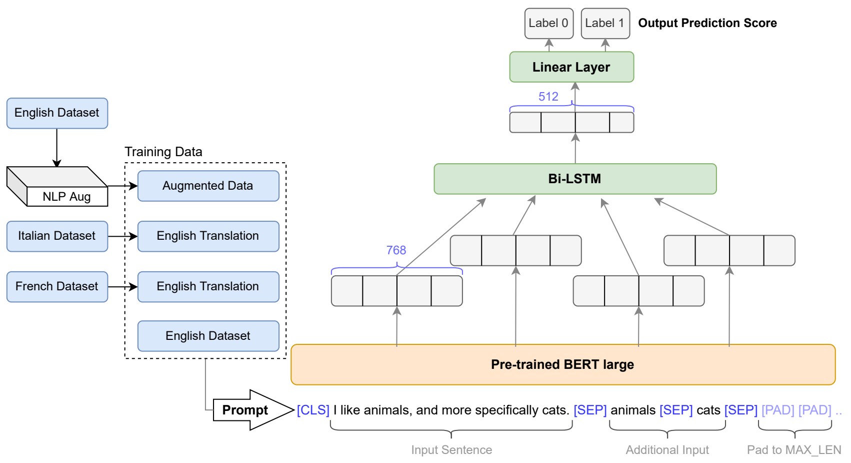


Figure 2: Sub-task 1: data-enriched fine-tuning model that employs an Bi-LSTM network on the top of pre-trained BERT. This model was used during the experimentation phase.

5.1.2 Ablation Study and Error Analysis

We conducted ablation experiments to evaluate the effect of data augmentation and our proposed two-stage fine-tuned ELECTRA model. The results of the analysis are presented in Table 4. We limit the ablation study and error analysis to the English dataset, as similar trends were observed in the French and Italian datasets [6].

Data augmentation effect. The need for data augmentation to generalize the model highly affects the performance of the pre-trained model. We perform two ablation analyses. In the first setting *(Ablation #1)*, we removed the translated dataset from the second stage, and our model was fine-tuned on data obtained from the NLPAug tool in the first stage and on the original training dataset in the second stage. The precision massively dropped by 11.42%. Similar behavior is observed in the second setting *(Ablation #2)*, when the NLPAug data is eliminated from our two-stage training, and the first stage is trained on the translated data instead, while in the second stage we fine-tuned using the original training data. This highlights the importance of our proposed dual augmentation using both NLPAug and translation to capture a wider range of perturbations to the original dataset.

Single-stage models' performance. To verify our two-stage fine-tuning approach, we evaluated it against a single-stage fine-tuning. This experiment was performed in two different settings; in the first *(Single-stage #1)* we trained on the originally provided data only, while in the second *(Single-stage #2)* setting we trained on the same data that was used in UU-Tax, which is obtained from NLPAug, translation, and the original training set. In both settings, we notice a drop in the F1 when comparing against UU-Tax. Nonetheless, we can observe that amongst the three experiments (UU-Tax, *Single-stage #1* and *Single-stage #2*) the highest recall of 96.26% is achieved in the (*Single stage #2*) along with the lowest precision of 71.15%. Our interpretation of this finding is that in the *(Single stage #2*) experiment, the model over-predicted positives, causing the model to achieve a high recall and a relatively low precision.

[5]https://huggingface.co/

[6]Results presented in Tables 3 and 4 may slightly vary due to fine-tuning instability of pre-trained language models (Mosbach et al., 2021).

Model Name	LM	Stage 1			Stage 2			Results		
		NLPAug	Trans	OT	NLPAug	Trans	OT	R	P	F1
Ablation #1	ELECTRA	✓					✓	95.30 %	78.73 %	86.22 %
Ablation #2	ELECTRA		✓				✓	95.59 %	95.59 %	86.58 %
Single Stage #1	ELECTRA			✓	-	-	-	90.20 %	79.41 %	84.47 %
Single Stage #2	ELECTRA	✓	✓	✓	-	-	-	**96.26 %**	71.16 %	81.83 %
Two-Stage #1	BERT	✓				✓	✓	92.36 %	68.97 %	78.97 %
Two-Stage #2	RoBERTa	✓				✓	✓	93.93 %	78.24 %	85.37 %
UU-Tax	ELECTRA	✓				✓	✓	95.26 %	**90.54 %**	**92.84 %**

Table 4: Sub-task 1: Results of various classification models trained during experimentation and ablation on the sub-task 1 dataset, using different combinations of input data obtained from NLPAug, translation (Trans) and the original training set provided (OT). Additional variations are single-stage versus two-stage models, and alternative pre-trained language models (LM). Recall (R), precision (P), and $F1_{\text{Binary}}$ (F1) are used as evaluation metrics. ✓ indicates which data is utilized in each fine-tuning stage, while - indicates that stage 2 is not applicable.

We attribute this behavior to two causes. First, the unbalanced ratio that NLPAug 'Substitution' operation caused as previously explained in Section 3.1[7]. Second, in UU-Tax a higher learning rate is deployed in the second fine-tuning stage, making the model focus more on the original dataset than on the NLPAug data.

Experimenting with different language models. Additionally, we experimented with different pre-trained language models, namely BERT *(Two-stage #1)* and RoBERTa *(Two-stage #2)*. As seen in Table 4, ELECTRA outperforms both RoBERTa and BERT by 7.47% and 13.92%, respectively, of the F1 score, which illustrates the strong generalizability of ELECTRA. Our findings agree with (Anaby-Tavor et al., 2020; Kumar et al., 2020), who demonstrate that generative models are suitable for data augmentation.

Error Analysis. By manually inspecting the wrong predictions generated by our proposed top three performing models (UU-Tax, multi-task fine-tuning, and data-enriched fine-tuning) we can observe that UU-Tax achieves the smallest percentage of incorrect predictions on both seen and unseen patterns, as observed in Figure 3. This shows that the proposed two-stage fine-tuning (UU-Tax) can learn better and generalize better than multi-task fine-tuning and data-enriched fine-tuning. In addition, we also noticed that proper names were the cause of many misclassifications. One possible mitigation to overcome this error is to create an improved model to envision proper names appearing in a sentence as hyponyms of the preceding or the subsequent noun appearing in the same sentence.

[7]The NLPAug 'Substitution' dataset is composed of 5568 instances all labeled '1', making 67.98% of the NLPAug data to have a '1' label.

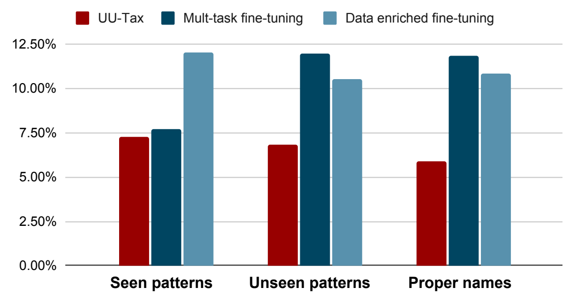


Figure 3: Sub-task 1: Percentage of incorrect predictions for all patterns in the test dataset, for the top three preforming models: UU-Tax, Multi-task fine-tuning and data-enriched fine-tuning.

5.2 Sub-task 2: Regression

Language	Model	Rho (ρ)
English	USE + SVR	0.478
French	USE + DT	-0.059
Italian	USE + LR	0.246
Average		0.221%

Table 5: Sub-task 2: UU-Tax submission results that achieved the highest score averaged over the three languages, out of the three submissions. ρ is Spearman's rank correlation coefficient.

As explained in Section 3.2, USE was employed for all three languages to obtain pre-trained word embeddings; we used SVR, DT, and LR regressors for English, French and Italian, respectively. We came in 5th in sub-task 2 out of 17 teams by achiev-

Language	Model						
	Baseline	BERT	BERT + Trans	USE + LR	USE + KNR	USE + DT	USE + SVR
English	0.247	-0.068	-0.027	-0.175	0.235	0.118	**0.478***
French	**0.230**	-0.075	-0.027	0.207	0.103	-0.059*	0.030
Italian	**0.370**	0.047	0.150	0.246*	0.081	0.171	0.137

Table 6: Sub-task 2: Rho (ρ) scores of different regression models that we experimented. Models that were part of the global score are marked with an * . Baseline is TF-IDF + SVR; BERT is multilingual.

ing a global average of 0.221. It is worth noting that we had a better performing French-language model in the first submission than in our top submission. The experiments we performed for sub-task 2 are discussed in Section 5.2.1. The ρ coefficients for the three languages in our best submission are reported in Table 5.

5.2.1 Experiments

Table 6 shows the results of our submitted models along with other experiments that we carried out using different regressors as explained in Section 4.2. In addition, we also experimented using multi-lingual BERT in two different settings; once with only fine-tuning on the provided dataset of the three languages and in the other setting, we augmented the provided training data with translation as in the translation process in sub-task 1.

In English our submitted USE + SVR model achieved the highest ρ score of 0.478 amongst all other models, surpassing the baseline by 94%. Although in the French version our final submitted model was, unfortunately, the model with the lowest score, we were able to achieve the highest score of 0.207 using LR, less than the baseline approach by $\Delta\rho = 0.023$. While in Italian, our submitted model was our highest rho score achieved of 0.246 which is $\Delta\rho = 0.123$ lower than the baseline. We infer from the fact that our model performed badly on French and Italian that USE is better optimized for English language.

5.2.2 Ablation Study and Error Analysis

Pre-trained language models did not perform well. We attribute this to the very limited training size of sub-task 2: only four different patterns made up the training data. The deployment of data augmentation—translation—to multi-lingual BERT was able to improve the performance on all three languages by more than 50%, which confirms our hypothesis that the limited pattern in the provided training set highly affected the performance of the pre-trained language model. This is supported by a similar trend when experimenting with different language models. Since this is a regression task, we were not able to use the NLPAug tool as the assigned score might be inaccurate after the substitution and insertion operations.

There is no consistently best performing classical ML algorithm: unlike for Italian and French, LR did not perform well on the English dataset, and SVR outperformed all other regressors on the English version. Interestingly, we see a consistent pattern across the French and Italian versions, showing that the LR regressor works best; we attribute this to the lexical and grammatical similarity between the French and Italian languages.

6 Conclusion

The limited size of the training dataset as compared to the test set made it impossible to train neural networks directly on the task. As a result, we took advantage of pre-trained language models. Nonetheless, the robustness of language models is highly affected by the size and variance of the downstream task data available for fine-tuning, which causes the language model to fail to generalize. Hereby, we relied upon data augmentation techniques using a two-stage fine-tuning process on ELECTRA. The first fine-tuning stage was carried out using an augmented version of the dataset, while in the second stage we used the translated versions of the provided PreTENS training data in addition to the original data. We ranked 3rd out of 21 teams in sub-task 1. For the second sub-task we proposed a simple model by training an SVR classifier with sentence embeddings obtained from USE; we ranked 5th out of 17 teams.

As an extension for future work, both sub-tasks could greatly benefit from adversarial training, which has proven its success across various NLP tasks in improving the model robustness and generalization (Liu et al., 2020b; Yoo and Qi, 2021).

References

Ola Altiti, Malak Abdullah, and Rasha Obiedat. 2020. Just at semeval-2020 task 11: Detecting propaganda techniques using bert pre-trained model. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1749–1755.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390.

Mariette Awad and Rahul Khanna. 2015. Support vector regression. In *Efficient learning machines*, pages 67–80. Springer.

Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. Semeval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)*, pages 1081–1091.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Subin Huang, Xiangfeng Luo, Jing Huang, Yike Guo, and Shengwei Gu. 2019. An unsupervised approach for learning a chinese is-a taxonomy from an unstructured corpus. *Knowledge-Based Systems*, 182:104861.

Vid Kocijan, Ana-Maria Cretu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. 2019. A surprisingly robust trick for the winograd schema challenge. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019*. Association for Computational Linguistics.

Oliver Kramer. 2013. K-nearest neighbors. In *Dimensionality reduction with unsupervised nearest neighbors*, pages 13–23. Springer.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Bai Li and Frank Rudzicz. 2021. Torontocl at cmcl 2021 shared task: Roberta with multi-stage fine-tuning for eye-tracking prediction. *arXiv preprint arXiv:2104.07244*.

Pai Liu. 2020. Qiaoning at semeval-2020 task 4: Commonsense validation and explanation system based on ensemble of language model. *arXiv preprint arXiv:2009.02645*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020a. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020b. Adversarial training for large neural language models. *arXiv preprint arXiv:2004.08994*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Anh Tuan Luu, Yi Tay, Siu Cheung Hui, and See Kiong Ng. 2016. Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 403–413.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *ACL/IJCNLP*.

Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. 2021. *Introduction to linear regression analysis*. John Wiley & Sons.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Anthony J Myles, Robert N Feudale, Yang Liu, Nathaniel A Woody, and Steven D Brown. 2004. An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 18(6):275–285.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial intelligence*, 193:217–250.

Alexander Panchenko, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cédrick Fairon, Simone Paolo Ponzetto, and Chris Biemann. 2016. Taxi at semeval-2016 task 13: a taxonomy induction method based on lexico-syntactic patterns, substrings and focused crawling. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1320–1327.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Joel Pocostales. 2016. Nuig-unlp at semeval-2016 task 13: A simple word embedding-based approach for taxonomy extraction. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1298–1302.

Abhinav Ramesh Kashyap, Laiba Mehnaz, Bhavitvya Malik, Abdul Waheed, Devamanyu Hazarika, Min-Yen Kan, and Rajiv Ratn Shah. 2021. Analyzing the domain robustness of pretrained language models, layer by layer. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 222–244, Kyiv, Ukraine. Association for Computational Linguistics.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst patterns revisited: Automatic hypernym detection from large text corpora. *arXiv preprint arXiv:1806.03191*.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020. Semeval-2020 task 4: Commonsense validation and explanation. *arXiv preprint arXiv:2007.00236*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Jin Yong Yoo and Yanjun Qi. 2021. Towards improving adversarial training of NLP models. *EMNLP*, abs/2109.00544.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. Semeval-2022 Task3 (PreTENS): Evaluating Neural Networks on Presuppositional Semantic Knowledge. In *Proceeding of SEMEVAL 2022*.

Yice Zhang, Jiaxuan Lin, Yang Fan, Peng Jin, Yuanchao Liu, and Bingquan Liu. 2020. Cn-hit-it. nlp at semeval-2020 task 4: Enhanced language representation with multiple knowledge triples. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 494–500.

Qian Zhao, Siyu Tao, Jie Zhou, Linlin Wang, Xin Lin, and Liang He. 2020. Ecnu-sensemaker at semeval-2020 task 4: Leveraging heterogeneous knowledge resources for commonsense validation and explanation. *arXiv preprint arXiv:2007.14200*.

A Appendix

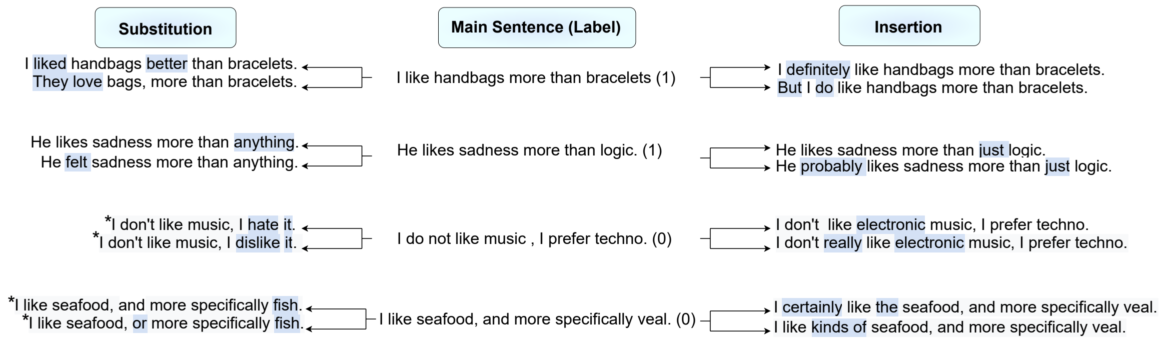

Substitution
Main Sentence (Label)
Insertion
I liked handbags better than bracelets.
They love bags, more than bracelets.
I like handbags more than bracelets (1)
I definitely like handbags more than bracelets.
But I do like handbags more than bracelets.
He likes sadness more than anything.
He felt sadness more than anything.
He likes sadness more than logic. (1)
He likes sadness more than just logic.
He probably likes sadness more than just logic.
*I don't like music, I hate it.
*I don't like music, I dislike it.
I do not like music , I prefer techno. (0)
I don't like electronic music, I prefer techno.
I don't really like electronic music, I prefer techno.
*I like seafood, and more specifically fish.
*I like seafood, or more specifically fish.
I like seafood, and more specifically veal. (0)
I certainly like the seafood, and more specifically veal.
I like kinds of seafood, and more specifically veal.

Figure 4: Sub-task 1: Example of the output generated by both, substitution and insertion operations of the NLPAug library. As explained in Section 3.1, for sentence with label 0, the substitution operation is not performed, this is indicated using an * in the figure.

KaMiKla at SemEval-2022 Task 3: AlBERTo, BERT, and CamemBERT—Be(r)tween Taxonomy Detection and Prediction

Karl Vetter
karl.vetter@student.uni-tuebingen.de

Miriam Segiet
miriam.segiet@student.uni-tuebingen.de

Klara Lennermann
klara.lennermann@student.uni-tuebingen.de

Abstract

This paper describes our system submitted for SemEval Task 3: Presupposed Taxonomies: Evaluating Neural Network Semantics (Zamparelli et al., 2022). We participated in both the binary classification and the regression subtask. Target sentences are classified according to their taxonomical relation in subtask 1 and according to their acceptability judgment in subtask 2. Our approach in both subtasks is based on a neural network BERT model. We used separate models for the three languages covered by the task, English, French, and Italian. For the second subtask, we used median averaging to construct an ensemble model. We ranked 15th out of 21 groups for subtask 1 (F1-score: 77.38%) and 11th out of 17 groups for subtask 2 (RHO: 0.078).

1 Introduction

The recognition of lexical relationships between words and the corresponding generalization has attracted increasing attention in computational linguistics. Today, there already exist resources covering manually marked semantic relationships, e.g., taxonomic relations, such as the lexical database WordNet (Miller, 1992) or the multilingual dictionary and semantic network BabelNet (Navigli and Ponzetto, 2010).

Luu et al. (2016) define taxonomic relations between two terms as an is–a relation. In such a relation there is a hypernym, i.e., a supertype, and a hyponym, i.e., a subtype. Both the supertype and subtype are sets covering, in our task, certain semantic categories. In a relation such as *animal–dog*, the *animal* is the superordinate category and the *dog* is the subordinate term. Furthermore, those specific sets are included in a relation forming a special hierarchy (Kay, 1971). As stated by Nguyen et al. (2017), in such an is–a relation the supertype necessarily implies the subtype, but not vice versa.

SemEval 2022 Task 3 is a taxonomy detection and prediction task consisting of two subtasks: a binary classification and a regression task, both covering the languages English, French, and Italian.

We propose an approach based on the transformer-based machine learning model BERT. Since BERT is a bidirectional model producing state-of-the-art results (Devlin et al., 2019), we used this pre-trained model for our analysis. For the three different languages, the corresponding BERT models were used (Devlin et al., 2019; Polignano et al., 2019; Martin et al., 2020).[1]

2 Task Description

In the present shared task (Zamparelli et al., 2022), the taxonomic sentence structures in the given files are composed of different artificially generated constructions enforcing presuppositions.

Table 1 shows example sentences provided in the English test set. According to the task description page, the French and Italian datasets are translated versions of the English dataset that were slightly adapted.

The argument nouns in the given files come from 30 semantic categories including, among others, dogs, birds, and mammals. The given word sets already show broader and narrower categories (*mammals* vs. *dogs/birds*). Nevertheless, as shown in the examples in table 1, not all pairs of nouns reflect such a superordinate–subordinate relation (*apple* vs. *cauliflower*) as described by Nguyen et al. (2017). Therefore, the taxonomies do not only represent a direct is–a relation such that one given nominal is the subcategory and the other one is the supercategory. Thus, it has to be considered that human language consists of many argument and sentence structures that restrict such relations. That means the sentence structures also cover comparisons where both nouns come from

[1]The source code of our model is available at `https://github.com/cicl-iscl/SemEval3`.

Construction	Example
`andtoo`	I like teaspoons, and mugs too.
`butnot`	I like cats, but not frogs
`comparatives`	I like apples as much as cauliflower.
`drather`	I would rather have veal than salmon.
`except`	I like seafood, with the only exception of salmon.
`generally`	I like peaches, and more generally fruits.
`particular`	I like jewelry, and in particular necklaces.
`prefer`	I do not like tiramisu, I prefer broccoli.
`type`	I like parrots, not other types of birds.
`unlike`	Unlike glass, PVC is often mentioned in this text.

Table 1: Example sentences from subtask 1 with a binary label 1 (i.e., acceptable).

the same broader category. This has also been covered in the work by Clarke (2012), who refers to taxonomies as a framework represented in a hierarchy where lexical counterparts or synonyms are considered. Therefore, the given shared task comes with a challenge different from only recognizing the taxonomic relation—furthermore, the embedding construction allowing or disallowing the given relation had to be checked.

The participants were provided with two datasets to work with the individual subtasks. The training set for subtask 1 was composed of 5,837 sentences for each language with binary labels representing 1 as an acceptable sentence and 0 as an unacceptable sentence. This subtask covers the binary prediciton of acceptability labels of each sentence given in the test set with 14,560 samples. Subtask 2 consists of the prediction of an average score on a seven-point Likert scale for 1,009 sentences in the test set. The original scores in the training set were annotated by humans.

Figure 1 shows the scores assigned to the sentences presented in the training set in subtask 2. This figure only shows the scores averaged over all annotators since per-annotator information is not available.

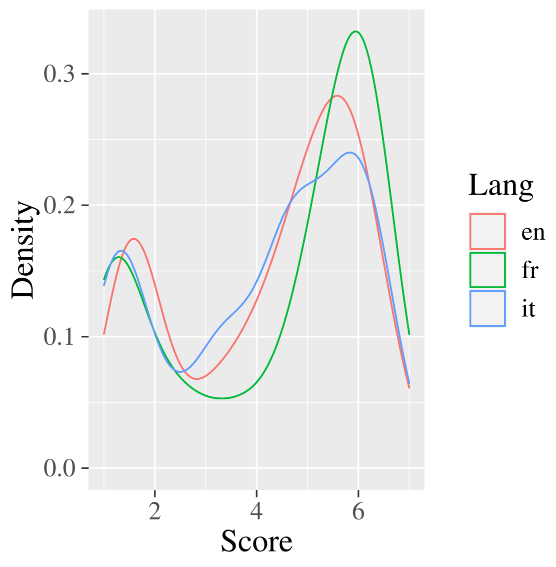


Figure 1: Distribution of the scores given to the sentences from the training set of subtask 2 by annotators.

3 System Overview

We used pre-trained BERT (Devlin et al., 2019) networks for subtasks 1 and 2. Separate models were used for each of the languages, specifically, the AlBERTo model (Polignano et al., 2019) was used for Italian, BERT base uncased was used for English, and the monolingual CamemBERT (Martin et al., 2020) model was used for French. The BERT models are powerful and highly versatile language models that possess the benefit of having learned good representations of the language they were trained on. This gives them a decisive edge over using models that are trained exclusively with the data provided for training as these pre-trained models will, for example, have encountered and learned representations for words that are not in the training set but are in the test set, whereas a model trained only on the training set will have trouble dealing with these unfamiliar words. As such they offer the opportunity for better generalization.

The bigger challenge of these tasks was not to produce models that perform well on the limited training data but to produce models that generalize well and do not merely overfit on the provided data. To this end, standard deep learning regularization techniques such as weight decay, dropout, and model averaging were used; nonetheless, the models performed much worse on the test data than on the validation data. Actually producing models that perform better at generalizing would likely have required data augmentation and/or alternative training routines.

The models were not shared between subtasks 1 and 2, meaning that while the same pre-trained models were used for subtasks 1 and 2, the model used for subtask 2 was not fine-tuned for subtask 1 and vice versa.

3.1 Subtask 1

For subtask 1, the sentences were tokenized using the tokenizers of the pre-trained BERT models. The BERT model was extended with one fully connected hidden layer and an output layer. The model was trained to perform the classification task using cross-entropy loss, backpropagated using the Adamw optimizer (Loshchilov and Hutter, 2017), which combines the Adam optimizer (Kingma and Ba, 2017) with weight decay regularization. Dropout was used for further regularization. Gradient clipping by-norm was applied to solve the exploding gradient problem. Thirty percent of the data was used as a validation set, learning was terminated through early stopping with the loss function as the stopping criterion.

3.2 Subtask 2

The model for subtask 2 is similar to the model for task 1, again extending the BERT models with a hidden layer and one output layer producing a single number for the regression task. The output and targets were normalized to be between zero and one. The inverse transformation was then applied to the model output to get the final output on the original scale. The model was trained on seventy percent of the training data using mean squared error as the loss function. Dropout was used and weight decay was applied through the AdamW optimizer. The training was terminated using early stopping with the remaining thirty percent of the training data used for validation. For this task, we trained ten models per language, each with its own training split of the data. The final prediction for the test set was the median prediction of these models. We chose to use the median and not the mean as it is less affected by outlier predictions.

3.3 Hyperparameters

Hyperparameters were determined using grid search over a limited selection of plausible candidate values, including learning rate (1×10^{-5} for all models), the number of fully connected layers, neurons per layer, and in the case of subtask 2 a multiclass approach was also tested. For subtask 1 choosing bigger models, in the end, 2 hidden layers with 512 neurons each were used, which led to improvements on the validation set but may have caused overfitting that negatively impacted performance on the test set. For subtask 2, bigger models did not perform better than small ones, and as a result, the final models contained only a single hidden layer with 16 neurons. The complete hyperparameters are listed in appendices A and C.

	total	**It**	**Fr**	**En**
Precision	0.75	0.73	0.77	0.74
Recall	0.80	0.73	0.89	0.80
F1	0.78	0.73	0.83	0.77

Table 2: Results for subtask 1. All scores are micro-averaged over different constructions. The total scores are also micro-averaged over the languages.

4 Results

Unless stated otherwise, we analyzed, evaluated, and visualized the results in R (R Core Team, 2020) with the help of the packages caret (Kuhn, 2021), dplyr (Wickham et al., 2021), ggplot2 (Wickham, 2016), plyr (Wickham, 2011), readr (Wickham and Hester, 2020), stringr (Wickham, 2019), tikzDevice (Sharpsteen and Bracken, 2020), tm (Feinerer et al., 2008), and xtable (Dahl et al., 2019).

4.1 Subtask 1

The test data contains 14,560 sentences per language. Table 2 shows the results of the evaluation for each language. Out of the 18 participating teams (and three additional teams who only submitted results for the English subset), the KaMiKla models ranked 15th in the overall competition and the Italian part and 12th and 16th in the French and English portion, respectively.

The highest overall score in the competition was 0.94, an F1 measure of 0.93 for Italian and French, and 0.97 for English. The trivial baseline that used n-grams as features reached a global score of 0.73. The F1 score for the Italian model is 0.68, 0.76 for French, and 0.73 for English.

After the evaluation phase ended, the test sets were published containing an additional label that denoted the syntactic construction used in the respective sentence. Table 1 illustrates examples of what the labels mean. Further analysis revealed that the type of construction had an enormous effect on the performance of the KaMiKla models. Figure 2 gives an overview of this. For the detailed evaluation metrics for each language by construction, see Appendix B.

The KaMiKla models performed with an F1 score in a range between 0.86 and 0.97 across all languages for `butnot`, `comparatives`, `drather`, `prefer`, and `andtoo` constructions which made up approximately 42.3% of the test dataset. These scores are about what we expected from performance on the validation set.

`except`, `particular`, and `unlike` constructions show stark differences in performance between languages. In all three cases, the French model achieves much better results than the English and Italian ones, which will be discussed in more detail later. Furthermore, the models performed poorly on `type` constructions across all languages, only overshadowed by scores close to 0 for generalizations.

4.1.1 except

Sentences containing `except` constructions made up approximately 12.9% of the test data. Especially the Italian model seemed to have trouble classifying exceptions correctly. For example, it gives the sentence *Adoro le verdure, eccetto le carote.*[2] the label "0" even though it is a semantically flawless Italian sentence. The problem seems to be the recall rather than precision. While a score of 0.82 is not much lower than the precision of the previously mentioned constructions, the recall score of 0.31 shows that the model could not accurately predict the taxonomic relations in a sentence containing an exception. This observation extends, if less prominently, to English and French.

4.1.2 particular

`particular` sentences, which make up 13% of the test data, show a striking difference in performance between languages. The precision ranges from 0.39 for the English model to 0.92 for the French model.

4.1.3 unlike

Sentences containing `unlike` constructions (9% of the test data) were still classified correctly relatively often by the French model, despite a recall score of 0.63. The Italian and English models performed much worse due to low recall. Interestingly, the English model has an almost-perfect precision of 0.99, while the Italian model only reached 0.49, which shows the difference between the models again.

[2] *I love vegetables, except for carrots.*

	total	It	Fr	En
MSE	3.72	2.88	5.29	2.97
RMSE	1.93	1.70	2.30	1.72
RHO	0.19	0.19	-0.01	0.06

Table 3: Results for subtask 2. All scores are micro-averaged over different constructions. The total scores are also micro-averaged over the languages.

4.1.4 type

About 13% of the test sentences contained `type` constructions like *I like fruits, an interesting type of lemon.* There is not much to say about them other than that the models' performance on them was terrible. F1 measures range from 0.06 to 0.11 with very low recall (0.25 to 0.47) and even worse precision (close to 0).

4.1.5 generally

Possibly the most surprising result is the utter confidence with which the models misclassified generalizations. Across languages, all evaluation metrics are below 0.05 for sentences labeled `generally`.

4.2 Subtask 2

The top-performing models reached a Spearman correlation of 0.81, 0.84, and 0.76 in Italian, French, and English, respectively, yielding an overall score of 0.80. The KaMiKla models performed much worse and ranked in 11th place with a global rank of 0.08. Table 3 shows an overview of the evaluation of the second subtask. Surprisingly, the n-gram-based regression model serving as a baseline ranked in 4th place with correlations of 0.34, 0.32, 0.27 in Italian, French, and English, respectively, outperforming many submissions, including the one present in this paper.

Similar to the first subtask, we analyzed these results grouped by the used construction. Appendix D contains the detailed analysis. The metrics considered are the mean squared error (MSE), the root mean squared error (RMSE), and Spearman correlation (RHO). The constructions of the second subtask are a subset of the ones used for the first one, with the `generally` label changed to `ingeneral`. Figure 3 visualizes the root mean squared errors of the second subtask grouped by construction.

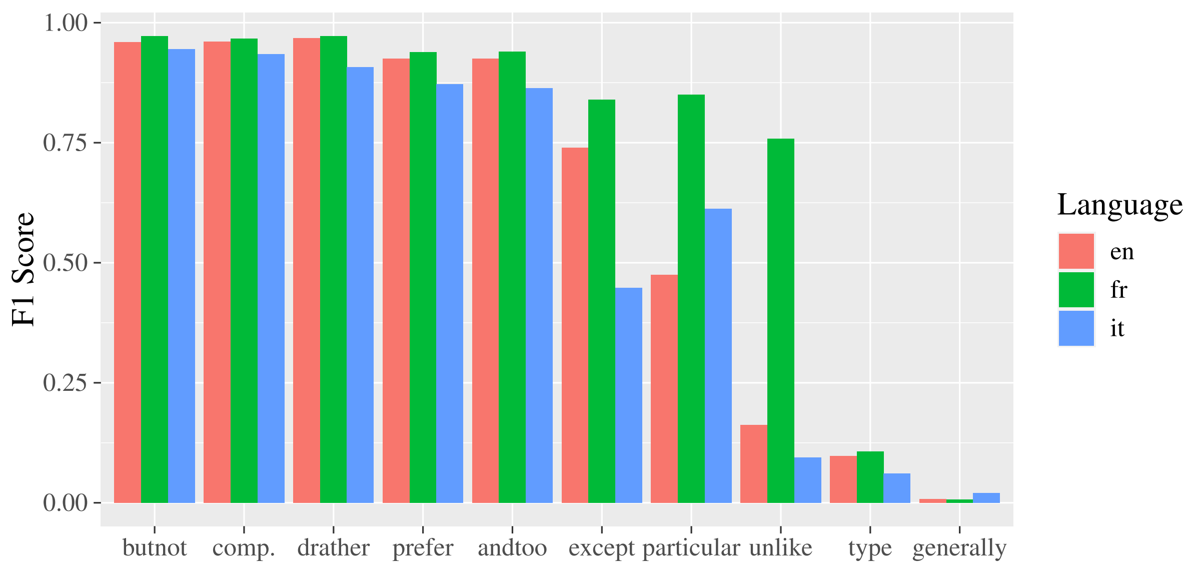


Figure 2: An overview over the F1 scores the KaMiKla models reached in subtask 1, ordered by score.

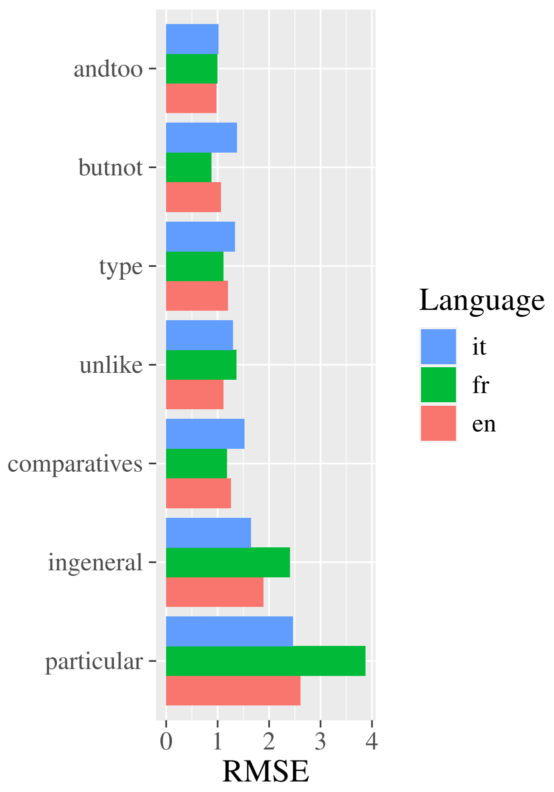


Figure 3: An overview over the root mean squared error scores the KaMiKla models reached in subtask 2, ordered by score.

The difference in performance between constructions is evident here as well. Interestingly, the French regression model seemed to have more trouble than its Italian and English counterparts, while the French classification model outperformed the other languages.

Another interesting comparison is that of the constructions. While the classification models failed on `type` sentences, those are some of the most successful sentences in subtask 2; this indicates that the bad scores on some constructions are due to fine-tuning and not some inherent difficulty the BERT models have with understanding them. On the other hand, this might be a side effect of the models not performing very well in general.

5 Discussion

Our models performed considerably worse on the test dataset than on the validation set, and these differences vary across languages and constructions. While we can't determine for sure where this significant drop in performance comes from, there are some theories worth investigating.

Of course, one challenge (especially of the second part) of the task is the scarcity of data. Subtask 1 contains 5,837 sentences in the training data per language; and 14,560 sentences in the test set. For subtask 2, there are 524 training sentences and 1,009 test instances in each language. Possibly, the regression models performed unsatisfyingly simply because there wasn't enough training data available.

We also wanted to investigate the inherent differences between train and test data. Figure 4 compares the sentence lengths of the train and test set, grouped by language. There are huge differences between the languages and, perhaps more importantly, between the train and test sets of the three languages. The French training data contained sentences of similar length to those in the test data (mean of 8.94 for the test and 9.28 for the training set), and those sentences were generally longer than those of the other two languages. The data from the Italian test set seems to contain sentences of more varied lengths than the training set.

The data of the second subtask show a similar discrepancy between training and test sentences. The sentences from the test data are longer on average than those from the training data in all languages. The lengths also seem to be more variable in the test set, possibly due to the higher number of sentences. The gap between the input sentences could have led to finetuning not remarkably improving the performance of the models.

We furthermore looked at the distribution of different types of constructions in more detail. There were no construction labels for the training data, so we trained a simple Naive Bayes classifier on the tf-idf-transformed test sentences in Python using pandas (pandas development team, 2022) and scikit-learn (Pedregosa et al., 2011). Because of the simple structure of the input sentences and a cross-validation score of 100%, we will assume the construction labels to be accurate in further discussion.

Figures 5 and 6 compare the distributions of the construction labels in train and test sets of subtask 1 and 2, respectively. There are once again huge differences between training and test set. Notably, the training data does not contain a single `unlike` sentence. Despite that, the models did not necessarily perform worse on this type of construction. In total, it does not seem like the distribution of construction types in the training data influenced model performance much at all. There are instances of models performing inadequately on frequent constructions in the training data, like `type` constructions in the first subtask. However, `drather` constructions were often classified correctly despite the scarcity of training sentences.

6 Conclusion

In this work, we discussed an approach to modeling taxonomic relationships using pre-trained language models, namely AlBERTo (Polignano et al., 2019), BERT (Devlin et al., 2019), and CamemBERT (Martin et al., 2020) in the context of the SemEval Task 3 of the year 2022. The KaMiKla group participated in both the classification and the regression subtask. While the performance of the models was overall unsatisfying, further analysis revealed that the type of taxonomic relation that the words in a given sentence severely affected how well the models did. While the reason for this remains unclear, it might be interesting to tailor the finetuning of the BERT-based model to specific constructions or combine it with a classifier that classified the input sentences according to the type of taxonomic relation.

Acknowledgements

We want to thank Çağri Çöltekin for his technical and topical assistance throughout this project.

References

Michael Clarke. 2012. 4 - the digital revolution. In Robert Campbell, Ed Pentz, and Ian Borthwick, editors, *Academic and Professional Publishing*, pages 79–98. Chandos Publishing.

David B. Dahl, David Scott, Charles Roosen, Arni Magnusson, and Jonathan Swinton. 2019. *xtable: Export Tables to LaTeX or HTML*. R package version 1.8-4.

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

Ingo Feinerer, Kurt Hornik, and David Meyer. 2008. Text mining infrastructure in r. *Journal of Statistical Software*, 25(5):1–54.

Paul Kay. 1971. Taxonomy and semantic contrast. *Language*, 47:866–887.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Max Kuhn. 2021. *caret: Classification and Regression Training*. R package version 6.0-90.

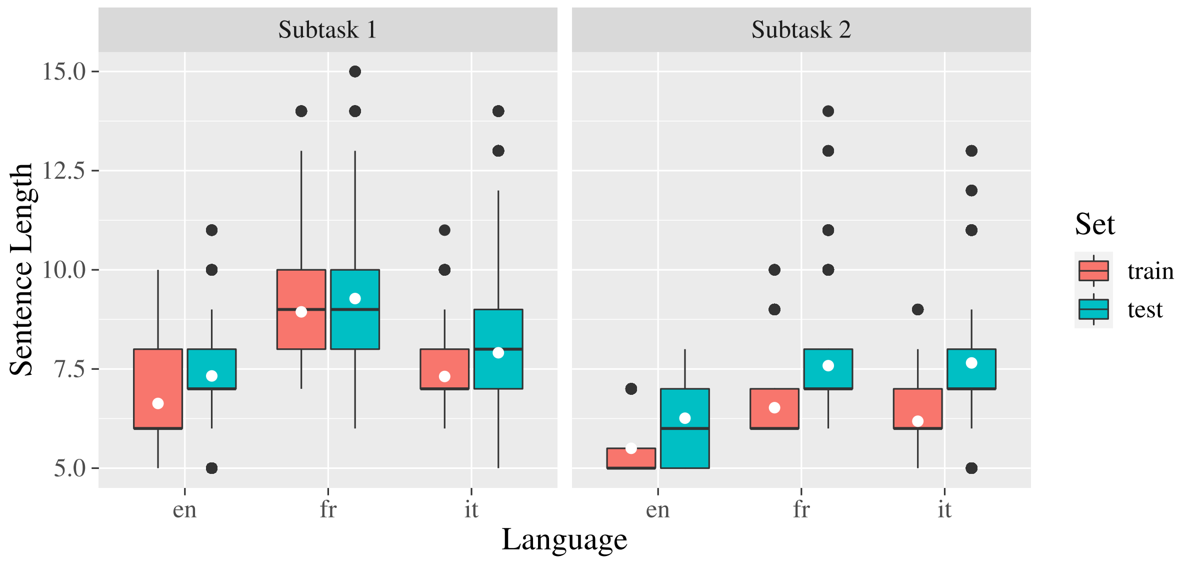


Figure 4: A comparison of the sentence lengths of train and tests sets in the different languages for both subtasks. The mean is visible in white. The difference seems to be less pronounced in the French data of subtask 1, which might have contributed to the better scores. In subtask 2, the sentences seem to be generally longer in the test data than the training data in all languages.

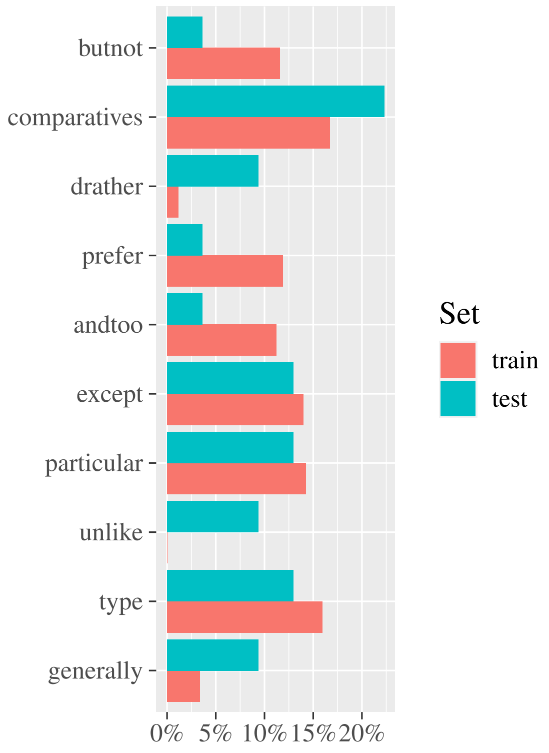


Figure 5: A comparison of the distribution of different constructions in train and test set of subtask 1, ordered by their score in the classification task. The x-axis shows the percentage of this label in one of the datasets.

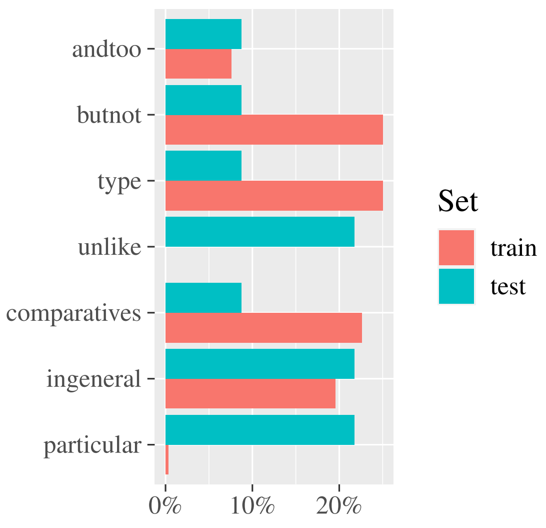


Figure 6: A comparison of the distribution of different constructions in train and test set of subtask 2, ordered by their score in the regression task. The x-axis shows the percentage of this label in one of the datasets.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*. Published as a conference paper at ICLR 2019.

Anh Tuan Luu, Yi Tay, Siu Cheung Hui, and See Kiong Ng. 2016. Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 403–413, Austin, Texas. Association for Computational Linguistics.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

George A. Miller. 1992. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Roberto Navigli and Simone Paolo Ponzetto. 2010. BabelNet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216–225, Uppsala, Sweden. Association for Computational Linguistics.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical embeddings for hypernymy detection and directionality. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 233–243, Copenhagen, Denmark. Association for Computational Linguistics.

The pandas development team. 2022. pandas-dev/pandas: Pandas 1.4.1.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. ALBERTO: Italian BERT language understanding model for NLP challenging tasks based on tweets. pages 1–6.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Charlie Sharpsteen and Cameron Bracken. 2020. *tikzDevice: R Graphics Output in LaTeX Format*. R package version 0.12.3.1.

Hadley Wickham. 2011. The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, 40(1):1–29.

Hadley Wickham. 2016. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.

Hadley Wickham. 2019. *stringr: Simple, Consistent Wrappers for Common String Operations*. R package version 1.4.0.

Hadley Wickham, Romain François, Lionel Henry, and Kirill Müller. 2021. *dplyr: A Grammar of Data Manipulation*. R package version 1.0.6.

Hadley Wickham and Jim Hester. 2020. *readr: Read Rectangular Text Data*. R package version 1.4.0.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. SemEval-2022 Task3 (PreTENS): Evaluating neural networks on presuppositional semantic knowledge. In *Proceeding of SEMEVAL 2022*.

A Hyperparameters For Subtask 1

Table 4 shows the hyperparameters used in subtask 1 for the different languages.

B Results For Subtask 1

Tables 5, 6, and 7 show the performance metrics of the BERT-based models in the classification task. There are considerable differences between constructions as well as between languages.

C Hyperparameters For Subtask 2

Table 8 shows the hyperparameters used in subtask 2 for the different languages.

D Results For Subtask 2

Tables 9, 10, and 11 show the metrics for the regression task.

	Batch Size	**Learning Rate**	**Max. Len. Sent.**	**Patience**	**Hidden Layers**	
					Number	**Size**
English	32	1×10^{-5}	15	5	2	512
French	32	1×10^{-5}	20	5	2	512
Italian	32	1×10^{-5}	15	5	2	512

Table 4: Hyperparameters used in subtask 1.

	total	***andtoo***	***butnot***	***comp.***	***drather***	***except***	***generally***	***particular***	***prefer***	***type***	***unlike***
Precision	0.74	0.86	0.93	0.93	0.94	0.85	0.01	0.39	0.88	0.06	0.99
Recall	0.80	1.00	0.99	0.99	1.00	0.65	0.01	0.62	0.98	0.40	0.09
F1	0.77	0.93	0.96	0.96	0.97	0.74	0.01	0.47	0.93	0.10	0.16

Table 5: Metrics for the English sentences in subtask 1, grouped by construction.

	total	***andtoo***	***butnot***	***comp.***	***drather***	***except***	***generally***	***particular***	***prefer***	***type***	***unlike***
Precision	0.77	0.90	0.95	0.94	0.95	0.94	0.01	0.92	0.90	0.06	0.96
Recall	0.89	0.98	1.00	0.99	1.00	0.76	0.01	0.79	0.98	0.47	0.63
F1	0.83	0.94	0.97	0.97	0.97	0.84	0.01	0.85	0.94	0.11	0.76

Table 6: Metrics for the French sentences in subtask 1, grouped by construction.

	total	***andtoo***	***butnot***	***comp.***	***drather***	***except***	***generally***	***particular***	***prefer***	***type***	***unlike***
Precision	0.73	0.83	0.92	0.92	0.94	0.82	0.03	0.74	0.85	0.03	0.49
Recall	0.73	0.90	0.97	0.95	0.87	0.31	0.02	0.52	0.89	0.25	0.05
F1	0.73	0.86	0.95	0.93	0.91	0.45	0.02	0.61	0.87	0.06	0.09

Table 7: Metrics for the Italian sentences in subtask 1, grouped by construction.

	Batch Size	**Learning Rate**	**Max. Len. Sent.**	**Patience**	**Hidden Layers**	
					Number	**Size**
English	32	1×10^{-5}	15	5	1	16
French	32	1×10^{-5}	20	5	1	16
Italian	32	1×10^{-5}	15	5	1	16

Table 8: Hyperparameters used in subtask 2.

	total	***andtoo***	***butnot***	***comparatives***	***ingeneral***	***particular***	***type***	***unlike***
MSE	2.97	0.96	1.13	1.59	3.57	6.83	1.45	1.24
RMSE	1.72	0.98	1.06	1.26	1.89	2.61	1.20	1.11
RHO	0.06	-0.32	0.04	0.07	-0.28	-0.25	0.26	0.18

Table 9: Metrics for the English sentences in subtask 2, grouped by construction.

	total	***andtoo***	***butnot***	***comparatives***	***ingeneral***	***particular***	***type***	***unlike***
MSE	5.29	1.00	0.77	1.39	5.77	14.99	1.24	1.87
RMSE	2.30	1.00	0.88	1.18	2.40	3.87	1.11	1.37
RHO	-0.01	0.43	0.19	0.35	-0.03	-0.29	0.25	0.38

Table 10: Metrics for the French sentences in subtask 2, grouped by construction.

	total	***andtoo***	***butnot***	***comparatives***	***ingeneral***	***particular***	***type***	***unlike***
MSE	2.88	1.03	1.90	2.32	2.71	6.05	1.79	1.69
RMSE	1.70	1.02	1.38	1.52	1.65	2.46	1.34	1.30
RHO	0.19	-0.21	-0.05	-0.28	-0.07	-0.00	-0.18	0.34

Table 11: Metrics for the French sentences in subtask 2, grouped by construction.

HW-TSC at SemEval-2022 Task 3: A Unified Approach Fine-tuned on Multilingual Pretrained Model for PreTENS

Yinglu Li, Min Zhang, Xiaosong Qiao, Minghan Wang
Hao Yang, Shimin Tao, Ying Qin
Huawei Translation Services Center, Beijing, China
{liyinglu, zhangmin186, qiaoxiaosong, wangminghan,
yanghao30, taoshimin, qinying}@huawei.com

Abstract

In the paper, we describe a unified system for task 3 of SemEval-2022. The task aims to recognize the semantic structures of sentences by providing two nominal arguments and to evaluate the degree of taxonomic relations. We utilise the strategy that adding language prefix tag in the training set, which is effective for the model. We split the training set to avoid the translation information to be learnt by the model. For the task, we propose a unified model fine-tuned on the multilingual pretrained model, XLM-RoBERTa. The model performs well in subtask 1 (the binary classification subtask). In order to verify whether our model could also perform better in subtask 2 (the regression subtask), the ranking score is transformed into classification labels by an up-sampling strategy. With the ensemble strategy, the performance of our model can also be improved. As a result, the model obtained the second place for subtask 1 and subtask 2 in the competition evaluation.

1 Introduction

As we all know, the proposal of BERT(Devlin et al., 2018; Vaswani et al., 2017), which is based on masked language modeling, is a huge milestone in the history of natural language understanding (Peters et al., 2018; Schuster et al., 2019). Compared with various language representation models, BERT successfully pushes the GLUE score at 7.7 points absolute improvement(Devlin et al., 2018). Soon, different types of language models such as XLNet(You et al., 2019), RoBERTa(Liu et al., 2019),mBert (Devlin et al., 2018; Radford and Narasimhan, 2018) and XLM (Lample and Conneau, 2019) are also proposed. Compared with some strong monolingual models introduced above, XLM-RoBERTa is more competitive on the GLUE and XNLI benchmarks(Conneau et al., 2019). There are lots of pretrained models proposed in recent years, which are capable of learning the implicit knowledge. Some works have proved that the neural language model has learnt the implicit linguistic knowledge and this knowledge can significantly affect the predictions through fine-tuning(Miaschi et al., 2020; Puccetti et al., 2021). It has been believed that the pretrained models trained on Wikipedia and other datasets already have some implicit knowledge. Implicit knowledge could be classified into two categories: the connection between two objects, and the implicit logic and syntax behind the sentence. It means, we need to build a model which is capable of verifying the rationality and reliability of sentences, testing whether the latent knowledge can be expressed explicitly.

Presupposed taxonomy is a kind of concept in computational linguistics. Two arguments could have several different taxonomy relationships. For example, the sentence "I like piano, but not the instrument" is a classic pattern which contains two arguments. In the transition sentence, "piano" and "instrument" have a taxonomic relation, so the conclusion could be drawn that the sentence is implausible and unacceptable. Similarly, there are a set of sentences that could have such contradiction in the competition. The goal of the SemEval-2022 Task 3 (Zamparelli et al., 2022) is to recognize the presupposed taxonomy relation between two nouns, which could be a complex linguistic problem. In order to obtain the skill, the model needs to understand the implicit meaning of the sentences as well as whether the presupposed taxonomy relation exists in one pair of entities.

ID	Sentence	Labels
5155	*I like **ham**, but not **fish**.*	1
2560	*I like **restaurants**, and **clerks** too.*	1
3711	*I like **jewlry** more than **skirts**.*	1
4481	*I like **scientists** more than **geneticists**.*	0
3104	*I do not like **seafood**, I prefer **salmon**.*	0

Table 1: Dataset Samples for subtask 1

2 Task Description

Along the lines of ideas above, there are several representative pretained-models in the competition. To be specific, we tried commonly used pretrained models including Bert, Roberta-large, and multilingual language models like XLMR. At the same time, we suspect that the former model can be generalized to subtask 2. The final ranking result supports our hypothesis strongly.

Subtask 1 is a binary classification task aiming at predicting the acceptability of sentences (A(1) VS UA(0)). There are two parameters that have the presupposed taxonomy relation, followed by the label "0"(the semantic relation is not acceptable) or "1" (the semantic relation is acceptable). Note that label "0" stands for the contradictory sentences, and label "1" stands for the plausible sentence. Some samples are shown in table 1, in which most of them obey a similar pattern: "I like A, but not B" or "I do not like A, I prefer B...", etc. The bold portion shows the key entities in the sentence. It is obvious that the taxonomies relation between two entities stands for some implicit information, which should be learnt by our model.

In this multilingual task, three languages (French, Italian and English) are included. Table 1 only presents the English samples and three datasets express completely consistent arguments. The same id indicates the same meaning in a different language.

On the contrary, subtask 2 is a regression task aiming at predicting the degree of Acceptance in a seven Likert-scale. The only difference between the dataset of subtask 1 and subtask 2 is the label. In subtask 2, the label(score) is a float number between 1 and 7, which indicates the acceptable degree of the sentence. Some English data samples are shown in table 2. Note that the sentence with higher score is more reasonable than the sentence with lower one. For instance, the sentence "I like seafood, but not crabs" sounds like a correct sentence in daily life. But the sentence "I like beef, an interesting type of caviar" is a contradictory sentence without any doubt, because the beef is not a caviar.

In addition, Table 3 provides the size of the train and test set. Obviously, the size of dataset for subtask 2 is much smaller than that for subtask 1. So it becomes important to expand the size of dataset of subtask 2. There are two main subtasks in PreTENS (Presupposed Taxonomies: Evaluating Neural Network Semantics).

ID	Sentence	Scores
261	*I like **beef**, an interesting type of **caviar**.*	1.09
440	*I like **trees** more than **grass**.*	5.64
207	*I like **shrubs**, an interesting type of **fir**.*	2.67
60	*I like **trees** but not **birches**.*	1.83
436	*I like **oaks** more than **grass**.*	5.83
104	*I like **seafood**, but not **crabs**.*	6.42

Table 2: Dataset Samples for subtask 2

Task	Type	Language	Train size	Test size
Subtask 1	Classification	En	5840	14560
		Fr	5840	14560
		It	5840	14560
Subtask 2	Regression	En	526	1009
		Fr	526	1009
		It	526	1009

Table 3: Task Dataset Description

3 System

3.1 Data Process for subtask 1

Based on a suitable single model and adaptive finetuning models which have learnt enough implicit information, it becomes possible to express the explicit knowledge(taxonomies) for a model.

The baseline model provided by competition reaches accuracy at 0.8, which is an amazing result. We also find the training set is extremely unbalanced in the proportion of positive and negative samples for some patterns, as we can see in Table 9. So we tried to adjust the proportion of those unbalanced patterns. However, the accuracy decreased a lot after the adjustment. This result shows that the model is actually learning the proportion of labels in the dataset instead of the implicit information. In order to solve the problem in the competition, we need to choose some models which are capable of learning implicit knowledge.

3.1.1 Language Tags

Considering the three languages in our competition, we apply the prefix tag to indicate which language the sentence is in. The XLM-RoBERTa is a multilingual model so the input data consists of mixed sentences in three languages.

As we can see from the figure, angle brackets and language abbreviation is used as the uniform prefixes. Specifically, <fr> stands for French, <en> stands for English and <it> stands for Italian. The

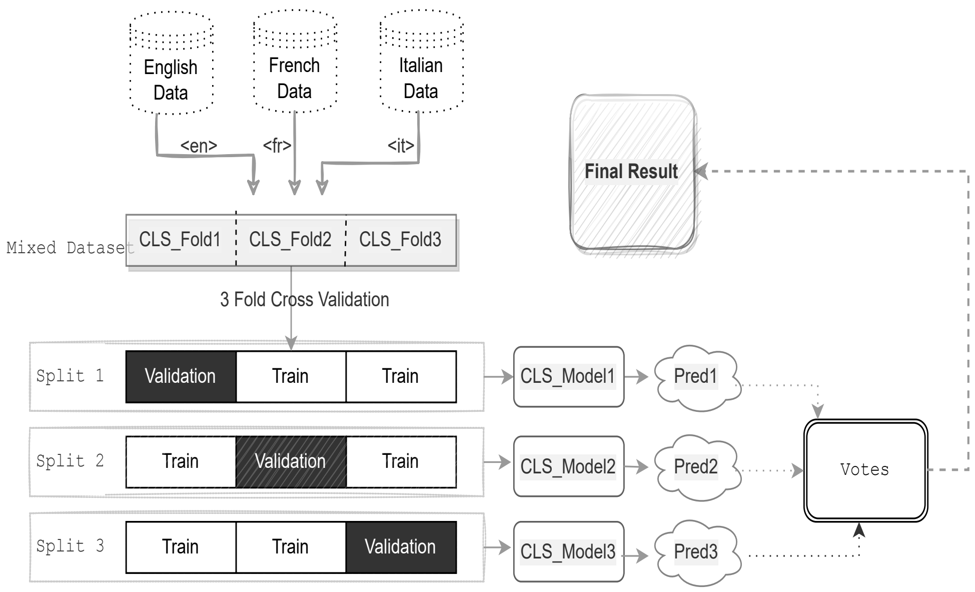


Figure 1: Ensemble classification models

Raw Sentence	**Language Tag**	**New Sentence**
I like jewelry more than skirts	en	***<en>**I like jewelry more than skirts.*
J' aime les bijoux plus que les jupes .	fr	***<fr>**J' aime les bijoux plus que les jupes .*
Amo i gioielli più delle gonne .	it	***<it>**Amo i gioielli più delle gonne .*

Table 4: Adding Language Tags for sentences with the same id

strategy helps to provide some linguistic information to the model artificially. Obviously, there are only linguistic differences between sentences with the same id, and their essential meanings are consistent. Imagining the situation without artificially added labels, there might be some difficulties in identifying sentence pairs with the same id but in different languages.

3.1.2 Dataset Split

We also apply the three-fold cross-validation in the competition. After further fine-tuning, the model has achieved good results on subtask 1. In the task description chapter, we mentioned that the same meaning is expressed in different languages in the dataset, which can be told by ids. In other words, sentences with same meanings in different languages share the common id.

We find that sentences with the same meaning might appear in the training set and the test set respectively in a multilingual language model. So, the model might learn the meaning of translation which should be avoided in our tasks. Therefore, in this section, we divide the data and put sentences with the same ID into the same set to prevent the model from learning the translation information. With such treatment, it can be ensured that the model learns implicit knowledge instead of cheating with translation.

3.2 A Unified Model for Subtask 2

The Situation becomes difficult in subtask 2. Considering the high similarity between datasets used by subtask 2 and subtask 1, and the size of the latter is much smaller than the former, it is important to use the data augmentation. In order to get good performance with the model in subtask 1, we make some efforts to transform the data provided in subtask 2 by using the method used above.

In other words, our method can be easily transferred from subtask 1 to subtask 2, which not only reduces the workload and training time but also makes the prediction more reliable because of the extension of datasets.

From table 1 to 3 which describes the sample and size of datasets, the label in subtask 2 are real values from 1 to 7. The label is a score used to assess the reasonableness of the sentence. Com-

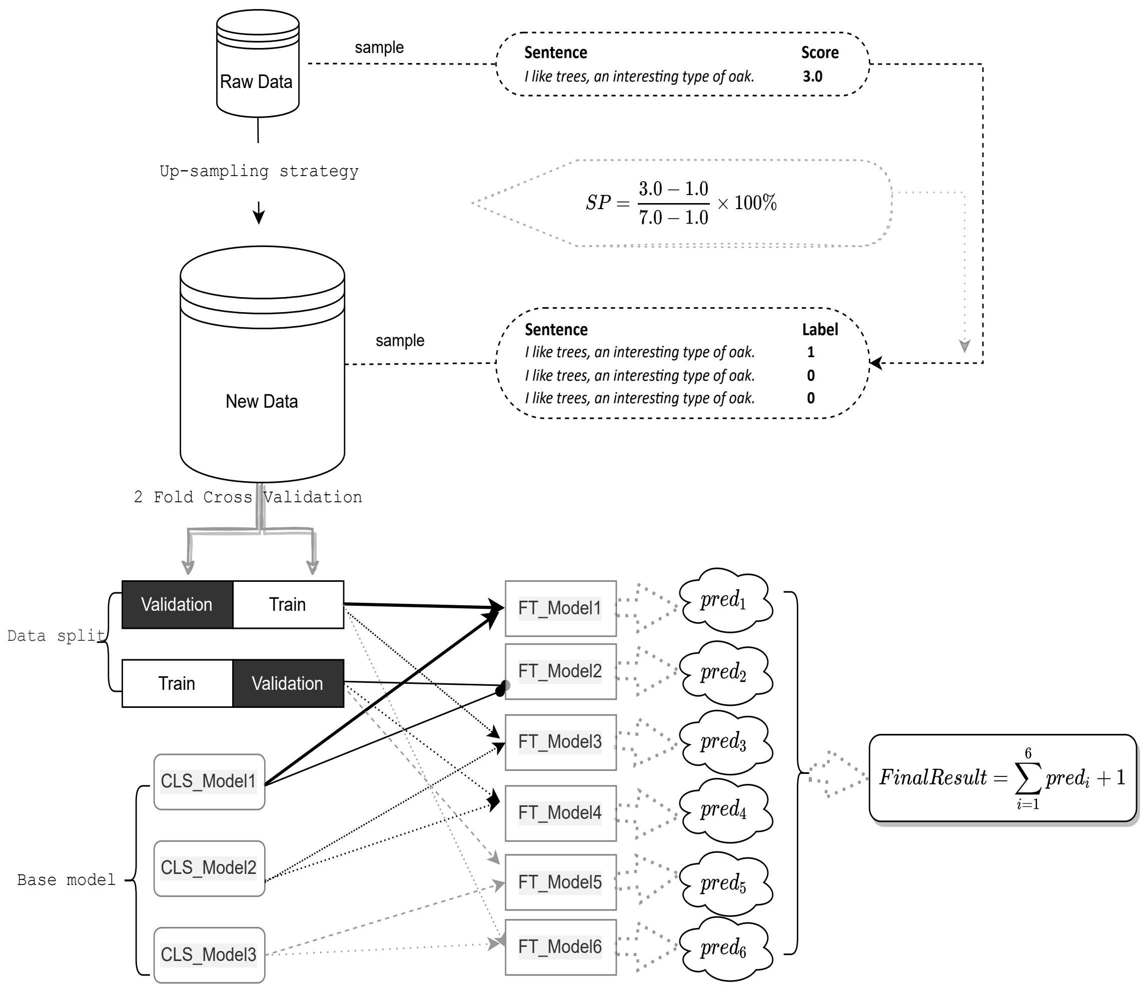


Figure 2: Ensemble regression models

paring tags in two subtasks, it is easy to imagine if sampling with the principle of score as probability, the meaning of the original score could be involved in the new label.

For each sentence from raw dataset, the score would be transformed to label according to the sampling possibility.

Sampling Probability(sp) is defined in the Eq 1, which depends on the ub(upper-bound of score) and lb(lower-bound of score).

$$sp = \frac{score - 1.0}{ub - lb} \times 100\% \qquad (1)$$

The up-sampling strategy is described in figure 2, which uses the expansion coefficient equalling 3 as an example. As for the sentence "I like trees, an interesting type of oak.", the sampling probability could be calculated according to the formula 1, which equals 1/3. Consequently, the raw sentence is duplicated to three same sentences with the label (1, 0, 0), which comes from the sampling strategy.

In data processing, we choose 10 as the expansion coefficient while converting the original score into sampling probability. 10 times the size of datasets, an expansion dataset is obtained in this way.

We use the new expansion dataset to train the model from subtask 1. At the same time, since there are three models obtained from subtask 1, and we split the new dataset into two copies, obviously $3 \times 2 = 6$ models have been used for the ensemble.

Considering the lack of regression data, we make some attempts to up-sampling the regression data. More specifically, the difference between classification data and regression data is the label. The former could be an integer while the latter could be a float between 1 and 7.

Apart from the label, the sentence have the similar pattern and entity. So it might be effective to reuse the model in subtask 1. Each original sentence is duplicated to ten same sentences with the label which might be 0 or 1 depending on the possi-

Model	**Train Set**	**Dev Set**	**Dev Acc**	**Global Rank**
CLS_Model1	CLS_fold1, 2	CLS_fold3	0.9426	88.4185
CLS_Model2	CLS_fold2, 3	CLS_fold1	0.9350	-
CLS_Model3	CLS_fold1, 3	CLS_fold2	0.9140	-
Ensemble	-	-	-	92.7968

Table 5: Ensemble Strategy for subtask 1

Model	**EN Macro**	**EN F1**	**FR F1 Macro**	**FR F1**	**IT F1 Macro**	**IT F1**
CLS_Model1	89.0199	89.0200	89.1187	89.1190	87.1169	87.1170
Ensemble	93.0410	92.5830	93.0116	92.5470	92.3388	91.8020

Table 6: Detailed Results of ensemble models for subtask 1

bility. Obviously, the possibility and the regression score is linearly and positively correlated. After obtaining an expanded dataset which has 10 times the size of original dataset of subtask 2, our model could learn some new knowledge in training.

3.3 Ensemble

Through the step described in section 3.1.2, we obtained six models for subtask 1. According to the requirements described in the competition, subtask 2 would be evaluated by the Spearman correlation coefficient, which is a metric used to express distribution trends. In other words, when a distribution is appropriately scaled, the magnitude of the Spearman correlation coefficient would remain the same. This inspires us to directly superimpose the results of the six models for ensemble. To keep the final result size between 1 and 7, we shift the score of the result by 1.

For subtask 1, the dataset is split into three folds for cross-validation and then three models are obtained. Since the target of subtask 1 is to figure out whether the sentence is plausible or not, we use the voting strategy for ensemble. Specifically, three models could have three decisions about the label of one sentence. Naturally, the voting ensemble strategy could be applied in the stage. The final result would be the majority decision.

For subtask 2, the situation is different. It has been cleared that there are three models in the classification task. Based on each model, we use the strategy of up-sampling to expand our dataset and transform it into the classification one.

4 Results and Analysis

Table 5 and Table 6 illustrate the ensemble strategy and ensemble results for subtask 1. The accuracy is the metric to evaluate the result in subtask 1. The results of subtask 2 are illustrated in table 7 and table 8, and the metric is Spearman. Because of the relative measurement of the metric, there are some small shifts in our predictions actually bring no influence to the value of Spearman. Those results indicate that our model has an outstanding improvement compared to baselines.

Except for the experiments introduced above, there were still a lot of directions we tested, but not all of them were useful.

1) From the perspective of the pattern, we found the positive and negative patterns are unbalanced in the dataset. From the Table 9, there are some examples that appear in the dataset. The number of positive samples and negative samples is very unbalanced. So, we balanced the dataset and trained the model. However, this strategy was useless for our model, and it even decreased the accuracy of the model. We suspect that the test set is also an unbalanced dataset.

2) We tried to extract the entities of each sample, and concatenated the embeddings of entities and embeddings of "<CLS>". Eq 2 shows the change between origin embedding and the new embedding.

$$[EA, EB] -> [CLS, EA, EB] \qquad (2)$$

Then, we classified it with a binary classifier (Softmax). But there were no improvements in the performance.

Model	Base Model	Train Set	Dev Set	Dev Spearman
FT_Model1	CLS_Model1	Reg_fold1	Reg_fold2	0.6871
FT_Model2	CLS_Model1	Reg_fold2	Reg_fold1	0.7429
FT_Model3	CLS_Model2	Reg_fold1	Reg_fold2	0.7038
FT_Model4	CLS_Model2	Reg_fold2	Reg_fold1	0.7382
FT_Model5	CLS_Model3	Reg_fold1	Reg_fold2	0.7242
FT_Model6	CLS_Model3	Reg_fold2	Reg_fold1	0.6993
Ensemble	-	-	-	0.7582

Table 7: Ensemble Strategy for subtask 2

Model	Global Rank	RHO(IT)	RHO(FR)	RHO(EN)
FT_Model1	0.6871	0.7376	0.7986	0.6917
Ensemble	0.7572	0.7591	0.8050	0.7060

Table 8: Detailed Results of Ensemble Fine-tuned Model for subtask 2

3) We tried to extract all entities and capture all the related contents on Wikipedia and trained our language model based on the dataset. But it only brought limited improvements. Because Wikipedia is a very common dataset, the language model might be trained on the dataset before.

Pattern	Label 0/1
He trusts a , except his b.	0 / 12
He does not trust a , he prefers his b.	12 / 0
He likes a , an interesting type of b .	0 / 9

Table 9: Some patterns with unbalanced ratio of positive samples with negative samples.

5 Conclusion

In the experiment, we propose a solution for two subtasks of SemEval2022 Task 3. We illustrate the importance of data preprocessing. The data split and the use of the language tags both have a positive influence on the model performance. Considering the similarity of the dataset and the good performance of the model in subtask 1, we also explore the feasibility of a unified model. The unified model has great performance on both subtask 1 and subtask 2. In the future, there are some other directions that could be tried in the following explorations. We find that the model trained on a large dataset performs well, so, exploring large models might be an interesting direction.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints*, page arXiv:1810.04805.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual Language Model Pretraining. *arXiv e-prints*, page arXiv:1901.07291.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*, page arXiv:1907.11692.

Alessio Miaschi, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2020. Linguistic profiling of a neural language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 745–756, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv e-prints*, page arXiv:1802.05365.

Giovanni Puccetti, Alessio Miaschi, and Felice Dell'Orletta. 2021. How do BERT embeddings organize linguistic knowledge? In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep*

Learning Architectures, pages 48–57, Online. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. *CoRR*, abs/1902.09492.

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

Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. 2019. Reducing BERT pre-training time from 3 days to 76 minutes. *CoRR*, abs/1904.00962.

Roberto Zamparelli, Shammur A. Chowdhury, Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. Semeval-2022 task3 (pretens): Evaluating neural networks on presuppositional semantic knowledge. In *Proceeding of SEMEVAL 2022*.

SemEval-2022 Task 4: Patronizing and Condescending Language Detection

Carla Perez-Almendros **Luis Espinosa-Anke** **Steven Schockaert**
School of Computer Science & Informatics
Cardiff University
{perezalmendrosc,espinosa-ankel,schockaerts1}@cardiff.ac.uk

Abstract

This paper presents an overview of Task 4 at SemEval-2022, which was focused on detecting Patronizing and Condescending Language (PCL) towards vulnerable communities. Two sub-tasks were considered: a binary classification task, where participants needed to classify a given paragraph as containing PCL or not, and a multi-label classification task, where participants needed to identify which types of PCL are present (if any). The task attracted 77 teams. We provide an overview of how the task was organized, discuss the techniques that were employed by the different participants, and summarize the main resulting insights about PCL detection and categorization.

1 Introduction

The study of unfair, misleading or offensive language has attracted the interest of many scholars from the NLP research community. Most relevant tasks in this context focus on explicit, aggressive and flagrant phenomena, such as fake news detection or fact-checking (Conroy et al., 2015; Nakov et al., 2018; Atanasova et al., 2019; Barrón-Cedeno et al., 2020); detecting propaganda techniques (Da San Martino et al., 2020); modeling offensive language (Zampieri et al., 2019, 2020) identifying hate speech (Basile et al., 2019); and rumour propagation (Derczynski et al., 2017). However, there also exist subtler but equally harmful types of language, which have received less attention by the NLP community, and which, due to their subtle nature, we can expect to be more difficult to detect. This is the case, among others, for Patronizing and Condescending Language (PCL), which was the focus of Task 4 at SemEval-2022.

An entity engages in patronizing or condescending communication when its use of language reveals a superior attitude towards others. These attitudes, when normalized, routinize discrimination and make it less visible (Ng, 2007). Furthermore, the use of PCL is often unconscious and well-intended, especially when referring to vulnerable communities (Wilson and Gutierrez, 1985; Merskin, 2011). This good will can make PCL especially harmful, as the audience receives this discriminatory language with low defense and is often unaware of its effects.

Research in sociolinguistics presents PCL as a subtle, often unconscious but harmful and discriminative kind of language (Mendelsohn et al., 2020). It creates and feeds stereotypes (Fiske, 1993), which result in greater exclusion, rumour spreading and misinformation (Nolan and Mikami, 2013). PCL also tends to strengthen power-knowledge relationships (Foucault, 1980), calling for charitable action instead of cooperation and presenting those who can help as *saviours* of those in a less privileged position (Bell, 2013; Straubhaar, 2015). Furthermore, PCL tends to conceal who is responsible for very deep-rooted societal problems, sometimes by implicitly or explicitly blaming the underprivileged communities or individuals for their situation, and often involves ephemeral and simple solutions. (Chouliaraki, 2010). The use of PCL by privileged communities has also been related to the so-called pornography of poverty (Nathanson, 2013), a communication style that depicts vulnerable situations with a pity discourse to move a target audience to charitable action and/or compassionate attitudes.

While the negative impact of PCL, both in social interactions and in corporate and political discourse, has been extensively studied in the social sciences, it still remains an under-explored phenomenon in NLP. Nonetheless, we believe that PCL detection offers a number of important challenges for NLP research, which warrant more work in this area, especially given the societal benefits that would result. For instance, given its subtle and subjective nature, we can expect PCL detection to be harder than tasks that are focused on more flagrant phenomena. Moreover, PCL detection often involves the need for an implied *understanding* of human values and ethics, which requires a form of commonsense reasoning that NLP models are likely to struggle with. In this context, we have organized SemEval 2022 Task 4: Patronizing and Condescending Language (PCL) Detection. This task has attracted more than 300 participants, organized in 77 teams, during the official competition. The competition remains open on CodaLab to encourage further research on this topic[1].

2 Related Work

As already mentioned in the introduction, PCL has been extensively studied within the context of sociolinguistics (Margić, 2017; Giles et al., 1993; Huckin, 2002; Chouliaraki, 2006). Within NLP, however, modelling of patronizing discourse has only received limited attention. As a notable exception, Wang and Potts (2019) compiled a corpus of Reddit comments, which were annotated as using a condescending tone or not. Note that in contrast to our SemEval task, their work did not specifically focus on vulnerable communities. In our previous work (Perez-Almendros et al., 2020), we introduced *Don't Patronize Me!*, which is, to the best of our knowledge, the first annotated corpus of PCL towards vulnerable communities. This corpus was used as the training data for the SemEval task. Some other works have studied types of discourse that are highly related to condescension, including Sap et al. (2020), who studied how certain uses of language indicate power relations, Mendelsohn et al. (2020), who discussed the dehumanization of minorities through language and Zhou and Jurgens (2020), who investigated how some expressions of condolences and empathy interplay with authoritative voices in online communities.

3 Dataset

The seed material for this task is *Don't Patronize Me!* (DPM), an annotated dataset with Patronizing and Condescending Language towards vulnerable communities, which was introduced in our previous work (Perez-Almendros et al., 2020). This dataset contains 10,469 paragraphs, which were used as the training set for the SemEval task. To create the test set for this task, we annotated 3,898 additional paragraphs, following the same process. All paragraphs were extracted from news stories from media in 20 English speaking countries, originally provided by the News on Web (NoW) corpus[2] (Davies, 2013).

We used a keyword-based strategy to collect paragraphs, focusing on texts in which vulnerable communities are mentioned (e.g., refugees or homeless). The data was annotated by three annotators, with backgrounds in communication, media and data science. For the main dataset, two annotators annotated the instances with the following labels: 0 (not PCL), 1 (borderline), and 2 (PCL), achieving an inter-annotator agreement (IAA) of 41% for the raw annotations and 61% when removing borderline cases. For all the total disagreements (paragraphs labeled 0 by one annotator and 2 by the other), the third annotator acted as a referee, providing a final label. The final dataset uses a scale from 0 to 4, indicating the level of agreement between the annotators. Labels 0 and 4 correspond to clearly not condescending and clearly condescending (i.e. both annotators assigned 0 or both assigned 2), label 2 means that both annotators marked that paragraph as a borderline case (1-1), and labels 1 and 3 correspond to cases where either one of the annotators assigned the borderline label (0-1 or 1-2), or there was a disagreement that was resolved by the third annotator. Each positive example from the dataset is furthermore labelled with one or more PCL categories. We briefly recall the meaning of these categories.

Unbalanced power relations (UNB): the author entitles themselves as being in a privileged situation, considering themselves as *saviours* of those in need (Bell, 2013; Straubhaar, 2015).

Shallow solution (SHAL): a charitable, superficial and short-term action is presented as life changing.

Presupposition (PRES): stereotypes and *clichés* are used to describe a community, relying on assumptions without having all the information.

Authority voice (AUTH): the author stands as spokesperson and defendant of the community or individual and/or allows themselves to give expert advice about how to overcome underprivileged situations.

Metaphor (MET): the author describes a difficult situation in a more poetic way through

[1]https://competitions.codalab.org/competitions/34344

[2]Used with permission from the author.

Cat.	Examples
UNB	*They deserve another opportunity* or *You can make a difference in their lives.*
SHAL	*Raise money to combat homelessness by curling up in sleeping bags for one night.*
PRES	*Elderly or disabled people who are simply unable to evacuate due to physical limitations.*
AUTH	*Accepting their situation is the first step to having a normal life.*
MET	*Poor children might find more obstacles in their race to a worthy future.*
COMP	*[...] discarded in the streets of Europe [...]*
MERR	*Her mom is disabled and living with her gives her strength to face everyday's life* or *Refugees are wonderful people.*

Table 1: Examples of the different PCL categories.

figures-of-speech such as metaphors and euphemisms.

Compassion (COMP): the message uses flowery wording to reflect on the vulnerability or toughness of the situation, raising a feeling of pity among the audience.

The poorer, the merrier (MERR): the author praises the vulnerability, granting positive values to all members of a vulnerable community and showing their admiration.

Table 1 contains examples for each of these categories. The average of the IAA among categories is 57.43%[3]. It is worth mentioning that the distribution of labels is highly unbalanced in our dataset, with only around 9.5% of the inputs being labeled as containing PCL (positive cases). For the categories, the distribution is as follows: 73% UNB, 19% SHALL, 23.1% PRES, 23.6% AUTH, 48.7% COMP, 20.1% MET and 4.1% MERR.

4 Task Description

The aim of the proposed task is to identify the presence of PCL (Subtask 1), and to identify the categories of PCL that are present in a given paragraph (Subtask 2).

Training data The 10,469 annotated paragraphs from the DPM corpus were provided as training data. To frame Subtask 1 as a binary classification problem, paragraphs with labels 0 and 1 were considered as negative examples, while paragraphs with labels 3 and 4 were considered as positive examples of PCL. The original labels on the scale from 0 to 4 were also made available. The 993 positive examples in the training data are labelled with the corresponding PCL categories. Span annotations for these categories were also provided.

Test data A total of 3,898 paragraphs were released as test set, with the same format and meta-information as the training set, but without labels and span annotations. Paragraphs initially labelled as 2 were excluded from the test data, as these correspond to borderline cases.

External resources We welcomed the use of external resources in this task. Participants were encouraged to explore transfer learning or data augmentation techniques with a variety of source corpora and language resources.

Evaluation System submissions were ranked in the two subtasks as follows: **Subtask 1**: F1 score for the positive class. **Subtask 2**: Macro-averaged F1 over all categories.

4.1 Participation Framework

The task was hosted on CodaLab[4], with participants needing to register and submit their results through the platform. The competition involved the following three phases:

- **Practice phase:** The 10,469 paragraphs from the training data were split into 8,376 training paragraphs and 2,095 validation paragraphs. This was done to allow participants to compare their systems on a public leader board. The training-validation split respected the natural distribution of labels in the data.

- **Evaluation phase:** This was the official evaluation phase for the SemEval competition. The test data was released and the leader board for this phase remained hidden to prevent participants from fine-tuning their systems on the test data. Each participant was allowed two different submissions for each subtask.

- **Post-evaluation phase:** The learderboard for the evaluation phase and the official ranking for each subtask were published, as the SemEval competition ended. Participation in the SemEval task is no longer possible, but the

[3]See Perez-Almendros et al. (2020) for further details.

[4]https://competitions.codalab.org/competitions/34344

competition remains open on CodaLab to allow participants to re-test and further improve their systems.

5 Results and Discussion

A total of 77 different teams participated in the evaluation phase of our task, with 145 valid submissions for Task 1 and 84 for Task 2. For the competition, we allowed a maximum of 2 submissions per team. A total of 42 out of 77 teams outperformed the baseline for Subtask 1, while 37 out of 48 outperformed the baseline for Subtask 2. Tables 2 and 3 present the rankings for Subtask 1 and 2, respectively, where we have only listed the best performing system for each team. For Subtask 1, the best-performing systems used the following strategies:

Team PALI-NLP used an ensemble of pre-trained RoBERTa models (Liu et al., 2019). While training, they applied grouped Layer-Wise Learning Rate Decay, a variant of LLRD (Howard and Ruder, 2018), based on the idea that different layers capture different types of information (Yosinski et al., 2014). By optimizing the learning rate in different layers, the model captures more diverse and fine-grained linguistic features of PCL. To tackle the class imbalance in the dataset, they use weighted random samples (Hashemi and Karimi, 2018) to emphasize the positive instances.

Team STCE created adversarial examples to train an ensemble model of RoBERTa and DeBERTa (He et al., 2020). They also used weighted samples to address the class imbalance and explored different loss functions, establishing Cross Entropy and the contrastive loss algorithm NT-Xent introduced by Chen et al. (2020) as first and second loss function, respectively.

For Subtask 2, the best-performing systems used the following strategies:

Team BEIKE NLP participated with a system based on prompt learning (Petroni et al., 2019; Brown et al., 2020). They first reformulate PCL detection as a cloze prompt task and then fine-tune a pre-trained DeBERTa model.

Team PINGAN Omini-Sinitic proposed an ensemble model which used prompt training and a label attention mechanism, by adding a new label-wise attention layer ((Dong et al., 2021; Vu et al., 2021). Their system over-samples the positive examples. They also use a form of transfer learning from Subtask 1 to Subtask 2, by pre-training on Subtask 1 and using the resulting model as the starting point for training a model for Subtask 2.

For both sub-tasks, unsurprisingly, most systems rely on pre-trained language models, although a few teams have used CNN, LSTM, SVM or Logistic Regression based systems (XU, PC1, I2C, Ryan Wang, McRock, Amrita_CEN, SATLab and Team Lego, among others), or an ensemble of some of the above together with language models (UTSA_NLP, Taygete). Although the use of language models usually outperformed other systems in this task, some LSTM models, such as the one submitted by team Xu, achieved competitive results.

The ensembling of different models has also been a popular technique. Other strategies that proved successful include adversarial training, data augmentation and multitask learning. In the following, we summarize how these techniques have been used by the different systems.

Ensemble learning Ensembling different models has previously been found useful for text classification (Nozza et al., 2016; Kanakaraj and Guddeti, 2015; Fattahi and Mejri, 2021). Accordingly, ensembling was one of the most common strategies for improving on baseline PCL detection methods. Most of the teams combined different language models (e.g. PALI-NLP, STCE, PINGAN Omini-Sinitic,PAI_Team, LRL_NC, SSN_NLP_MLRG, ASRtrans, amsqr, UMass PCL). Considering the choice of language models, the most successful systems either used RoBERTa, DeBERTa or an ensemble which included the former ones and other models. For instance, these models were used by the best performing teams for both subtasks, i.e. PALI-NLP and STCE for Subtask 1 and BEIKE NLP and PINGAN Omini-Sintic for Subtask 2. To fine-tune the language models effectively, incorporating a contrastive loss function, in addition to the standard cross-entropy loss, has also proved useful. Finally, it should be noted that the combination of language models with different types of neural networks (Taygete, UTSA_NLP) has also proven useful.

	TEAM	P	R	F1
1	PALI-NLP	64.6	65.6	65.1
2	stce	63.3	66.9	65.0
3	ymf924	63.8	65.6	64.7
4	BEIKE NLP	61.2	67.2	64.1
5	holdon	60.3	67.5	63.7
6	cnxup	62.7	64.7	63.7
7	abcxyzw	58.8	68.5	63.3
8	nowcoder	58.2	68.5	62.9
9	PINGAN Omini-Sinitic	61.8	63.7	62.7
10	bigemo	57.1	69.4	62.7
11	Leo_team	60.1	64.0	62.0
12	PAI-Team	66.3	57.7	61.7
13	Anonymus	53.5	70.4	60.8
14	BLING	63.5	55.5	59.3
15	Taygete	53.6	66.3	59.2
16	NLP-Commonsense Reasoning team	61.2	56.8	58.9
17	GUTS	61.3	54.9	57.9
18	DH-FBK	64.2	52.7	57.9
19	ULFRI	56.4	58.7	57.5
20	TUG-CIC	60.2	54.9	57.4
21	amsqr	54.8	59.9	57.2
22	UMass PCL	52.9	58.4	55.5
23	LastResort	51.5	59.9	55.4
24	Team Double_A	47.2	66.6	55.2
25	thetundramanagainstpcl	54.3	55.5	54.9
26	Xu	46.2	66.9	54.6
27	ML_LTU	58.0	51.4	54.5
28	ZYBank-AI	54.8	53.9	54.4
29	Team LRL_NC	60.7	49.2	54.4
30	CS-UM6P & ESL	55.2	53.3	54.3
31	Felix&Julia	40.1	77.3	52.8
32	Stanford ACM	40.2	76.7	52.7
33	UtrechtUni	44.6	62.5	52.0
34	CSECU-DSG	59.0	46.4	51.9
35	Sapphire	59.4	46.1	51.9
36	Ablimet	61.5	44.8	51.8
37	SSN_NLP_MLRG	42.3	66.6	51.7
38	Team PiCkLe	46.0	58.0	51.3
39	sua	54.0	48.6	51.2
40	UCL xNSI	41.5	65.3	50.7
41	MS@IW	50.2	51.1	50.6
42	University of Bucharest Team	49.1	50.8	49.9
43	**RoBERTa Baseline**	**39.4**	**65.3**	**49.1**
44	rematchka	44.5	53.9	48.8
45	fengxing	63.8	39.4	48.7
46	flerynn	67.2	38.2	48.7
47	Team YNU-HPCC	65.9	36.6	47.1
48	niksss	51.8	42.0	46.3
49	JustTeam	55.0	39.8	46.2
50	BWQ	51.0	41.3	45.6
51	Tesla	36.0	57.7	44.3
52	ASRtrans	35.6	58.4	44.2
53	RNRE NLP	39.0	50.2	43.2
54	SATLab	34.8	55.2	42.7
55	J.U.S.T-DL	49.0	37.5	42.5
56	MaChAmp	58.8	32.8	42.1
57	I2C	61.1	31.2	41.3
58	SMAZ	36.3	47.6	41.2
59	MASZ	36.3	47.6	41.2
60	Amrita_CEN	32.2	52.1	39.8
61	Anonymus	27.6	59.9	37.8
62	matan-bert	35.4	40.4	37.7
63	Team LEGO	24.8	56.5	34.5
64	TüSoXi	38.8	29.3	33.4
65	RNRE NLP RFC	30.0	36.9	33.1
66	jct_meir	25.3	47.0	32.9
67	isys	22.4	59.3	32.5
68	AliEdalat team	18.4	87.1	30.3
69	ms_pa	23.4	39.1	29.3
70	Waad	64.0	18.0	28.1
71	Ryan Wang	17.0	60.9	26.6
72	PC1	37.8	18.6	25.0
73	UTSA_NLP	14.0	35.0	20.0
74	yaakov	11.2	10.1	10.6
75	ilan	14.5	6.0	8.5
76	Jiaaaaaa	8.2	6.3	7.1
77	Anonymus	29.7	3.5	6.2
78	Anonymus	10.6	2.8	4.5

Table 2: SemEval Task 4: Ranking by teams for Subtask 1: Binary Classification. The table reports Precision (%), Recall (%) and F1-Score (%) for the positive class.

	TEAM	UNB	SHAL	PRES	AUTH	MET	COMP	MERR	Avg
1	BEIKE NLP	65.6	52.9	36.9	40.7	35.9	49.2	47.1	46.9
2	PINGAN Omini-Sinitic	59.7	53.1	41.7	43.4	42.7	51.3	15.4	43.9
3	PAI_Team	57.6	45.2	35.2	39.4	38.4	44.5	26.7	41.0
4	stce	62.2	54.8	38.1	32.8	33.3	51.0	8.7	40.1
5	PALI-NLP	61.8	54.1	37.7	32.8	32.8	51.2	8.7	39.9
6	Leo_team	57.3	47.0	28.8	36.1	34.8	47.4	27.0	39.8
7	Anonymus	59.9	49.1	38.5	37.1	35.0	48.6	8.3	39.5
8	ymf924	61.6	54.1	36.8	31.3	33.3	50.0	8.7	39.4
9	bigemo	62.5	56.1	38.0	24.3	31.3	49.4	8.7	38.6
10	holdon	62.2	56.1	32.9	23.1	33.3	48.7	8.7	37.9
11	cnxup	60.2	53.3	30.6	24.1	40.0	48.1	8.7	37.8
12	Taygete	59.7	45.8	33.3	21.8	30.4	53.6	18.8	37.6
13	DH-FBK	52.5	36.2	27.0	37.7	31.9	46.0	30.3	37.4
14	abcxyzw	60.7	53.3	34.5	21.8	32.8	50.0	8.3	37.4
15	nowcoder	59.8	50.0	32.2	22.8	39.4	47.8	8.3	37.2
16	GUTS	55.6	47.4	24.0	34.3	25.6	44.4	27.6	37.0
17	BLING	55.1	38.9	23.4	29.0	31.5	50.9	26.7	36.5
18	UMass PCL	53.9	42.4	29.1	30.7	33.3	40.8	23.5	36.3
19	CS-UM6P & ESL	57.0	42.0	25.7	25.2	20.5	46.8	21.4	34.1
20	Fengxing	46.4	46.3	23.0	26.5	33.3	38.7	24.0	34.0
21	Team LRL_NC	52.1	42.7	25.2	30.4	28.8	43.3	14.8	33.9
22	thetundramanagainstpcl	50.5	50.0	18.4	16.5	20.3	41.5	24.0	31.6
23	Xu	55.0	48.4	28.0	24.0	13.6	49.0	0.0	31.1
24	SATLab	42.4	33.1	17.0	23.2	17.5	31.5	14.2	25.6
25	Felix&Julia	36.6	35.1	17.6	22.1	21.1	28.5	16.7	25.4
26	AliEdalat team	53.9	37.7	25.6	26.2	13.5	11.3	9.1	25.3
27	Tesla	43.7	38.3	16.3	19.2	17.9	35.7	0.0	24.5
28	Waad	36.9	33.3	17.5	15.3	16.5	28.7	19.5	24.0
29	ms_pa	32.3	32.9	19.2	20.6	22.2	26.4	7.1	23.0
30	rematchka	37.7	21.4	18.8	21.2	15.5	26.1	13.0	22.0
31	Team Double_A	33.5	31.9	18.4	19.1	23.4	24.5	0.0	21.5
32	SSN_NLP_MLRG	34.6	33.8	20.7	19.3	12.1	27.7	0.0	21.2
33	ASRtrans	18.6	8.8	8.3	19.8	13.2	27.8	35.7	18.9
34	MaChAmp	30.4	21.3	3.6	10.9	30.8	5.0	6.3	15.5
35	Team PiCkLe	10.9	22.5	14.4	21.0	19.2	6.5	11.5	15.2
36	LastResort	15.8	24.8	10.0	9.3	16.0	11.3	14.8	14.6
37	Ablimet	12.6	14.1	6.5	7.2	14.0	17.2	17.1	12.7
38	**RoBERTa Baseline**	**35.4**	**0.0**	**16.7**	**0.0**	**0.0**	**20.9**	**0.0**	**10.4**
39	BWQ	16.0	12.5	7.2	9.7	7.0	11.4	3.9	9.7
40	Stanford ACM	16.0	26.5	4.2	0.0	0.0	8.6	12.1	9.6
41	Team LEGO	11.8	20.6	1.9	6.4	6.5	10.2	0.0	8.2
42	CSECU-DSG	33.4	0.0	0.0	0.0	0.0	21.8	0.0	7.9
43	University of Bucharest Team	14.8	21.7	3.5	0.0	3.9	8.3	0.0	7.4
44	PC1	11.8	12.0	6.1	8.7	2.6	8.9	0.0	7.2
45	Team YNU-HPCC	10.9	0.8	3.5	3.3	0.0	5.8	0.0	3.5
46	NLP-Commonsense Reasoning team	9.7	0.2	0.0	3.2	3.2	4.4	1.1	3.1
47	Jiaaaaaa	2.8	1.9	0.0	2.0	0.0	4.8	6.9	2.6
48	Anonymus	5.9	8.3	0.0	2.4	0.0	1.4	0.0	2.6
49	niksss	0.0	1.0	0.0	0.0	0.0	0.0	1.1	0.3

Table 3: SemEval Task 4: Ranking by teams for Subtask 2: Categories Classification. The table reports F1-Score (%) for each one of the categories and the macro-averaged F1-score (%) for all categories. The categories stand for: Unbalanced Power Relations (UNB), Shallow Solution (SHAL), Presupposition (PRES), Authority Voice (AUTH), Metaphors (MET), Compassion (COMP) and The poorer, the merrier (MERR).

Balancing class distribution The class imbalance in the dataset has been addressed by participating teams in different ways. Some teams opted for downsampling the number of negative examples (Ryan Wang, LastResort, MS@IW), while others tried a cost-sensitive learning approach to address this issue (Amrita_CEN). However, the most popular approach to balance the class distribution has been through data augmentation (amsqr, Xu, Utrech Uni, UMass PCL, among others). To create new positive examples, participants have used strategies such as the use of large generative models like GPT3 (Brown et al., 2020) or T5 (Raffel et al., 2020) (MS@IW, PINGAN Omini-Sinitic and Tesla); back-translation (Taygete); the addition of synonymous sentences to the original data (I2C), or the application of the so-called Easy Data Augmentation methods, a set of simple but effective techniques such as synonym replacement, random insertion, random swap, and random deletion (AliEdalat) (Wei and Zou, 2019; Rastogi et al., 2020).

External resources Various types of external resources have been used. For example, lexical databases such as WordNet (Miller, 1995) have been used to augment, enrich and improve the training data (Ali Edalat). Datasets from related tasks, including TalkDown (Wang and Potts, 2019), and two metaphor detection datasets, namely MOH (Mohammad et al., 2016) and VUA (Steen et al., 2010), have been used both for pre-training and / or for data augmentation by different teams. PAWS, a dataset with Paraphrase Adversaries from Word Scambling, (Zhang et al., 2019) and xTREME, a bechmark for Cross-Lingual Transfer Evaluation of Multilingual Encoders (Hu et al., 2020), have also been used to improve several systems (Ali Edalat, ASRtrans, Tesla, MaChAmp). Other related NLP challenges have served as auxiliary tasks for pre-training PCL models (AliEdalat, UMass PCL), although such strategies have not always been successful (ULFRI). The MaChAmp team used 7 SemEval-2022 tasks, including ours, for training a model based on multi-task learning. The DH-FBK team also opted for multi-task learning, but they only used the data from the Don't Patronize Me dataset itself to create auxiliary tasks. For instance, they trained their model to predict the uncertainty of a label in Subtask 1, using the fine-grained set of labels (0-4); the agreement of the annotators in Subtask 2; the spans where the categories were present; or the country of origin of the news outlets. AliEdalat similarly used the meta-information from the Don't Patronize Me dataset as additional features for training their model.

Prompt learning Using prompts has also proven useful for PCL detection (BEIKE NLP, PINGAN Omini-Sintic, Ablimet). Specifically, the teams used prompts such as "*[paragraph]* is *[label]*", or "is *[paragraph] [label]*?" where *[paragraph]* is the original input. For Subtask 1, *[label]* is a natural language description of the binary class label (e.g. *"is (not) condescending or patronizing"*). For Subtask 2, *[label]* is the label of a given PCL category.

6 Conclusions

Patronizing and Condescending Language detection is a relatively new challenge for the NLP community. However, the high level of participation in this task has provided the community with valuable new insights about how to tackle this problem. A total of 42 out of 77 teams in Subtask 1 and 37 out of 48 for Subtask 2 outperformed the RoBERTa baseline. The performance of the best-performing systems shows that a judicious usage of state-of-the-art text classification techniques can bring significant benefits to PCL detection, especially when it comes to addressing the relative scarcity of the available training data and closely related external resources. However, there still remains considerable scope for further improvements. It is our expectation that further improvements may need to rely on techniques that are specifically targeted at PCL, e.g. by exploiting insights from linguistics about the linguistic features of PCL, or by building explicit models of stereotypes of vulnerable communities.

References

Pepa Atanasova, Preslav Nakov, Georgi Karadzhov, Mitra Mohtarami, and Giovanni Da San Martino. 2019. Overview of the clef-2019 checkthat! lab on automatic identification and verification of claims. task 1: Check-worthiness. In *CEUR Workshop Proceedings, Lugano, Switzerland.*

Alberto Barrón-Cedeno, Tamer Elsayed, Preslav Nakov, Giovanni Da San Martino, Maram Hasanain, Reem Suwaileh, and Fatima Haouari. 2020. Checkthat! at clef 2020: Enabling the automatic identification and verification of claims in social media. In *European Conference on Information Retrieval*, pages 499–507. Springer.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63.

Katherine M Bell. 2013. Raising Africa?: Celebrity and the rhetoric of the white saviour. *PORTAL Journal of Multidisciplinary International Studies*, 10(1).

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Lilie Chouliaraki. 2006. *The spectatorship of suffering*. Sage.

Lilie Chouliaraki. 2010. Post-humanitarianism: Humanitarian communication beyond a politics of pity. *International journal of cultural studies*, 13(2):107–126.

Niall J Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4.

Giovanni Da San Martino, Alberto Barrón-Cedeno, Henning Wachsmuth, Rostislav Petrov, and Preslav Nakov. 2020. Semeval-2020 task 11: Detection of propaganda techniques in news articles. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1377–1414.

Mark Davies. 2013. Corpus of news on the web (now): 3+ billion words from 20 countries, updated every day. *Retrieved January*, 25:2019.

Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. Semeval-2017 task 8: Rumoureval: Determining rumour veracity and support for rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 69–76.

Hang Dong, Víctor Suárez-Paniagua, William Whiteley, and Honghan Wu. 2021. Explainable automated coding of clinical notes using hierarchical label-wise attention networks and label embedding initialisation. *Journal of biomedical informatics*, 116:103728.

Jaouhar Fattahi and Mohamed Mejri. 2021. Spaml: a bimodal ensemble learning spam detector based on nlp techniques. In *2021 IEEE 5th International Conference on Cryptography, Security and Privacy (CSP)*, pages 107–112. IEEE.

Susan T Fiske. 1993. Controlling other people: The impact of power on stereotyping. *American psychologist*, 48(6):621.

Michel Foucault. 1980. *Power/knowledge: Selected interviews and other writings, 1972-1977*. Vintage.

Howard Giles, Susan Fox, and Elisa Smith. 1993. Patronizing the elderly: Intergenerational evaluations. *Research on Language and Social Interaction*, 26(2):129–149.

Mahdi Hashemi and Hassan Karimi. 2018. Weighted machine learning. *Statistics, Optimization and Information Computing*, 6(4):497–525.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Monisha Kanakaraj and Ram Mohana Reddy Guddeti. 2015. Performance analysis of ensemble methods on twitter sentiment analysis using nlp techniques. In *Proceedings of the 2015 IEEE 9th international conference on semantic computing (IEEE ICSC 2015)*, pages 169–170. IEEE.

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

Branka Drljača Margić. 2017. Communication courtesy or condescension? linguistic accommodation of native to non-native speakers of english. *Journal of English as a lingua franca*, 6(1):29–55.

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A framework for the computational linguistic analysis of dehumanization. *Frontiers in Artificial Intelligence*, 3:55.

Debra L Merskin. 2011. *Media, minorities, and meaning: A critical introduction*. Peter Lang.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33.

Preslav Nakov, Alberto Barrón-Cedeno, Tamer Elsayed, Reem Suwaileh, Lluís Màrquez, Wajdi Zaghouani, Pepa Atanasova, Spas Kyuchukov, and Giovanni Da San Martino. 2018. Overview of the clef-2018 checkthat! lab on automatic identification and verification of political claims. In *International conference of the cross-language evaluation forum for european languages*, pages 372–387. Springer.

Janice Nathanson. 2013. The pornography of poverty: Reframing the discourse of international aid's representations of starving children. *Canadian Journal of Communication*, 38(1).

Sik Hung Ng. 2007. Language-based discrimination: Blatant and subtle forms. *Journal of Language and Social Psychology*, 26(2):106–122.

David Nolan and Akina Mikami. 2013. 'the things that we have to do': Ethics and instrumentality in humanitarian communication. *Global Media and Communication*, 9(1):53–70.

Debora Nozza, Elisabetta Fersini, and Enza Messina. 2016. Deep learning and ensemble methods for domain adaptation. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 184–189. IEEE.

Carla Perez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

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

Chetanya Rastogi, Nikka Mofid, and Fang-I Hsiao. 2020. Can we achieve more with less? exploring data augmentation for toxic comment classification. *arXiv preprint arXiv:2007.00875*.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Association for Computational Linguistics*.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna A Kaal, and Tina Krennmayr. 2010. Metaphor in usage. *Cognitive Linguistics*, 21(4).

Rolf Straubhaar. 2015. The stark reality of the 'white saviour'complex and the need for critical consciousness: A document analysis of the early journals of a freirean educator. *Compare: A Journal of Comparative and International Education*, 45(3):381–400.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2021. A label attention model for icd coding from clinical text. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3335–3341.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388.

Clint C Wilson and Felix Gutierrez. 1985. Minorities and the media. *Beverly Hills, CA, London: Sage*.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447, Barcelona (online). International Committee for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1298–1308.

Naitian Zhou and David Jurgens. 2020. Condolences and empathy in online communities. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 609–626.

JUST-DEEP at SemEval-2022 Task 4: Using Deep Learning Techniques to Reveal Patronizing and Condescending Language

Mohammad Makahleh and **Naba Bani Yaseen** and **Malak Abdullah**
Jordan University of Science and Technology
Irbid Jordan
maalmakahleh20, nmbaniyaseen21 @cit.just.edu.jo, mabdullah@just.edu.jo

Abstract

Classification of language that favors or condones vulnerable communities (e.g., refugees, homeless, widows) has been considered a challenging task and a critical step in NLP applications. Moreover, the spread of this language among people and on social media harms society and harms the people concerned. Therefore, the classification of this language is considered a significant challenge for researchers in the world. In this paper, we propose JUST-DEEP architecture to classify a text and determine if it contains any form of patronizing and condescending language (Task 4- Subtask 1). The architecture uses state-of-art pre-trained models and empowers ensembling techniques that outperform the baseline (RoBERTa) in the SemEval-2022 task4 with a 0.502 F1 score.

1 Introduction

The language used when talking about other people significantly impacts society and individuals. Talking about others and using caring and sympathetic language to express them causes them to be concerned, regardless of the author's intention, which is often to help others by raising awareness of their cause. Unfair treatment of vulnerable groups on social media increases exclusion and inequality(Kučak et al., 2018).

Several researchers study modeling language that intentionally undermines others, such as offensive language or hate speech (Zampieri et al., 2019; Faraj and Abdullah, 2021). PCL modeling is still a relatively new topic of research in NLP. For illustration, the use of PCL in the media is frequently unconscious, subtler, and more subjective than the sorts of discourse that are typically addressed in NLP. To the best of our knowledge, a particular focus on PCL for vulnerable communities has not yet been considered. Through a broader context, some work on PCL had been studied on communication between two people, such as in social media interactions, where others patronize an individual. For example, the authors in (Inui et al., 2019) published the talk down corpus for detecting condescension in Reddit comment-reply pairs. Finding or creating a high-quality dataset that covers all or most cases of PCL is very difficult and requires significant effort from specialized researchers. As researchers in NLP, we use it to investigate PCL in vulnerable communities(Perez-Almendros et al., 2020).

In this paper, we describe our model on task 4 of Semeval 2022 sub task1 (P'erez-Almendros et al., 2022) to identify PCL and categorize the linguistic techniques used to express it. Specifically when referring to communities identified as being vulnerable to unfair treatment in the media. We compare the outcomes of multiple pre-trained models (BERT, RoBERTa, and ensemble on them). We furthermore show the effect of tuning their hyperparameters values (batch-size and epoch) on model prediction. Also, we illustrate our proposed architecture where we feed the data to an ensemble model with the stacking of 2 BERT models and -in parallel- to another ensemble model with the stacking of 2 RoBERTa models. Finally, the results of the two ensembling models are fed to a max voting ensemble, which predicts the outcome with achieving the best F1 score at 0.502.

The rest of the paper is organized as follows; section 2 presents a Related Work, and section 3 offers Methodology. Results are discussed in section 4. Finally, section 5 presents conclusion.

2 Related Work

Many researchers applied machine learning models to classify text (Abdullah and Shaikh, 2018; Qawasmeh et al., 2019). Lai et al.(Lai et al., 2015) classified text using a recurrent convolutional neural network. They captured the key components in texts using the max-pooling layer from word representations. Then they grabbed contextual information from it. To check the efficacy of the proposed method, they conducted several experiments on four commonly used data sets. As a result, they found that their suggested method outperforms the latest methods in many data sets.

Gunal et al.(Kowsari et al., 2019) studied text classification algorithms on several sides. Like limitations of each technique and its application in real-world situations are discussed. They found some steps useful in decreasing the time complexness and memory complexity of existing text classification algorithms like central component analysis and incidental projection, etc.

Using deep neural networks with LSTM modules, Semberecki et al.(Semberecki and Maciejewski, 2017) classified text documents. They tried to construct feature vectors, which represent the documents to be categorized: each feature vector represents the sequence of words that are included in the documents. First, they convert the terms into vector representations and then use the sequences of these vector representations as features of the documents. They evaluated the feasibility of this approach to text categorization utilizing a set of Wikipedia articles. They show that the LSTM network-based approach with documents represented as vectors achieves an accuracy of 86%.

3 Methodology

Our approach methodology can be summarized as follows: We begin by describing the dataset for this task. Then, the preprocessing step is described. Our final section describes how JUST-DEEP can identify the patronizing language in the text.

3.1 Data Set:

The SemEval-task 1 competition provided three files (rial, train, and test dataset(Perez-Almendros et al., 2020)) The files contain several columns as follows:

- par_id: the identification number for each paragraph.
- art_id: the identification number for each article.
- keyword: word related to patronizing. .
- country_code: the code for each country.
- text: the text that we want to classify.
- label: denotes if the text contains patronizing or not (4 levels 0,1,2,3 where 0 and 1 means no PCL, as well as 2 and 3, means high PCL).

3.2 Data pre-processing:

In this section we describe the basic data preprocessing steps we applied for all of our experiments:

1. In the beginning, we transform the Label column into a binary format. If the value is zero or one, we convert it to zero. And if it is two or three, we convert it to One. So with this transformation, the Label is a Binary column with two values: the Zero means that there are no PCL in the text, and the value One implies that there is PCL.We converted the column to binary because we are working on sub-task one, so we want to determine if it contains PCL or not; we don't care about the degree of PCL.

2. Pre-trained models don't work with raw text, so we converted the text into numbers and added unique tokens to separate sentences at the beginning and end of each sentence. Then we pass the resulting sequence to the models to perform the classification process.

3. Pre-trained models work with fixed-length sequences. So we used a simple strategy to choose the appropriate maximum length for all sequences. First, we found that most series have a size of 160, so we set the size of all series to 160.

3.3 JUST-DEEP Architecture:

Our task aims to detect whether a text contains PCL language or not. As shown in Figure 1, JUST-DEEP architecture uses multiple pre-trained language models (BERT and RoBERTa) from the transformers library.

The first step is data pre-processing. The training dataset is fed to the augmentation processor, responsible for adding more data to the training dataset since it is originally imbalanced data where the number of Zero class instances is ten times the One class instances. The augmentation processor input is all One instances text. The processor augments the text of the input instances by adding the same instance to the primary dataset multiple times with slightly different text but with the same meaning. Next, the textual data is processed into their corresponding embeddings(tokens) to feed them to the classifiers.

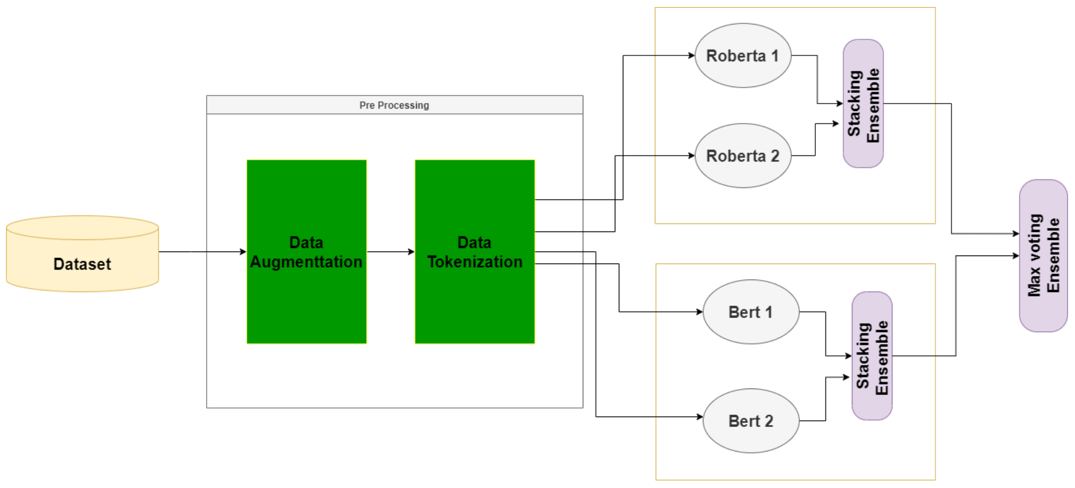


Figure 1: Model Architecture

The second step is the classification step. Finally, the input tokens are fed in parallel to two ensembling models; the first is composed of stacking of 2 BERT models, and the second contains stacking of 2 RoBERTa models.

Finally, the predictions of the two ensembling models are joined by a max voting classifier to produce the final prediction output. We conducted several experiments with different models and hyperparameters, but the JUST-DEEP architecture achieved the best results.

Figure 2 show an example use case for architecture. If we fed the sentence *'Fast food employee who fed disabled man becomes internet sensation'* which is an example from the train data labeled as 1 (contains PCL). In augmentation step we generate more instances with same meaning of this row by the augmentation processor which might add data like *'Fast food worker who fed weakened guy becomes internet rumor'*. Next, the sentence is converted into numeric token and passed to both ensemble models.
For instance, the first ensemble model which contains stacking of 2 RoBERTa models produce the prediction of 1 for this row. The other ensemble model with stacking of 2 Bert classifiers predicts Zero as a class label. Finally, the last step which implement a max voting will generate One as the final output label for the instance.

4 Result

We performed several experiments to determine which is the best suitable model for this task and which model produces the highest value of F1 score. First, we experimented with BERT and RoBERTa pre-trained models and tuned their hyperparameters (batch_size and epoch) to achieve the maximum possible F1 score. Table 1 shows the experiments we did, the models, their parameters, and the value of the f1 score we got in each experiment in the test dataset.

As a first step, we explored BERT and RoBERTa's best hyperparameters, such as batch_size, epochs, and max sequence length. Table 2 describes these hyperparameters.

We use five different models: BERT model, RoBERTa model, stacking of 2 BERT, stacking of 2 RoBERTa, and JUST-DEEP model as described in the architecture section. As shown in the table1 model, JUST-DEEP achieves the best results compared to the rest of the models we tested, as it reached an F1 score value equal to 0.502.

As we explained earlier, in the JUST-DEEP model, we trained the dataset with augmentation. Then we applied the Stacking Ensemble Technique separately on 2 BERT Models and 2 RoBERTa Models. Then, we took the Max voting between the two models as a result; therefore, this model produces the best results because it combines the two models that we used, BERT and RoBERTa.

The other models show less favorable results; the application of the stacking ensemble technique to the 2 BERT models gives the highest results after the JUST-DEEP, where the F1 result is close to 46%, followed by the model resulting from the application of the stacking ensemble technique to the 2 RoBERTa, where it achieved results close to 0.44. Then, RoBERTa's model achieved the F1 with a score of 0.43. The worst model was Bert achieved 0.42.

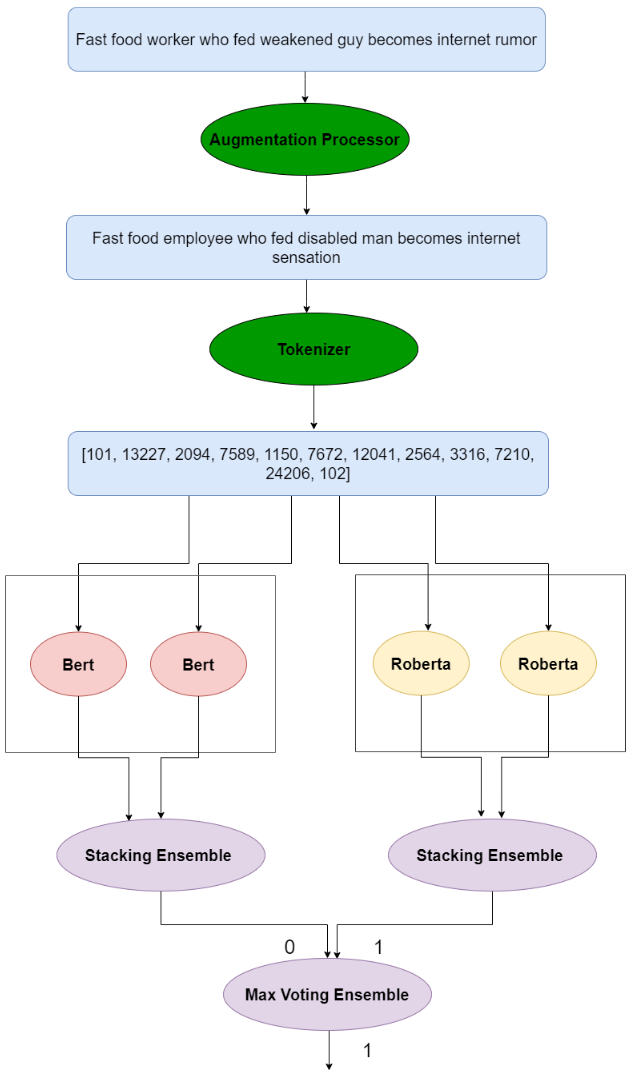


Figure 2: Example Description

5 Conclusion

Deep learning helped in the development of many aspects these days, and with in-text classification, we become to have the ability to make a lot of applications related to it and so on. In this paper, we describe our JUST-DEEP model solving the SemEval-2022 Task 4 Subtask 1 to check if the text contains any form of PCL. The JUST-DEEP model obtained an F1 score of 0.5 using ensembling of pre-trained language models BERT and RoBERTa. Our strategy depends on training data set with augmentation then applying stacking ensemble technique to 2 BERT and 2 RoBERTa and take max voting between the models to achieve F1 score higher than the other experiments we conduct where we use plain BERT and RoBERTa models with different hyperparameters.

References

Malak Abdullah and Samira Shaikh. 2018. Teamuncc at semeval-2018 task 1: Emotion detection in english and arabic tweets using deep learning. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 350–357.

Dalya Faraj and Malak Abdullah. 2021. Sarcasmdet at semeval-2021 task 7: Detect humor and offensive based on demographic factors using roberta pretrained model. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 527–533.

Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan. 2019. Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

#	Classifier	Batch size	Epoch	F1
1	RoBERTa	16	10	0.43
2	RoBERTa	16	4	0.40
3	RoBERTa	8	10	0.42
4	RoBERTa	8	4	0.41
5	BERT	16	10	0.42
6	BERT	16	4	0.428
7	BERT	8	10	0.41
8	BERT	8	4	0.405
9	stacking of 2 RoBERTa	16	10	0.44
10	stacking of 2 RoBERTa	16	4	0.45
11	stacking of 2 RoBERTa	8	10	0.435
12	stacking of 2 RoBERTa	8	4	0.438
13	stacking of 2 BERT	16	10	0.46
14	stacking of 2 BERT	16	4	0.45
15	stacking of 2 BERT	8	10	46.5
16	stacking of 2 BERT	8	4	0.45
17	JUST-DEEP	16	10	0.5

Table 1: Our Experiments.

Parameter	Description	Values
Batch_Size	The number of samples per batch	16,8
Epoch	Training examples in both directions (backward and forward)	4,10
Dropout	Number of examples that can be neglected during training	0.3
Max Sequence Length	Sequence length the model can support	160

Table 2: Parameters

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. Text classification algorithms: A survey. *Information*, 10(4):150.

Danijel Kučak, Vedran Juričić, and Goran Ðambić. 2018. Machine learning in education-a survey of current research trends. *Annals of DAAAM & Proceedings*, 29.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Carla Perez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ethar Qawasmeh, Mais Tawalbeh, and Malak Abdullah. 2019. Automatic identification of fake news using deep learning. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 383–388. IEEE.

Piotr Semberecki and Henryk Maciejewski. 2017. Deep learning methods for subject text classification of articles. In *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 357–360.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.

PINGAN Omini-Sinitic at SemEval-2022 Task 4: Multi-prompt Training for Patronizing and Condescending Language Detection

Ye Wang **Yanmeng Wang** **Baishun Ling** **Zexiang Liao**
Shaojun Wang **Jing Xiao**
Ping An Technology, Beijing 100191, China
{wangye430,wangyanmeng219,lingbaishun271,liaozexiang901,
wangshaojun851,xiaojing661}@pingan.com.cn

Abstract

This paper describes the second-placed system for subtask 2 and the ninth-placed system for subtask 1 in SemEval 2022 Task 4: Patronizing and Condescending Language Detection. We propose an ensemble of prompt training and label attention mechanism for multi-label classification tasks. Transfer learning is introduced to transfer the knowledge from binary classification to multi-label classification. The experimental results proved the effectiveness of our proposed method. The ablation study is also conducted to show the validity of each technique.

1 Introduction

Patronizing and Condescending Language (PCL) is proposed by Pérez-Almendros et al. (2020), which builds a dataset for PCL detection. The Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022) contains two text classification tasks. Subtask 1 is a binary classification task, which requires a system to predict whether the paragraph contains any form of PCL. Subtask 2 is a multi-label classification task, which must identify which PCL categories express the condescension.

PCL Detection is a sentiment analysis task and we treated subtask 2 as an Aspect-Based Sentiment Analysis (ABSA) (Jo and Oh, 2011) task, which is also a multi-label classification to classify the sentence sentiment on different aspects.

Early works on ABSA focus on feature engineering (Wagner et al., 2014) and subsequent neural network-based methods (Wang et al., 2017). Recently, Pre-trained Language Models (LMs) such as BERT (Devlin et al., 2019), ROBERTA (Liu et al., 2019), ALBERT (Lan et al., 2020) have been proposed and brought significant improvement in various NLP applications. As there is not much improvement in ABSA task through direct implementation of those pre-trained LMs, Bert-pair (Sun et al., 2019) has been proposed to help them adapt to ABSA effectively. Furthermore, Bu et al. (2021) proposed an attention between sentence embeddings and label embeddings to focus on the crucial tokens which are related to the label. Recently, CapsNet-Bert (Jiang et al., 2019) used the attention between the label and the input with capsule network to improve the performance.

Nowadays, the Pattern-Exploiting Training (PET) (Schick and Schütze, 2021) and P-tuning (Liu et al., 2021) has been proposed to utilize the pre-trained LMs more effectively, which trains finetune model by the pre-training tasks such as Mask Language Modeling (MLM) by adding a prompt sequence to the input.

Considering the similarity between PCL and ABSA, we applied PET and Bert-pair in PCL Detection through the following steps: Firstly, we treat the subtask 1 as a prompt training task (as described in PET) by using a PCL description as the prompt. Secondly, we transform the subtask 2 from a multi-label classification task to multiple binary classification tasks by using the label names as the prompts. Thirdly, we conduct transfer learning from subtask 1 to subtask 2 to further improve the performance. Last but not least, we also proposed a label attention mechanism based on the multi-prompt training for multi-label classification.

Our contribution can be summarized as follows: we apply prompt training on PCL detection subtask 1 and 2, and prove its effectiveness on both tasks; in subtask 2, we propose a label attention mechanism with multi-prompt training; we apply transfer learning from subtask 1 to subtask 2, where the transfer learning is proved to be effective in ABSA.

2 Background

2.1 Task Description

The tasks intend to detect whether the input paragraphs have any forms of PCL, and which PCL categories are contained. The PCL categories defined

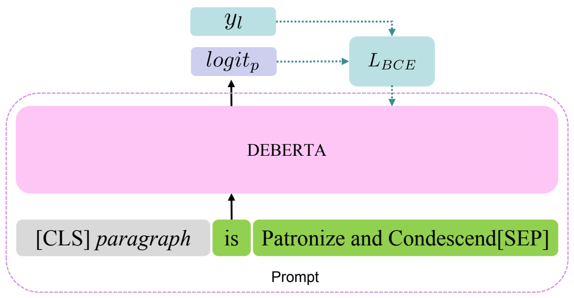


Figure 1: Subtask 1 overview, $logit_p$ is the corresponding logits of the word "is" in prompt.

in the task paper Pérez-Almendros et al. (2022) are 'Unbalanced power relations', 'Shallow solution', 'Presupposition', 'Authority voice', 'Metaphors', 'Compassion','The poorer, the merrier'. The tasks can be formalized as follows:

The training data of subtask 1 consists of tuples (q, l_{binary}), where q is a paragraph extracted from articles, and l_{binary} is the binary classification label with values 0, 1. The training data of subtask 2 consists of tuples (q, $l_{category}$), where q is a paragraph extracted from articles and $l_{category}$ is the label for multi-label multi-classification task. $l_{category}$ is a 7-digit binary vector. Each digit represents whether the paragraph contains the corresponding PCL category and it is possible for one paragraph to contain multiple PCL categories. We also defined a set A, which contains the names of categories. $a \in A$ represents a category name from all PCL categories.

2.2 DEBERTA

Our comparison of present large pre-trained Language Models(LM) showed that DEBERTA (He et al., 2021b) seems to be the most effective model. Different from other works, DEBERTA implemented a disentangled attention mechanism which utilizes the input contents and relative positions. We conduct most experiments based on DEBERTA-v3 (He et al., 2021a), which trained the DEBERTA model with replaced token detection (RTD) pre-training task, proposed by ELECTRA (Clark et al., 2020). RTD is a more sample-efficient pre-training task than replacing Mask Language Modeling (MLM). The experiments in He et al. (2021a) shows that DeBerta-v3-large model achieves better performance on GLUE benchmark even compared with larger pre-trained LMs.

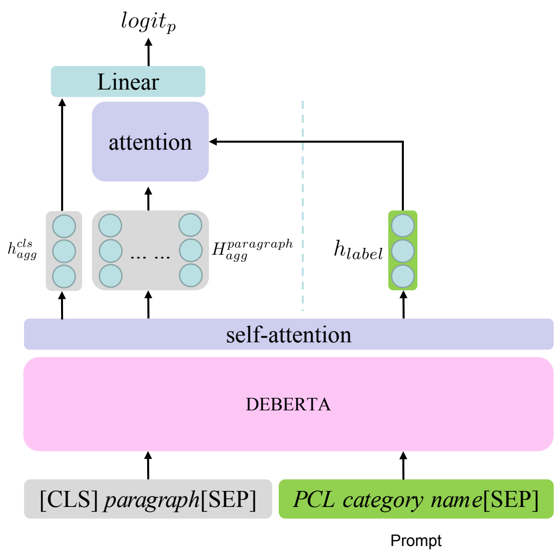


Figure 2: Subtask 2 overview, we conduct a attention mechanism between the logits of paragraph and the logits of label name.

3 System overview

3.1 Prompt Classification

Inspired by PET, we adopt a prompt design on traditional classification tasks based on pre-trained LMs. As shown in Figure 1, we adopt DEBERTA-v3 as the pre-trained LM. The input sequence consists of the original paragraph q and the prompt sequence, which is "is patronize and condescend" in subtask 1, denoted as r. The total input sequence can be described as "$[CLS]\, q\, r\, [SEP]$" for subtask1. Since the DEBERTA-v3 trained by RTD as same as ELECTRA, we don't use the hidden states of [CLS] for classification. Instead we use the hidden state of word "is" (as described in Figure 1) for a binary classification, and the label is 1 when the paragraph contains PCL, 0 when the paragraph has no PCL. The input sequence is forward to DEBERTA and the hidden states are calculated by Equation 1

$$H_q = F([q; r]) \tag{1}$$

$$logit_p = D(H^p_q) \tag{2}$$

where F is the pre-trained 24-layer transformers and D is a linear layer which classify the hidden states of prompt word from DEBERTA. Then we use the hidden states $H^p_q \in \mathbb{R}^d$ selected from H_q as the input to D, which denotes the hidden states of the prompt word "is", the d is the hidden size of DEBERTA. BCE (Binary Cross Entropy) loss, which measures the Binary Cross Entropy between the golden label and the output, is used for binary classification as Equation 3.

$$L_{BCE} = BCE(Sigmoid(logit_p), l_{binary}) \quad (3)$$

where l_{binary} is the binary label of subtask 1.

In this way, the semantics of prompt can be jointly learned with the input paragraph. By using the same training method with pre-trained LM, the representations stored in the pre-trained LM has been maximum reserved.

3.2 Multi-label Prompt Classification

For subtask 2, we transform the multi-label classification task to multi binary classification tasks by concatenating the paragraph with each PCL category label names. As shown in Figure 2, the input paragraph q is concatenated with one PCL category name a, which can be denoted as "[CLS]q[SEP]a[SEP]". We use an self-attention layer as an aggregator to aggregate the sequence embeddings, which is more effective then using the [CLS] embedding. It can be formulated as:

$$H_q = F([q; a]) \quad (4)$$

$$H_{agg} = S(H_q) \quad (5)$$

where S is the self-attention layer proposed by BERT (Devlin et al., 2019), and H_{agg} represents the aggregated hidden states of input sequence. Then we use the [CLS] aggregated hidden states H^{cls}_{agg} to classify whether the paragraph has the corresponding PCL category, which can be formulated as:

$$logit_{cls} = D(H^{cls}_{agg}) \quad (6)$$

$$L_{BCE} = BCE(Sigmoid(logit_{cls}), l_c) \quad (7)$$

where l_c is the binary label of PCL category detection, selected from the category label $l_{category}$. The real output for subtask 2 is the combination of all PCL category detection results.

From previous works, this approach have been proved effectively for improving the multi-label classification tasks. We named this approach MPrompt in this paper.

3.3 Label Attention

As shown in Figure 2, we also propose a label attention above the MPrompt approach. Although the transformers model has self-attention between every tokens, an external attention between paragraph and label prompt is still helpful for the model to focus on more important words in the paragraph. First, We split the output hidden states H_{agg} into $H^{paragraph}_{agg}$ and H^{label}_{agg}, where $H^{paragraph}_{agg}$ is the hidden states corresponding to the sequence "[CLS]q[SEP]", and H^{label}_{agg} is the hidden states corresponding to the sequence "a[SEP]". Then, the label embedding h_{label} is computed as average pooling over H^{label}_{agg}, where $h_{label} \in \mathbb{R}^d$.

As shown in equation 8, we use an attention layer to combine the h_{label} and $H^{paragraph}_{agg}$, where h_{label} is used as query (Q), and $H^{paragraph}_{agg}$ is used as key (K) and value (V) followed the definition of Scaled Dot-Product Attention in Vaswani et al. (2017).

$$h_{LA} = softmax(\frac{QK}{\sqrt{d}})V \quad (8)$$

where h_{LA} denotes the output of attention layer. Then we concatenate h_{LA} with the aggregated hidden states of [CLS] as follows:

$$logit_p = D([H^{cls}_{agg} : h_{LA}]) \quad (9)$$

where D is the Linear layer for binary classification.

3.4 Transfer Learning

We also implemented multi-task learning, joint learning and transfer learning between subtask 1 and 2. Only the transfer learning has improved the performance of subtask 2. First we trained a subtask 1 model with prompt training, then we use the subtask 1 model as the initial checkpoint, trained a subtask 2 model with MPrompt. Experiments in section 5.1 proved the effectiveness of this approach. Unfortunately, transfer learning from subtask 2 to subtask 1 or other approach has no improvement.

3.5 Other Tricks

The DEBERTA which has large amount of parameters tends to over-fit on small training dataset. We utilize RecAdam (Chen et al., 2020) to fine-tune the pre-trained model to address the over-fitting problem. RecAdam optimizer is proposed to address the catastrophic forgetting problem of sequential transfer learning paradigm by introducing a recall and learning mechanism into Adam optimizer, which maintain the learned knowledge in pre-trained model while learning a new task.

The numbers of each category is very unbalanced in train dataset, we over-sample the positive samples to alleviate this problem. On subtask 2, we keep the proportion of each positive category unchanged during oversampling.

Data augmentation is not applied in our approach because we haven't find other proper datasets.

4 Experimental setup

4.1 Data

We use the official released dataset of SemEval2022 Task4 for experiments. The dataset contains 8375/2094/3832 samples for train, dev and test data. The subtask 1 and subtask 2 share the same input paragraphs, and has different labels respectively. The maximum, mean length of training data is 1005 and 55.28 in words perspective, and 90% of training data are shorter than 95 words. The ratio of positive samples is only 9.48% in subtask 1, even much smaller in subtask 2, since some of PCL category such as "The poorer, the merrier" is quiet few. The shortage of positive samples makes the model more difficult to distinct the PCL descriptions from negative samples, and is hard to train a model with good generalization.

4.2 Parameter settings

Our implementation is based on the Pytorch framework for transformer-based models Wolf et al. (2020). We trained our model based on the pre-trained DEBERTA-v3-large model. We use Adam/RecAdam optimizer with a learning rate of 3e-5, batch-size of 32 to train our models. The max sequence length is 256 and the epoch of training is set to 10 in subtask 1, 3 in subtask 2. To address the over-fitting problem, we apply RecAdam with sigmoid annealing function, where the annealing rate is 0.01 and the annealing time-step is 500. Specially, in subtask 2, we apply one self-attention layer as the aggregator and one attention layer to calculate the label attention. We pick the best checkpoint based on the performance on the dev set. Besides using DEBERTA, we also trained models based on ROBERTA, which is competitive with DEBERTA on subtask 2.

Since only 2 submissions are permitted in submitting phase, we trained multiple models under different settings for model ensemble. We also adopt 7-fold cross-validation training to improve the system generalization.

4.3 Ensemble

Two strategies are used for our final submissions on test data: 1) we ensemble all 7 models from 7-fold cross-validation training by averaging their outputs, which is trained on the train data of each subtask; 2) we trained multiple models on the train data with different model structures. Eight top different models are selected based on the dev accuracy for models ensemble, then average their outputs as the final output.

5 Results and Analysis

5.1 Single Model Performance

In subtask 1, the F1-score of PCL is used as the official metric and the results of dev set is shown in Table 2. We implement a baseline model for comparison, which is the traditional classification model with pre-trained LMs by using the [CLS] embedding. We trained two traditional classification models which are based on ROBERTA and DEBERTA. The results shown that DEBERTA is better than ROBERTA for subtask 1. The prompt classification improved the F1-scores by 2.80% on the same pre-trained LM, which proved that prompt classification is more suitable with the pre-trained LMs.

In subtask 2, the official metric is the average score of F1-scores of all PCL categories. As shown in Table 1, our method achieves significant improvement compared with baselines. The ROBERTA and DEBERTA denote the baseline models which uses a multi binary classification head upon pre-train LMs. The MPrompt both improved the performance on ROBERTA and DEBERTA by 8.28%, 20.29%, respectively. The information of label names is utilized by MPrompt method effectively.

For Transfer Learning(TL), as shown in Table 1, the scores is improved by 3.62% based on DEBERTA, while is dropped by 3.17% based on ROBERTA. We assume that is because the structure of ROBERTA in subtask 1 is not suitable for MPrompt in subtask 2.

Label Attention(LA) is also proved to be an effective approach both on ROBERTA and DEBERTA. Since the transfer learning is not useful in ROBERTA, we further experiment the ROBERTA MPrompt with LA and the DEBERTA MPrompt

Method	Unb	Sha	Pre	Aut	Met	Com	Poo	avg-F1
ROBERTA	59.28	47.05	25.00	30.76	42.85	49.73	42.85	42.50
ROBERTA MPrompt	58.95	38.09	46.15	35.00	48.71	55.23	39.99	46.02
+TL	61.53	36.36	40.00	34.61	45.83	51.74	42.10	44.56
+LA	61.42	47.05	45.87	37.38	43.03	54.63	39.99	47.05
DEBERTA	55.24	29.16	34.28	25.28	45.45	49.26	19.99	36.95
DEBERTA MPrompt	57.14	40.54	32.81	30.76	44.44	52.77	52.63	44.45
+TL	57.65	29.63	43.10	39.34	60.67	47.61	44.44	46.06
+TL+LA	60.25	30.55	43.69	45.97	53.48	49.49	47.61	**47.29**

Table 1: Single model performance of subtask 2 on dev dataset, Unb...Poo denotes the F1-scores of each categories, avg-F1 is the average F1-score. TL denotes the Transfer Learning, and LA denotes the Label Attention.

Method	Acc	Recall	F1
ROBERTA	60.56	64.82	62.62
DEBERTA	66.67	61.31	63.87
DEBERTA Prompt	65.50	65.82	**65.66**

Table 2: Single model performance of subtask 1 on dev dataset.

	Method	Subtask1	Subtask2
Dev set	7-fold ensemble	-	-
	top ensemble	72.16	52.99
Test set	7-fold ensemble	**62.73**	**43.87**
	top ensemble	58.93	43.20

Table 3: Ensemble performance on dev and test dataset, where 7-fold is the models from 7-fold cross-validation training, top ensemble means that ensembles the models with top dev accuracy.

with TL and LA. The results shown that LA can improve the performance by 2.23% on ROBERTA, and 2.67% on DEBERTA, which proved that an external attention between tokens and label names is benefit for picking more important tokens related to the label category.

5.2 Ensemble Performance

The performances of ensemble models are shown on Table 3, which is obtained from the competition leader-board. Our system got the second place in subtask 2 and the ninth place in subtask 1. Ensemble results on dev dataset are exhibited for comparison. Since the 7-fold training has trained the dev set, we don't exhibit the dev ensemble results of 7-fold training. It is obvious that the top ensemble method is much over-fit on dev set, for the scores of test set dropped much on top ensemble method. 7-fold ensemble method is an effective way to avoid over-fitting and got the best test scores of our system.

In top ensemble, we also found that integrating different pre-trained models improves the results significantly. The top ensemble method will be more useful if the distribution of dev set and test set are similar.

6 Conclusion

In this paper, we propose a multi-prompt training with label attention mechanism to improve the performance of multi-label classification task. Pre-trained models have made great performance gain compared to traditional neural network models in many natural language tasks. The above experimental results may suggest that the current pre-trained model mechanism still has room for improvement in ABSA tasks.

References

Jiahao Bu, Lei Ren, Shuang Zheng, Yang Yang, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. ASAP: A chinese review dataset towards aspect category sentiment analysis and rating prediction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2069–2079.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7870–7881.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.*

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

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *CoRR*, abs/2111.09543.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A challenge dataset and effective models for aspect-based sentiment analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6279–6284.

Yohan Jo and Alice H. Oh. 2011. Aspect and sentiment unification model for online review analysis. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, pages 815–824.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *CoRR*, abs/2103.10385.

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

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 255–269.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 380–385.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Joachim Wagner, Piyush Arora, Santiago Cortes, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. 2014. DCU: aspect-based polarity classification for semeval task 4. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*, pages 223–229.

Bo Wang, Maria Liakata, Arkaitz Zubiaga, and Rob Procter. 2017. Tdparse: Multi-target-specific sentiment recognition on twitter. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 483–493.

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

BEIKE NLP at SemEval-2022 Task 4: Prompt-Based Paragraph Classification for Patronizing and Condescending Language Detection

Yong Deng, Chenxiao Dou[*], Liangyu Chen
Deqiang Miao, Xianghui Sun, Baochang Ma, Xiangang Li
AI Center@Beike, China
{dengyong013, douchenxiao001,chenliangyu003}@ke.com

Abstract

PCL detection task is aimed at identifying and categorizing language that is patronizing or condescending towards vulnerable communities in the general media. Compared to other NLP tasks of paragraph classification, the negative language presented in the PCL detection task is usually more implicit and subtle to be recognized, making the performance of common text-classification approaches disappointed. Targeting the PCL detection problem in SemEval-2022 Task 4, in this paper, we give an introduction to our team's solution, which exploits the power of prompt-based learning on paragraph classification. We reformulate the task as an appropriate cloze prompt and use pre-trained Masked Language Models to fill the cloze slot. For the two subtasks, binary classification and multi-label classification, DeBERTa model is adopted and fine-tuned to predict masked label words of task-specific prompts. On the evaluation dataset, for binary classification, our approach achieves an F1-score of 0.6406; for multi-label classification, our approach achieves an macro-F1-score of 0.4689 and ranks first in the leaderboard.

1 Introduction

Patronizing and Condescending Language (PCL) towards vulnerable communities in the media has become a hot issue, which is heatedly discussed in society at present. The language often refers to the discourse which is published with a mixture of pity and superiority towards unprivileged groups. Such attitude is always adopted unconsciously or even out of kindness, however, its negative effect indeed exacerbates social prejudice and routinizes discrimination towards disadvantaged people (Ng, 2007). Despite the PCL emerging in numerous media, its style can not be precisely captured by far, since the usage of PCL is commonly unintended, unaffected, and subjective. This poses many challenges to the detection of PCL, which attracts great attention from the NLP community. Even though there exist substantial NLP studies on paragraph identification in various fields, research specifically for PCL identification has not been yet seriously introduced (Pérez-Almendros et al., 2020). Nowadays with social discrimination continuously rising, an effective approach, which can automatically identify PCL towards vulnerable communities, becomes more and more necessary and important to our society.

To encourage more research on the PCL problem, SemEval-2022 Task 4 (Pérez-Almendros et al., 2022) provides an English PCL dataset for language-modeling study and evaluation. The main task contains two PCL-classification subtasks, one for binary classification (SubTask 1) and the other for multi-label classification (SubTask 2). Given a paragraph, SubTask 1 is aimed to identify whether or not it is written with PCL style. F1 over the positive class is taken as the evaluation metric of this task. For the same paragraph, the goal of SubTask 2 is to classify which specific PCL categories it belongs to. In detail, there are 7 different PCL types needed to be recognized, and one paragraph can have up to 7 PCL labels at the same time. For evaluation, SubTask 2 introduces macro F1 over the 7 classes as the metric.

In this paper, we propose our approach targeting PCL detection, which is submitted to SemEval-2022 Task 4. To tackle the sparsity and implicity issues of PCL language, we first reformulate the task as a specific form of cloze prompt, and then apply prompt-based learning on it to predict the appropriate label words and corresponding task labels. To take advantage of the pre-trained Language Model (LM), DeBERTa (He et al., 2020) is adopted as our based model, which improves BERT (Devlin et al., 2018) by introducing two novel techniques: disentangled attention and decoding enhanced masking.

[*] Corresponding Author

The same classification method and model architecture are used for both subtasks to train and predict. Experiments are conducted to show the effectiveness of our approach. At last, during the evaluation phase, our approach achieves an F1 score of 0.6406 on SubTask 1 (rank 4th on the leaderboard), and a macro-F1 score of 0.4689 on SubTask 2 (rank 1st on the leaderboard).

2 Related Work

Condescending and patronizing treatment is a controversial social problem, which attracts attention from researchers of various fields (Margić, 2017; Huckin, 2002). To the NLP community, although extensive work on detecting different kinds of harmful language is presented, the targeted language styles are often explicit, aggressive, and flagrant, which can be perceived obviously. Unlike the previously studied language, the style of PCL language can not be easily sensed, since the usage of PCL is commonly unintended, unaffected, and subjective, which makes it a challenging identification problem. In recent years, more and more researchers (Mendelsohn et al., 2020; Sap et al., 2019) start researching this emergent NLP topic, but the room for progress is still large.

Paragraph classification is a basic and important NLP task, which has been continuously studied by industry and academia. Applications based on paragraph classification are increasingly permeating our lives, such as spam filtering (Kumar, 2020) and text sentiment analysis (Gao et al., 2019). Traditional classification models, such as SVM (Joachims, 1998) and XGBoost (Chen and Guestrin, 2016), predict the category of paragraph mainly based on the statistical features extracted from raw text data. With the appearance of BERT (Devlin et al., 2018), many researchers (Croce et al., 2020; Jin et al., 2020) achieve success on paragraph classification, by utilizing the contextualized word vectors of pre-trained language models.

In recent years, with the development of prompt-based learning (Schick and Schütze, 2020b,a; Brown et al., 2020), researchers start reformulating the paragraph classification problem as a masked language modeling problem. Unlike the traditional models, prompt-based models first predict the label word of mask in a prompt and then map the label word to the label of category by a verbalizer. Thanks to the power of pre-trained language models, prompt-based models (Hu et al., 2021; Gu et al., 2021) demonstrate their strength in performing few-shot or even zero-shot learning on the scenarios of paragraph classification with few or no labeled data. The above studies have shown that prompt-based learning is highly effective to solve the problem of paragraph classification. To study the effect of prompt-based learning on identifying paragraphs of PCL, in the rest of this paper, our modeling method and experimental analysis are introduced.

3 Our Approach

In this section, we present our approach to utilize prompt-based learning for PCL detection. We first give the overall paradigm of our model and then introduce the ensemble strategy used in this task.

3.1 Prompt-Based Model

Unlike other traditional classification methods which take original text as input, the prompt-based classification method first wraps the original text with some specific-designed template and then feeds the synthesized text into a language model for classification. In this way, the paragraph classification problem is reformulated into a masked language modeling problem. Formally, let $x = (x_0, x_1, ..., x_n)$ be a raw paragraph with a length of n from the PCL dataset, T be a prompt template and x^T be the reformulated paragraph of x using T. For an example of SubTask 1 shown in Fig 1, with x = *"People ordered pizzas to be delivered , with the ample leftovers donated to local homeless shelters."* and T = *"Is it patronizing or condescending? [MASK]"*, we wrap them into x^T as *"People ordered pizzas to be delivered , with the ample leftovers donated to local homeless shelters. Is it patronizing or condescending? [MASK]"*.

After the PCL problem is reformulated, an LM model M can be used to compute the probability of each word v in a vocabulary list to appear in the position of [MASK], denoted as $P([\text{MASK}] = v|x^T)$. To answer the cloze question *"Is it patronizing or condescending? [MASK]"*, if M has a large probability to fill [MASK] with some words of the meaning related to *"YES"*, then x is predicted as PCL class; if words having similar semantics to *"NO"* are more likely to be filled, then x is deemed as non-PCL class. In order to map the probabilities of predicted words to the probabilities of the labels *"YES"* and *"NO"*, we define a verbalizer to map a word v, from the label word set V_y, to a label y, form the label set Y. To construct the verbalizer,

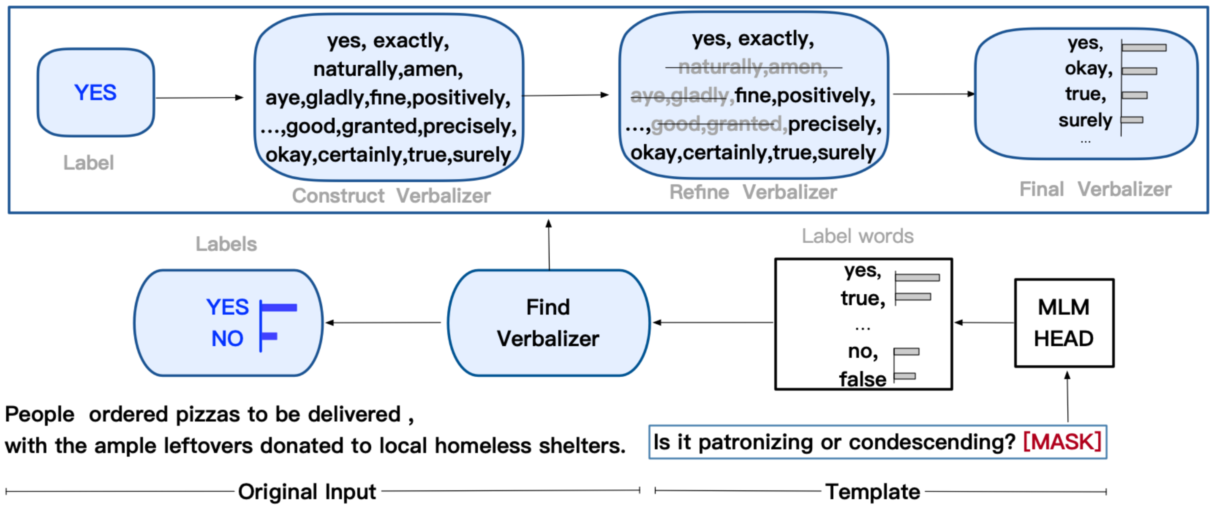

YES
Label
yes, exactly, naturally,amen, aye,gladly,fine,positively, ...,good,granted,precisely, okay,certainly,true,surely
Construct Verbalizer
yes, exactly, ~~naturally,amen,~~ ~~aye,gladly,~~fine,positively, ...,~~good,granted,~~precisely, okay,certainly,true,surely
Refine Verbalizer
yes, okay, true, surely ...
Final Verbalizer
Labels
YES
NO
Find Verbalizer
Label words
yes, true, ... no, false
MLM HEAD
People ordered pizzas to be delivered , with the ample leftovers donated to local homeless shelters.
Is it patronizing or condescending? [MASK]
Original Input
Template

Figure 1: An example to illustrate our approach.

we first use synonym dictionaries to find the synonyms of labels as the candidate label words, and then we select the top-k candidate words, according to their usage frequency in public corpora, as the final label words. Assuming that all label words of the same label contribute equally to label prediction and the prior distribution of them is uniform, we use the mean of their predicted probabilities as the output probability of their corresponding label. Then, given a reformulated paragraph x^T, its predicted label $\hat{y}$ can be obtained by

$$\hat{y} = \arg\max_{y \in Y}\left(\frac{1}{|V_y|}\sum_{v \in V_y} P([\text{MASK}] = v|x^T)\right)$$

With the above definitions, in the training process, a cross-entropy loss is applied to fine-tune the model M as our output model.

In SemEval-2022 Task 4, for the two subtasks, we use the same method to model and solve them. The only difference is the template used for each subtask. For the binary-classification subtask, only one template is provided to identify whether or not a paragraph is PCL class. For the multi-label-classification subtask, as there exist 7 PCL categories, each of them is given with a unique template for type detection.

3.2 Ensemble Strategy

To improve the robustness of the proposed approach, we take advantage of the cross-validation to make our model more stable and reliable. We randomly split the original PCL dataset into 10 parts, each of which has 1/10 samples of the dataset. According to the hold-one-out strategy, 10 folds of the data can be acquired, and each fold has 9/10 samples for training and 1/10 samples for validation. During the training process, the best-trained model for each fold is saved, and the average output probability of all models is taken as the final prediction score. According to the score, each instance is assigned with a predicted label, and then the predicted labels are compared with the golden labels to compute the F1 score for validation. The best models acquired in the validation process are kept for the final online evaluation.

4 Experiment

4.1 Settings

The PCL dataset (Pérez-Almendros et al., 2020) containing 10,469 paragraphs is shared by both SubTask 1 and SubTask 2. The goal of SubTask 1 is to detect 993 PCL samples from the whole dataset, and the goal of SubTask 2 is to classify the 993 PCL samples into 7 specific PCL types. In our experiments, all trials are performed with Nvidia Tesla A100 and large-DeBERTa is used as the based pre-trained language model. Considering the small size of the task data, we take AdamW as the optimizer in the experiments with a learning rate of 1e-5 and a maximum epoch number of 10. Early-Stop strategy is also used in our training process. For other parameters, we set the batch size as 16 and the maximum sequence length as 256.

4.2 Used Strategies

For the sake of result analysis, we list all strategies used in our approach towards SemEval-2022 Task

Strategy	SubTask 1			SubTask 2		
	R	P	F1	macro-R	macro-P	macro-F1
CLS	61.7%	61.0%	61.4%	41.7%	41.1%	41.3%
Prompt	62.2%	61.8%	61.9%	44.3%	44.7%	44.4%
Prompt + Ensemble	63.0%	61.6%	62.3%	46.6%	46.3%	46.4%
Prompt + Ensemble + R-Drop	63.1%	62.0%	62.5%	46.7%	46.4%	46.5%
Prompt + Ensemble + R-Drop + EDA	63.3%	63.0%	63.1%	46.6%	48.2%	47.5%

Table 1: Experimental Results on SubTask 1 and SubTask 2

4 as the following.

- **Prompt:** Prompt strategy is the method described in Section 3.1.
- **Ensemble:** Ensemble strategy is the method described in Section 3.2.
- **CLS:** We directly feed the original text into DeBERTa model and apply a softmax layer on the CLS token for classification. This method is used as a comparison strategy, which does not reformulate the classification problem.
- **EDA:** (Wei and Zou, 2019) is a method of data augmentation, which introduces 4 operations, synonym replacement, random insertion, random swap, and random deletion, to boost performance on paragraph classification.
- **R-Drop:** (Wu et al., 2021) is a method to regularize dropout, which minimizes the bidirectional KL-divergence between the distributions of two sub-models sampled by dropout, to reduce model randomness.

4.3 Results and Analysis

Table 1 shows all strategy results with Precision, Recall, and F1. From the results, the effect of each strategy we used can be clearly observed. The last method including all Prompt, Ensemble, EDA and R-Drop performs best compared to other settings.

To show the superiority of the Prompt paradigm, we first compare it with CLS, which uses the traditional paradigm of classification. From the table, it can be seen that Prompt leads CLS by about 0.5% F1 score in SubTask 1 and 3.1% macro-F1 score in SubTask 2. As Prompt has natural advantages on few-shot learning and the size of the PCL dataset is small, such result is not out of expectation. With the limited training data, to avoid the occurrence of overfitting and improve the robustness of our model, we then apply Ensemble strategy to the learning process. As shown in the table, with the robustness improved, a 0.4% increase of F1 score is achieved in SubTask 1. And for SubTask 2, the improvement is larger with a 2% increase of macro-F1 score, since the data size of each category in SubTask 2 is far less than 1/10 of that in SubTask 1. To further enhance model robustness, R-Drop is also adopted in our approach, which aims to reduce the randomness generated by the dropout module of network. Though the performance improvement of R-Drop is not big, the stability and convergence of our approach increases, making the results of repeated trials more similar.

Besides the techniques of modeling, we also try to improve the learning by increasing the quantity of the training data. As described in the previous sections, the size of the training data is small and PCL language is often written in an implicit style. Therefore, EDA strategy is performed to augment the PCL data, which replaces implicit words with their synonyms, and transforms the structure of paragraphs variously, to produce more data for training. With the augmented data, our approach achieves a 0.6% lift of F1 score on SubTask 1 and a 1% lift of macro-F1 score on SubTask 2.

5 Conclusion

In this paper, we propose a prompt-based learning approach for PCL detection, based on pre-trained language models. To reformulate the PCL detection problem into a masked language modeling problem, how our prompt template is designed have been fully discussed. We also introduce several other techniques, which have a positive impact on prompt-based learning. Through experimental comparison, our approach is proven as an effective solution to detect PCL language. As a result, in SemEval-2022 Task 4, our approach ranks 4th on the leaderboard of SubTask 1, and ranks 1st on the leaderboard of SubTask 2.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. Gan-bert: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 2114–2119.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhengjie Gao, Ao Feng, Xinyu Song, and Xi Wu. 2019. Target-dependent sentiment classification with bert. *Ieee Access*, 7:154290–154299.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juanzi Li, and Maosong Sun. 2021. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *arXiv preprint arXiv:2108.02035*.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

Pradeep Kumar. 2020. Predictive analytics for spam email classification using machine learning techniques. *International Journal of Computer Applications in Technology*, 64(3):282–296.

Branka Drljača Margić. 2017. Communication courtesy or condescension? linguistic accommodation of native to non-native speakers of english. *Journal of English as a lingua franca*, 6(1):29–55.

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A framework for the computational linguistic analysis of dehumanization. *Frontiers in artificial intelligence*, 3:55.

Sik Hung Ng. 2007. Language-based discrimination: Blatant and subtle forms. *Journal of Language and Social Psychology*, 26(2):106–122.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2019. Social bias frames: Reasoning about social and power implications of language. *arXiv preprint arXiv:1911.03891*.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

DH-FBK at SemEval-2022 Task 4: Leveraging Annotators' Disagreement and Multiple Data Views for Patronizing Language Detection

Alan Ramponi
Fondazione Bruno Kessler
Trento, Italy
alramponi@fbk.eu

Elisa Leonardelli
Fondazione Bruno Kessler
Trento, Italy
eleonardelli@fbk.eu

Abstract

The subtle and typically unconscious use of patronizing and condescending language (PCL) in large-audience media outlets undesirably feeds stereotypes and strengthens power-knowledge relationships, perpetuating discrimination towards vulnerable communities. Due to its subjective and subtle nature, PCL detection is an open and challenging problem, both for computational methods and human annotators. In this paper we describe the systems submitted by the DH-FBK team to SemEval-2022 Task 4, aiming at detecting PCL towards vulnerable communities in English media texts. Motivated by the subjectivity of human interpretation, we propose to leverage annotators' uncertainty and disagreement to better capture the shades of PCL in a multi-task, multi-view learning framework. Our approach achieves competitive results, largely outperforming baselines and ranking on the top-left side of the leaderboard on both PCL identification and classification. Noticeably, our approach does not rely on any external data or model ensemble, making it a viable and attractive solution for real-world use.

1 Introduction

Detecting patronizing and condescending language (PCL) is an open, challenging, and underexplored research area in natural language processing (Pérez-Almendros et al., 2020; Wang and Potts, 2019). A patronizing and condescending attitude is expressed as a good-natured and beneficial attitude from a person of authority towards others, who are typically depicted in a subtly compassionate way (Pérez-Almendros et al., 2020).

PCL is a mildly perceived phenomenon. It is often unconscious, driven by good intentions, and expressed through *flowery wordings* (Wong et al., 2014; Huckin, 2002). This makes PCL identification and classification difficult both for NLP systems and human annotators (cf. Figure 1), as it cannot be linked to specific words. Nonetheless,

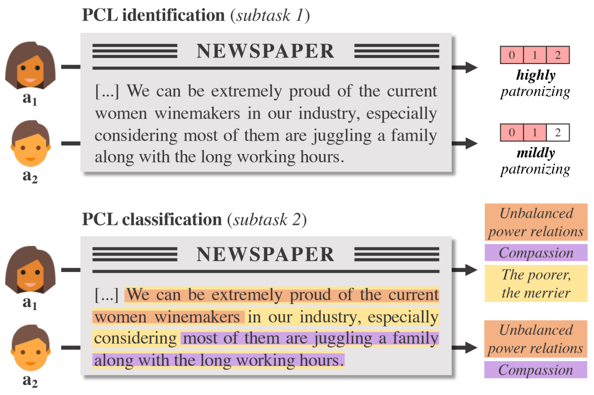

PCL identification (subtask 1)
NEWSPAPER
[...] We can be extremely proud of the current women winemakers in our industry, especially considering most of them are juggling a family along with the long working hours.
a1
a2
0 1 2
highly patronizing
0 1 2
mildly patronizing
PCL classification (subtask 2)
NEWSPAPER
[...] We can be extremely proud of the current women winemakers in our industry, especially considering most of them are juggling a family along with the long working hours.
Unbalanced power relations
Compassion
The poorer, the merrier
Unbalanced power relations
Compassion

Figure 1: Example showing that patronizing and condescending language is a subtle linguistic phenomenon that human annotators (a_1, a_2) often perceive differently, and thus annotate in different ways.

it undesirably conveys harmful messages in many ways, as it promotes stereotypes, and a superiority and discriminatory mindset (Fiske, 1993). This is particularly damaging when used by large-audience media outlets, since it drives greater exclusion of already vulnerable communities (Pérez-Almendros et al., 2020). Automatically detecting PCL has the potential to enable a range of applications and research directions, such as suggestion tools for news editors to mitigate condescension in writing before publication, and studies on the interplay between condescension and sociodemographic factors.

To encourage research on patronizing and condescending language detection, the SemEval-2022 Task 4 (Pérez-Almendros et al., 2022) has recently been proposed. The shared task aims at investigating methods for the identification of PCL (subtask 1; Figure 1, top) and the categorization of the linguistic techniques which are used to express it (subtask 2; Figure 1, bottom) on English news stories mentioning vulnerable communities (Section 2).

In this paper, we present the **DH-FBK** entry for the SemEval-2022 Task 4 (Pérez-Almendros et al., 2022). Motivated by the subtle nature of

Example	Category
“We can be extremely proud of the current women winemakers”	Unbalanced power relations
“The inclusion of a refugee team”	Shallow solution
“An immigrant to a developed country lives in two worlds”	Presupposition
“women must wake up”	Authority voice
“trapped in the prison of poverty”	Metaphor
“more than 400 suspected asylum seekers are awaiting their fate”	Compassion
“how talented disabled people can be”	The poorer, the merrier

Table 1: Examples of text excerpts expressing patronizing and condescending language, along with their category.

patronizing language and the subjectivity of human interpretation, we propose a multi-task, multi-view learning approach which leverages annotators' uncertainty and disagreement as auxiliary tasks (Section 3) in judging for PCL presence (subtask 1) or category (subtask 2). Further, we investigate the effectiveness of sequentially fine-tuning on subtasks of increasing complexity (subtask 1 $\mapsto$ subtask 2), as well as the use of additional information such as the geographical provenance of news outlets.

Our systems achieve competitive results on the SemEval-2022 Task 4, outperforming the organizers' RoBERTa (Liu et al., 2019) baseline by a large margin (subtask 1: $+8.8$ F_1; subtask 2: $+26.9$ F_1) and consistently ranking on the top-left side of the leaderboard (subtask 1: 18th out of 78 teams; subtask 2: 13th out of 49 teams) without using any external data or ensemble strategy, making it a viable solution for real-world use. We make our code publicly available to the research community to encourage future work on this direction.[1]

2 Data and task description

In this section, we present relevant details on data and the associated shared task. We firstly summarize the dataset (Section 2.1) and describe the task setup (Section 2.2). Next, we focus on the data annotation process as it is central for understanding our methods (Section 2.3).

2.1 “Don't Patronize Me!” data

The organizers of SemEval-2022 Task 4 provide participant teams with the “Don't Patronize Me!” annotated dataset, originally introduced in Pérez-Almendros et al. (2020) and further updated for the purpose of the shared task (v1.4). The dataset comprises a selection of 10,469 paragraphs published in the news of 20 English-speaking countries[2] from all over the world between 2010 and 2018, and sampled from the “News on Web” corpus (NoW; Davies, 2013). Each paragraph mentions one of ten selected vulnerable communities (i.e., *disabled*, *homeless*, *hopeless*, *immigrant*, *in need*, *migrant*, *poor families*, *refugee*, *vulnerable*, and *women*). These communities have been chosen because they are often target of PCL. Notably, attention has been paid to balance paragraphs across communities and news outlets' countries. For further details, we refer to Pérez-Almendros et al. (2022, 2020).

2.2 Task setup

The SemEval-2022 Task 4 challenge is divided into the following two subtasks:

1. **PCL identification**: given an input paragraph, identify whether it entails any form of PCL. Formally, this is a binary classification task;

2. **PCL classification**: given an input paragraph, decide what linguistic techniques are used to express the condescension (if any). This is a multi-label classification task, with 7 possible labels – i.e., *unbalanced power relations* (UNB), *shallow solution* (SHA), *presupposition* (PRE), *authority voice* (AUT), *metaphor* (MET), *compassion* (COM), and *the poorer, the merrier* (THE). These categories follow a validated PCL taxonomy (Pérez-Almendros et al., 2020) that we summarize in Appendix A. Examples for each category are in Table 1.

As PCL is a subtle and mild phenomenon, the annotation process was not straightforward (Pérez-Almendros et al., 2020). In the following section, the annotation scheme followed by dataset creators

[1]The source code is available at `https://github.com/dhfbk/pcl-detection-disagreement`.

[2]Covered English-speaking country codes: au, bd, ca, gb, gh, hk, ie, in, jm, ke, lk, my, ng, nz, ph, pk, sg, tz, us, and za.

Annotation task	Individual decisions (a_1,a_2)	Score	Instances	Gold label
Subtask 1: *"Does the paragraph contain any form of PCL?"* Values: $0, 1, 2$	(0,0)	0	8,529	NO
	(0,1), (1,0), *	1	947	
	(1,1), *	2	144	YES
	(2,1), (1,2), *	3	458	
	(2,2)	4	391	
Subtask 2: *"Which PCL category does the span express (if any)?"* Values: $c_i, c_j \in C$, NONE	(c_i, NONE), (NONE, c_i)	1	1,359	c_i
	$(c_i, c_j)_{c_i \neq c_j}$, $(c_j, c_i)_{c_j \neq c_i}$			c_i, c_j
	(c_i, c_i)	2	1,401	c_i

Table 2: The annotation process from individual annotators' decisions to gold labels for both subtasks. For subtask 1, two annotators (a_1, a_2) assigned value 0 (no PCL), 1 (mild PCL) or 2 (high PCL) to each instance of the dataset. The "Score" column indicates the sum of their decisions. In subtask 2, the annotators further characterized the PCL instances of subtask 1, by identifying exact text spans and determining categories of the PCL (if any). We generalize with $c_i, c_j \in C$ two of the $|C| = 7$ possible PCL categories which annotators could have chosen, to show the process in case of disagreement. The "Score" column indicates the number of annotators which agreed on the category. *Includes cases of total disagreement – i.e., (0,2) and (2,0) – resolved by a third annotator a_3.

is presented, as it is especially relevant for understanding the data itself and our methods.

2.3 Annotation process

The dataset has been manually labeled by expert annotators[3] following a two-step process as described below. The resulting annotations are the gold-standard reference for the subtasks (Section 2.2).

Subtask 1 The annotation was performed by three annotators. Two annotators labeled the whole dataset (a_1 and a_2), while a third one (a_3) intervened in case of clear disagreement between a_1 and a_2. Authors reported that "*this annotation step proved more difficult than expected, stemming from the often subtle and subjective nature of PCL*" (Pérez-Almendros et al., 2020). Because of this difficulty, annotators were given the possibility to assign each paragraph a value 0 (no PCL), 1 (borderline), or 2 (highly PCL). Information about the annotation is available as a 5-point scale, which reflects a joint notion of uncertainty and agreement between annotators. For subtask 1, organizers map values into a binary form (i.e., $\{0, 1\} \mapsto$ NO-PCL, $\{2, 3, 4\} \mapsto$ PCL), evaluating systems accordingly. As anticipated, cases of total disagreement (i.e., a_1: 0 and a_2: 2, and viceversa) received a third independent annotation by a_3.[4] If a_3 considered the paragraph not to contain PCL, a borderline case, or an otherwise clear PCL case, the paragraph was assigned a value of 1, 2 or 3, respectively. This has the effect to leave extreme values (0 and 4) reserved for clear-cut cases. The number of PCL-expressing paragraphs is 993 (9.5%). A summary of the annotation process for subtask 1 is in Table 2, top.

Subtask 2 In the second round of annotation, paragraphs previously labeled as containing PCL were further characterized by a_1 and a_2. The aim is two-fold: (i) identify the paragraph segments (or spans) that express PCL, and (ii) categorize each of them into one or more PCL categories (cf. Section 2.2). As a consequence, each identified span exhibits one or multiple labels, depending on whether one or both annotators identified it, and on their agreement on the type(s) of condescension expressed by the text segment. This results in a per-span per-type agreement information on a 2-point scale (1 or 2). Organizers frame subtask 2 as a paragraph-level classification problem, and thus each paragraph can express zero or more condescension types based on the resulting 2,760 span annotations (2.8 annotations per paragraph, on average). An overview of the annotation process for subtask 2 is presented in Table 2, bottom.

3 Methods

Models proposed for PCL identification and classification are all based on multi-task learning (Caruana, 1997) and use multiple views of input data, inspired by Clark et al. (2018). In this section, we firstly introduce the general framework on which all our models are based on (Section 3.1).

[3] Dataset authors reported the annotators' background is on the fields of communication, media, and data science.

[4] According to the first dataset release, these account for 5.5% of the annotations (Pérez-Almendros et al., 2020).

Then, we provide details on data- and task-specific components that we use in our systems, namely dataset views (Section 3.2) and auxiliary tasks (Section 3.3). Lastly, we present the composition of our final models (Section 3.4).

3.1 General framework

Our approach is based on multi-task learning, a learning paradigm that aims to leverage training signals of related tasks at the same time by exploiting a shared representation in the model (Caruana, 1997). In all our models, we employ a *main* task, namely PCL identification or classification, which is a task of direct interest. Additionally, we employ *auxiliary* tasks (see Section 3.3), namely tasks which can provide useful signals to potentially improve the performance on the main task.

All our models use RoBERTa-base (Liu et al., 2019) as shared encoder, and a separate decoder for each task. This way, all tasks benefit from mutual signals encoded by a shared contextualized representation that is jointly fine-tuned during training.

The input is a text instance that is encoded using byte-pair encoding (BPE; Sennrich et al., 2016), whereas the output label is given by task-specific decoders which consist of a linear classification layer and operate on the contextual embedding of the special `[CLS]` classification token.

In our models, each auxiliary task makes use of a specific form (or view) of the original dataset, which we introduce in the following (Section 3.2). When training with multiple views of data, each input batch to the model consists of examples from a single data view, and the loss function is only activated for tasks associated to that data view. For further details on multi-view (or multi-dataset) training, refer to van der Goot et al. (2021). An overview of the framework is presented in Figure 2.

3.2 Data views

Our models employ different forms (or views) of the original "Don't Patronize Me!" dataset provided by organizers. Specifically, we use (i) paragraph and (ii) span views as detailed as follows.

Paragraph data view (D_P) This corresponds to the dataset in its standard form – i.e., whole paragraphs – as provided by organizers (Section 2.1).

Span data view (D_S) A dataset consisting of all text excerpts – i.e., paragraph substrings – that have been marked as expressing patronizing and condescending language. As a result, this dataset represents a different view of D_P data, where only snippets of PCL are included. Examples of text instances in D_S are reported in Table 1.

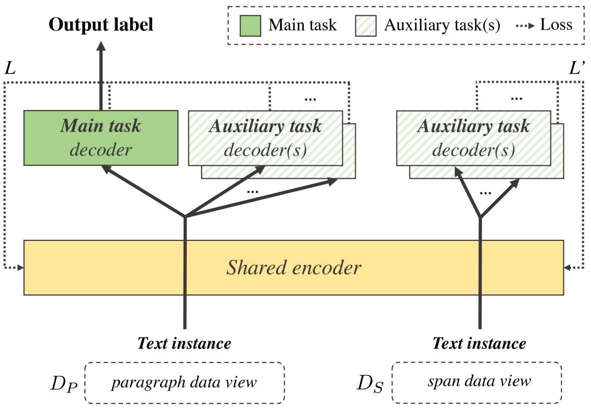


Figure 2: A high-level overview of our multi-task, multi-view learning framework.

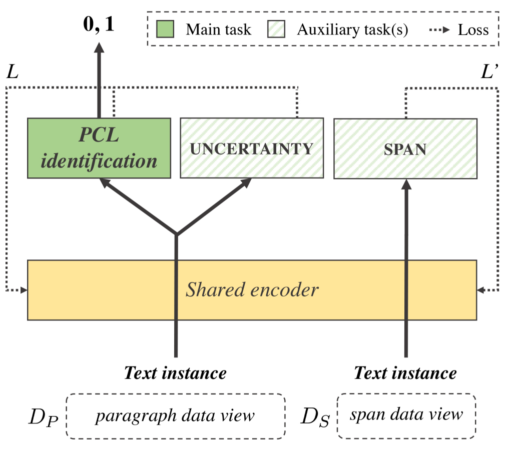
These different data views are used by specialized task decoders, as presented in Section 3.3.

3.3 Auxiliary tasks

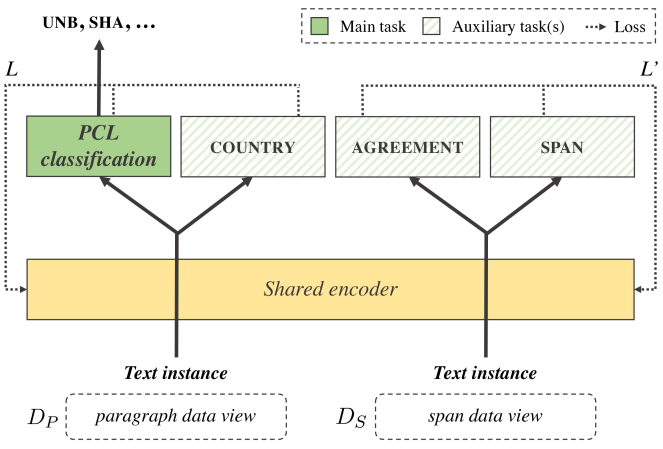
In this section, we describe the auxiliary tasks we used in one or more of our final models (Section 3.4), along with the data view each task uses. For details on the interplay between main and auxiliary tasks, we refer the reader to Section 4.1.

Paragraph uncertainty level (UNCERTAINTY) This task is used for subtask 1 to consider different annotators' point of view in identifying PCL. We use the aggregated 5-point scale score (cf. Section 2.3, subtask 1) assigned to each paragraph as auxiliary task. Although disaggregated annotators' decisions have not been made available for the shared task, we argue that the combined annotation provided by organizers can be viewed as a joint notion of uncertainty and agreement between annotators in identifying PCL, and is thus valuable information that can inform PCL identification. Since this is a paragraph-level information, this auxiliary task uses the D_P data view. For each paragraph, a label $l \in \{0, 1, 2, 3, 4\}$ must be predicted.

Span agreement level (AGREEMENT) This task is used for subtask 2 as it potentially drives useful signals for PCL classification. We hypothesize that the number of annotators which agree on a particular PCL category for a span (cf. Section 2.3, subtask 2) is a crucial information as it can provide the main task with different shades of PCL based on annotators' interpretation and sensibility. The



(a) MTMW(UNC+SPAN) model for subtask 1.



(b) MTMW(AGR+COU+SPAN) model for subtask 2.

Figure 3: Our selected models for PCL identification (subtask 1) and classification (subtask 2).

agreement level is a span-level information, and thus we use the D_S view for the associated task. For each span, a label $l \in \{1, 2\}$ must be predicted.

Span categorization (SPAN) This auxiliary task is used for both subtask 1 and 2, as we argue that making use of small and focused units of information such as condescending text excerpts would be useful to inject knowledge in the encoder about which paragraph segments are important for recognizing PCL. The task uses the D_S data view, and for each span a label $l \in \{$UNB, SHA, PRE, AUT, MET, COM, THE$\}$ has to be predicted (i.e., one among the condescending types in Section 2.2).

News outlet country (COUNTRY) We employ this auxiliary task for our subtask 2 model.[5] We hypothesize that the diverse background culture of countries results in variation in language use, and thus could have an impact on the expression of patronizing and condescending language. The news outlet provenance is provided by organizers along with the original dataset. The auxiliary task uses the D_P view, and the label to be predicted is a country code, i.e., $l \in \{$au, bd, ca, gb, gh, hk, ie, in, jm, ke, lk, my, ng, nz, ph, pk, sg, tz, us, za$\}$.

3.4 Models

Our approaches to PCL identification and classification are all centered on the hypothesis that leveraging annotators' uncertainty and disagreement during training is beneficial for capturing the subtle language which characterizes PCL. This is in line with recent work emphasizing the importance of modeling annotators' disagreement in subjective tasks (Davani et al., 2022; Leonardelli et al., 2021; Uma et al., 2021) and initiatives supporting the release of disaggregated annotations in NLP (Abercrombie et al., 2022).

We participated in the SemEval-2022 Task 4 challenge with two submissions for both subtasks. To this end, we built three models. Two models are our best systems – as evaluated on the development set (cf. Section 4.3) – on subtask 1 (Section 3.4.1) and subtask 2 (Section 3.4.2). We submit them for the corresponding subtasks. The third model was instead built aiming at a generic and unified solution, and represents our second submission for both subtasks (Section 3.4.3).

3.4.1 MTMW(UNC+SPAN) model for PCL identification (subtask 1)

In this multi-task, multi-view model (MTMW), we consider PCL identification as the main (binary classification) task, and employ UNCERTAINTY and SPAN as auxiliary tasks, as described in Section 3.3. We refer to this model to as MTMW(UNC+SPAN). An overview of the resulting approach for PCL identification is presented in Figure 3a.

3.4.2 MTMW(AGR+COU+SPAN) model for PCL classification (subtask 2)

In this multi-task, multi-view model (MTMW), the main task is PCL classification, whereas AGREEMENT, COUNTRY and SPAN are auxiliary tasks. For PCL classification, each label is modeled through a dedicated binary classification decoder. We use MTMW(AGR+COU+SPAN) to refer to this approach in the remainder of this paper. We graphically

[5]We have experimented with this auxiliary task on subtask 1 too; however, we noticed a substantial performance degradation compared to using uncertainty only (cf. Section 4.3).

present the model in Figure 3b, and refer the reader to Section 3.3 for details on auxiliary components.

3.4.3 Sequential PCL identification and classification (subtasks 1 & 2)

Given that PCL classification is a fine-grained version of the PCL identification task, we argue that sequentially fine-tuning encoder weights on tasks of increased complexity should be beneficial to improve the performance on both subtasks. This approach borrows the idea from modern data-centric adaptation methods in NLP (Ramponi and Plank, 2020) such as continued pretraining of language models (Gururangan et al., 2020), however employing it at the fine-tuning stage, similarly to intermediate-task transfer (Phang et al., 2018).

We firstly run the PCL identification model described in Section 3.4.1. Then, we use the resulting fine-tuned encoder weights as initialization for the encoder of the PCL classification model (Section 3.4.2) with SPAN auxiliary only.[6] Finally, we fine-tune it on subtask 2. This results in a single model that has incrementally learnt the complexity of PCL detection as a whole. Prediction of labels for subtask 1 were done simply considering a paragraph as containing PCL if it exhibits at least a PCL category label in subtask 2. We refer to this approach to as SEQ. FINE-TUNING model.

4 Experiments

In this section, we first outline the experimental setup (Section 4.1). Then, we present the results of our models (Section 4.2), as well as additional analyses and discussion (Section 4.3).

4.1 Experimental Setup

We implemented our models using the MaChAmp v0.2 toolkit (van der Goot et al., 2021) and employ RoBERTa-base (Liu et al., 2019) as shared encoder since it has been shown to outperform other commonly used pretrained language models on PCL detection tasks (Pérez-Almendros et al., 2020).

For training, we use default hyperparameters (Appendix B) and fine-tune each model – roughly 110M trainable parameters – for 10 epochs on a single GPU.[7] We use a cross-entropy loss with balanced class weights to give equal importance to all classes during fine-tuning, and thus emphasizing underrepresented classes in training data. The multi-task learning loss is computed as $L = \sum_t \lambda_t L_t$, where L_t is the loss for task t, and λ_t the corresponding weighting parameter. In our experiments, we empirically set $\lambda_t = 1$ for the main task, and $\lambda_t = 0.25$ for auxiliary tasks.[8]

We solely rely on data provided by organizers, and use the provided 80% train and 20% development split as one fold, additionally creating the remaining 80%/20% splits in a stratified fashion. To avoid confounding our results, we ensure training splits for the span data view do not contain any text excerpt appearing in development data of the paragraph data view. This results in 5 folds that we use for selecting models for our submissions. For official test set evaluation, we then submit the selected models trained on the provided data split.

For the purpose of the shared task, models for PCL identification (subtask 1) are evaluated using F_1 score over the positive class, whereas models for PCL classification (subtask 2) are evaluated based on macro-average F_1 score. We employ the same metrics at the model selection stage.

	P	R	F_1
Organizers' baseline	39.35	65.30	49.11
MTMW(UNC+SPAN)	64.23	52.68	**57.89**
SEQ. FINE-TUNING	53.99	55.52	54.74

Table 3: Official test set results of our models compared to the organizers' RoBERTa baseline on PCL identification (subtask 1). P: Precision; R: Recall; F_1: F_1 score over the positive class. Best results are in bold.

4.2 Results

We present the official results for our proposed models for subtask 1 and 2 in Table 3 and Table 4, respectively. We also include scores of the organizers' RoBERTa baseline for informed comparison based on the shared task metrics.

Subtask 1 As shown in Table 3, our submitted MTMW(UNC+SPAN) model outperforms the RoBERTa baseline by a large margin, with most of the benefit coming from the precision metric. This indicates that the uncertainty and agreement

[6]This choice is motivated by the need for a simpler and unified solution for both subtasks. We decided to leave country and agreement auxiliaries out for this submission due to negligible differences in performance on subtask 2 (Section 4.3).

[7]NVIDIA Tesla V100-SXM2.

[8]Except for AGREEMENT, for which we empirically set $\lambda_t = 0.125$ since it is actually *auxiliary of an auxiliary task*. A thorough investigation on the impact of the weighting parameter on individual auxiliary tasks is left for future work.

	UNB	SHA	PRE	AUT	MET	COM	THE	F_1
Organizers' baseline	35.35	0.00	16.67	0.00	0.00	20.87	0.00	10.41
MTMW(AGR+COU+SPAN)	52.46	36.22	26.95	**37.71**	**31.86**	**45.95**	**30.30**	**37.35**
SEQ. FINE-TUNING	**54.00**	**46.73**	**28.07**	22.22	29.73	44.28	20.69	35.10

Table 4: Official test set results of our models compared to the organizers' RoBERTa baseline on PCL classification (subtask 2). UNB: Unbalanced power relations; SHA: Shallow solution; PRE: Presupposition; AUT: Authority voice; MET: Metaphor; COM: Compassion; THE: The poorer, the merrier; F_1: macro-average F_1. Best results are in bold.

level as well as the use of focused text excerpts do play a positive role in PCL identification. On the other hand, while also the SEQ. FINE-TUNING approach provides better results compared to the baseline, it scores lower than MTMW(UNC+SPAN). A reason for this behavior can be attributed to the way we infer labels for this subtask (i.e., based on predictions for subtask 2, as anticipated in Section 3.4.3), or due to *catastrophic forgetting* (McCloskey and Cohen, 1989), a phenomenon in which prior knowledge is largely forgotten when learning a new task. On the shared task leaderboard, MTMW(UNC+SPAN) ranked 18th out of 78 teams.

Subtask 2 Similarly to results for subtask 1, both our submitted systems largely outperform the RoBERTa baseline, as shown in Table 4. We notice that the SEQ. FINE-TUNING approach still scores lower than a tailored approach for subtask 2, i.e., MTMW(AGR+COU+SPAN). Specifically, MTMW(AGR+COU+SPAN) shows an absolute improvement of +26.9 points in macro-average F_1 score over the RoBERTa baseline, with a clear advantage over the SEQ. FINE-TUNING model on underrepresented classes (i.e., AUT, MET and THE). Our MTMW(AGR+COU+SPAN) model ranked 13th out of 49 participating teams on the official leaderboard. Overall, our models do not require any external data or model ensemble, and we think this makes them viable approaches for real-world use.

4.3 Analysis and discussion

In order to provide insights for future work on PCL detection, we conduct analyses on the contribution of auxiliary tasks to performance of our models (Section 4.3.1), and an in-depth study on the role of uncertainty and disagreement (Section 4.3.2).

4.3.1 Contribution of auxiliary tasks

Our submitted models for PCL identification and classification leverage training signals coming from selected auxiliary tasks (cf. Section 3.4). These

	Model	F_1 score
subtask 1	Our single task baseline	$56.73_{\pm 3.2}$
	Multi-task setup	
	+ COUNTRY	$55.99_{\pm 2.7}$
	+ UNCERTAINTY	$56.92_{\pm 3.2}$
	+ COUNTRY, UNCERTAINTY	$57.74_{\pm 3.5}$
	Multi-task, multi-view setup	$55.69_{\pm 2.0}$
	+ COUNTRY	$57.35_{\pm 1.9}$
	+ UNCERTAINTY	$\mathbf{58.38}_{\pm 3.7}$
	+ COUNTRY, UNCERTAINTY	$57.53_{\pm 4.6}$
subtask 2	Our single task baseline	$37.02_{\pm 2.8}$
	Multi-task setup	
	+ COUNTRY	$36.26_{\pm 2.3}$
	Multi-task, multi-view setup	$38.25_{\pm 3.6}$
	+ COUNTRY	$37.16_{\pm 2.3}$
	+ AGREEMENT	$37.53_{\pm 0.8}$
	+ COUNTRY, AGREEMENT	$\mathbf{38.81}_{\pm 2.9}$

Table 5: Contribution of auxiliary tasks to performance of models for subtask 1 (top) and subtask 2 (bottom). We report mean and standard deviation of F_1 scores on development splits. The multi-task, multi-view setups use SPAN by default. Results for AGREEMENT and its combinations in the multi-task setup of subtask 2 are omitted, since they would require the D_S view, and thus only refer to the multi-task, multi-view configuration.

model variants have been chosen after a thorough performance evaluation on the development splits. In Table 5 we report the mean and standard deviation of F_1 scores for each model with different auxiliary task configurations. We denote with "Our single task baseline" our baseline RoBERTa model with no multi-task nor multi-view learning. Note that this is different from organizers' RoBERTa baseline, since we employ a different hyperparameter setup with the addition of class weights (Section 4.1). We do not include the organizers' baseline here due to incomparability –

we only have access to results on a single development split. For reference, organizers reported 48.29 F_1 on subtask 1, and 13.40 F_1 on subtask 2 for their baseline. Multi-task setups refer to configurations where D_P-based auxiliaries are used, whereas multi-task, multi-view setups also exploit the D_S view, thus using SPAN as default auxiliary task on all experiments.

Subtask 1 Table 5 (top) shows the results of models with different auxiliary task combinations on subtask 1 development data. All multi-task, and multi-task multi-view setups outperform the baseline, with the only exception of the multi-task configuration solely using COUNTRY as auxiliary task. Overall, the use of UNCERTAINTY as auxiliary task consistently improves the performance over the baseline, even when coupled with other auxiliaries. The best results are obtained when employing a multi-task, multi-view setup with UNCERTAINTY only (58.38 F_1), suggesting that information coming from the COUNTRY auxiliary is not as useful as in the multi-task scenario. We hypothesize this behaviour could be attributed to the use of SPAN in the multi-task, multi-view setup, which indirectly provides a more useful inductive bias for PCL identification. In future work we aim to further dig into this aspect, exploring various loss weight configurations to assess the strength of this finding.

Subtask 2 We present in Table 5 (bottom) the contribution of auxiliary tasks on subtask 2 development data. Similarly to subtask 1, COUNTRY in the multi-task setup is the only auxiliary task that does not improve the performance over our single task RoBERTa baseline. However, when coupled with AGREEMENT in the multi-task, multi-view configuration, it provides the best overall performance over all model variants (38.81 F_1). This suggests that the AGREEMENT auxiliary provides signals orthogonal to COUNTRY, as shown by the multi-task, multi-view alternatives employing these auxiliaries in isolation. By a closer look, the performance of the multi-task, multi-view setup alone (i.e., SPAN only) are highly competitive, confirming our hypothesis that using PCL-expressing text excerpts is beneficial for PCL detection as a whole.

4.3.2 Role of uncertainty and disagreement

To delve into the role of agreement and uncertainty, we further study performance of our best systems on subtask 1 and 2 as a function of the agreement/uncertainty level. To the goal, we use the predictions of our models on development splits.

level	0	1	2	3	4
F_1	49.27	44.67	27.32	33.39	41.95

Table 6: Performance of our model for subtask 1 as a function of different levels of uncertainty/disagreement. F_1 scores are averages over the five development splits.

Subtask 1 To investigate the impact of uncertainty/agreement on the performance of our model for subtask 1, we first divided each development split by uncertainty/agreement level (cf. "Score", Table 2, top). Then, we calculated the per-level F_1 score on each split. Finally, we averaged the per-level performance on all folds. We report the experimental results in Table 6. It is interesting to observe how in this task the uncertainty/agreement levels 0, 2 and 4 reflect agreement between annotators (0+0, 1+1, or 2+2), but performance for score 2 – where both annotators agree in being uncertain – is much worse. This suggests a prominent role of uncertainty in worsening classifier's performance, rather than disagreement. On the other hand, uncertainty and disagreement represent two sides of the same coin, as the less certain and clear a decision is, the greater probability is to have disagreement between annotators.

Subtask 2 We perform a similar analysis for subtask 2. The agreement level (cf. "Score", Table 2, bottom) has a clear effect on model's performance: out of the 2,760 PCL-expressing spans, only 44% of 1,359 spans with an agreement level of 1 are correctly labeled, compared to 56% of 1,401 spans for which both annotators signaled a form of PCL. Considering paragraphs with a single PCL-expressing span only, this results to 35% of 801 examples for agreement level 1, and 52% of 798 examples for agreement 2, further confirming that instances exhibiting disagreement are more difficult to classify.

5 Conclusion

In this paper, we presented our submitted systems to SemEval-2022 Task 4. We showed that leveraging annotators' uncertainty and disagreement during training in a multi-task, multi-view framework is beneficial for the identification and classification of patronizing and condescending language, and achieves competitive results on the official leaderboard without relying on any external data or model ensemble. We also showed that sequential fine-tuning is a viable alternative to tackle PCL identification and classification jointly, with the goal of reducing the use of computational resources during inference, although it obtains lower performance compared to our tailored solutions. A thorough analysis on the impact of diverse auxiliary tasks on the performance of our models for PCL detection, and an investigation on the role of uncertainty and disagreement further confirmed the importance of considering annotators' point of view in PCL detection. As future work, we aim to test the presence and assess the impact of spurious lexical biases in the dataset (Ramponi and Tonelli, 2022) and extend our models to other genres, such as social media (Wang and Potts, 2019). We hope this work will encourage future efforts towards annotators-centric NLP, on PCL detection and other subjective tasks more broadly.

Acknowledgements

Part of this work was funded by the PROTECTOR European project (ISFP-2020-AG-PROTECT-101034216-PROTECTOR). This research was also supported by the PERCEPTIONS project under European Union's H2020 Research & Innovation Action, Grant Agreement No. 833870.

References

Gavin Abercrombie, Valerio Basile, Verena Rieser, Sara Tonelli, and Alexandra Uma, editors. 2022. *Proceedings of the 1st International Workshop on Perspectivist Approaches to NLP*. European Language Resources Association, Marseille, France.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium. Association for Computational Linguistics.

Aida Mostafazadeh Davani, Mark Díaz, and Vinodkumar Prabhakaran. 2022. Dealing with Disagreements: Looking Beyond the Majority Vote in Subjective Annotations. *Transactions of the Association for Computational Linguistics*, 10:92–110.

Mark Davies. 2013. Corpus of news on the web (now): 3+ billion words from 20 countries, updated every day. Available online at `https://corpus.byu.edu/now/`.

Susan T Fiske. 1993. Controlling other people: The impact of power on stereotyping. *American psychologist*, 48(6):621.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Elisa Leonardelli, Stefano Menini, Alessio Palmero Aprosio, Marco Guerini, and Sara Tonelli. 2021. Agreeing to disagree: Annotating offensive language datasets with annotators' disagreement. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10528–10539, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alan Ramponi and Sara Tonelli. 2022. Features or Spurious Artifacts? Data-centric Baselines for Fair and Robust Hate Speech Detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Seattle, Washington, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Alexandra N Uma, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, and Massimo Poesio. 2021. Learning from disagreement: A survey. *Journal of Artificial Intelligence Research*, 72:1385–1470.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

Gloria Wong, Annie O Derthick, EJR David, Anne Saw, and Sumie Okazaki. 2014. The what, the why, and the how: A review of racial microaggressions research in psychology. *Race and social problems*, 6(2):181–200.

Appendix

A PCL categories

In the following, we provide the precise definitions of the PCL categories according to Pérez-Almendros et al. (2020).

Unbalanced power relations "*By means of the language, the author distances themselves from the community or the situation they are talking about, and expresses the will, capacity or responsibility to help them. It is also present when the author entitles themselves to give something positive to others in a more vulnerable situation, especially when what the author concedes is a right which they do not have any authority to decide to give.*"

Shallow solution "*A simple and superficial charitable action by the privileged community is presented either as life-saving/life-changing for the unprivileged one, or as a solution for a deep-rooted problem.*"

Presupposition "*When the author assumes a situation as certain without having all the information, or generalises their or somebody else's experience as a categorical truth without presenting a valid, trustworthy source for it (e.g. a research work or survey). The use of stereotypes or cliches are also considered to be examples of presupposition.*"

Authority voice "*When the author stands themselves as a spokesperson of the group, or explains or advises the members of a community about the community itself or a specific situation they are living.*"

Metaphor "*[Metaphors] can conceal PCL, as they cast an idea in another light, making a comparison between unrelated concepts, often with the objective of depicting a certain situation in a softer way. [...] Euphemisms are considered as an example of metaphors.*"

Compassion "*The author presents the vulnerable individual or community as needy, raising a feeling of pity and compassion from the audience towards them. It is commonly characterized by the use of flowery wording that does not provide information, but the author enjoys the detailed and poetic description of the vulnerability.*"

The poorer, the merrier "*The text is focused on the community, especially on how the vulnerability makes them better (e.g. stronger, happier or more resilient) or how they share a positive attribute just for being part of a vulnerable community. People living vulnerable situations have values to admire and learn from. The message expresses the idea of vulnerability as something beautiful or poetic. We can think of the typical example of 'poor people are happier because they don't have material goods'.*"

Hyperparameter	Value
Optimizer	AdamW
β_1, β_2	0.9, 0.99
Dropout	0.3
Epochs	10
Batch size	32
Learning rate (LR)	0.0001
LR scheduler	Slanted triangular
Decay factor	0.38
Cut fraction	0.2
Main task loss weight	1
Aux task loss weight	0.25
Aux's aux task loss weight	0.125

Table 7: Hyperparameter values used for all our experiments.

B Hyperparameters

The hyperparameter setting for all our models is presented in Table 7. This reflects the default MaChAmp's hyperparameter values (van der Goot et al., 2021), with the addition of loss weights, as introduced in Section 4.1, and 10 epochs of training as suggested in the original RoBERTa publication (Liu et al., 2019).

C Credits

People icons included in Figure 1 are by `https://icons8.com/`.

PALI-NLP at SemEval-2022 Task 4: Discriminative Fine-tuning of Transformers for Patronizing and Condescending Language Detection

Dou Hu and **Mengyuan Zhou** and **Xiyang Du** and **Mengfei Yuan**
and **Meizhi Jin** and **Lianxin Jiang** and **Yang Mo** and **Xiaofeng Shi**
Ping An Life Insurance Company of China, Ltd.
{HUDOU470, ZHOUMENGYUAN425, DUXIYANG037, YUANMENGFEI854,
JINMEIZHI005, JIANGLIANXIN769, MOYANG853, SHIXIAOFENG309}
@pingan.com.cn

Abstract

Patronizing and condescending language (PCL) has a large harmful impact and is difficult to detect, both for human judges and existing NLP systems. At SemEval-2022 Task 4, we propose a novel Transformer-based model and its ensembles to accurately understand such language context for PCL detection. To facilitate comprehension of the subtle and subjective nature of PCL, two fine-tuning strategies are applied to capture discriminative features from diverse linguistic behaviour and categorical distribution. The system achieves remarkable results on the official ranking, including 1st in Subtask 1 and 5th in Subtask 2. Extensive experiments on the task demonstrate the effectiveness of our system and its strategies.

1 Introduction

"*Don't worry, I know this is a mistake you usually make, we all make it sometimes, but I am bringing you a solution.*", which is a typical example of Patronizing and Condescending Language (PCL) (Giles et al., 1993; Huckin, 2002), shows a superior attitude and apparent kindness towards others, while is generally expressed unconsciously. The impact of PCL can potentially be very harmful, as it feeds inequalities and routinizes discrimination (Ng, 2007), especially if it is geared towards vulnerable communities in the media. If we are able to detect and identify when we are condescending or patronizing towards others, a corrective action (e.g., a more inclusive message) could be taken for a more responsible communication.

Recently, some works (Wang and Potts, 2019; Sap et al., 2020) on PCL are gradually emerging in NLP community. Remarkably, Pérez-Almendros et al. (2020) have shown that general pre-trained language models (Devlin et al., 2019; Liu et al., 2019) can achieve nontrivial performance. However, the behaviour of PCL is usually more unconscious, subtler, and subjective than other harmful types of discourse that are widely studied, i.e., hate speech (Basile et al., 2019), offensive language (Zampieri et al., 2019), intended sarcasm (Du et al., 2022), fake news (Zhang et al., 2021b) and rumor (Wei et al., 2021). These characteristics make PCL detection a difficult challenge, both for human judges and existing NLP systems.

To address this, we propose a novel Transformer-based model BERT-PCL (and its ensembles) with two **discriminative fine-tuning** strategies, to accurately understand such language context for PCL detection. The two strategies are grouped layer-wise learning rate decay (Grouped LLRD) and weighted random sampler (WRS), and both are beneficial for task-adaptive fine-tuning based on language models.

A brief description of these two strategies is as follows: **1)** As different layers capture different types of information (Yosinski et al., 2014), Grouped LLRD, a variant of LLRD (Howard and Ruder, 2018; Zhang et al., 2021a), is applied to group hidden layers of the pre-trained Transformer (Vaswani et al., 2017) into different sets and apply different learning rates to each in a certain extent. And then, we can make full use of different layers to capture more diverse and fine-grained linguistic features, which can boost understanding of the subtle and subjective nature of PCL. **2)** There is a quite common phenomenon that positive samples (patronizing or condescending) have a smaller number than the negative, which reflects usage rates of PCL in public forums (Wang and Potts, 2019; Pérez-Almendros et al., 2020). But the positive samples are more important when detecting PCL, due to the harmful impacts. To deal with imbalanced classes scenarios, we introduce WRS to place more emphasis on the minority classes. Under this strategy, our classifier can capture discriminative features from the categorical distribution and detect whether it contains PCL in an unbiased manner.

At SemEval-2022 Task 4 (Pérez-Almendros

et al., 2022), our proposed system achieves **1st in Subtask 1** and **5th in Subtask 2** on the evaluation leaderboard[1]. Meanwhile, in the post-evaluation phase, we further verified the results of the system on the test set of both subtasks. For Subtask 1, the single model BERT-PCL and its ensembles obtained **63.69%** and **65.41%** performance in terms of F1 of positive class, respectively. For Subtask 2, the single model BERT-PCL and its ensembles obtained **43.28%** and **45.66%** performance in terms of macro-average F1, respectively. Moreover, a series of experiments are conducted on the two subtasks of PCL detection. Results consistently demonstrate that our model and its ensembles significantly outperform comparison methods and the effectiveness of two strategies used in our system.

2 Background

2.1 Task and Data Description

The aim of SemEval-2022 Task 4 (Pérez-Almendros et al., 2022) is to identify PCL, and to categorize the linguistic techniques (categories) used to express it, specifically when referring to vulnerable communities in the media.

This challenge is divided into two subtasks, each corresponding to a subset of the *Don't Patronize Me!* (DPM) dataset (Pérez-Almendros et al., 2020). The 10,469 annotated paragraphs (i.e., sentences in context) from the DPM corpus are used as training data, where each paragraph mentions one or several predefined vulnerable communities. These paragraphs are collected using a keyword-based strategy and cover English language news sources from 20 different countries. A short description of the two subtasks and training data is as follows:

- **Subtask 1: Binary classification**. Given a paragraph, a system must predict whether or not it contains any form of PCL. The training set consists of 10,469 paragraphs annotated with a label ranging from 0 to 4. Label 2, 3, and 4 means positive examples (condescending or patronizing) of PCL and the remaining labels means negatives.

- **Subtask 2: Multi-label classification**. Given a paragraph, a system must identify the categories of PCL that are present. The 993 unique paragraphs (positive examples) in the training set, totaling 2,760 instances of PCL, are labeled with one or more PCL categories: *Unbalanced power relations*, *Shallow solution*, *Presupposition*, *Authority voice*, *Metaphor*, *Compassion*, *The poorer, the merrier*.

In addition, the test set for the evaluation phase contains around 4,000 manually annotated paragraphs with the PCL annotation scheme. More details about the task can be found on the competition page[2].

2.2 Related Work

Harmful language detection/recognition has been widely studied in various forms of discourse, such as hate speech (Basile et al., 2019), offensive language (Zampieri et al., 2019), intended sarcasm (Du et al., 2022), fake news (Zhang et al., 2021b) and rumors (Wei et al., 2021; Hu et al., 2021). Unlike these works generally focused on explicit, aggressive and flagrant phenomena, the study of patronizing and condescending Language (PCL) (Giles et al., 1993; Huckin, 2002; Chouliaraki, 2006; Margić, 2017) has been almost ignored in NLP community until recently.

To encourage more research on PCL language, Wang and Potts (2019) present a condescension detection task and provides a *TALKDOWN* dataset in comment-reply pairs from Reddit. Besides, Pérez-Almendros et al. (2020) introduce a *Don't Patronize Me!* dataset and the challenge of PCL detection towards vulnerable communities (e.g. refugees, homeless people, poor families). These works establish several advanced baselines using pre-trained language models (Devlin et al., 2019; Liu et al., 2019), and suggest that detecting such language is a challenging task both for humans and NLP systems due to its subtle and subjective nature.

3 System Overview

In this section, we review our system adopted in SemEval-2022 Task 4, where we design a novel Transformer-based model BERT-PCL (and its ensembles) with two discriminative fine-tuning strategies for both subtasks of PCL detection.

3.1 Model Architecture

BERT (Devlin et al., 2019) uses masked language models to enable pretrained deep bidirectional representations, and can be fine-tuned to create task-

[1] https://sites.google.com/view/pcl-detection-semeval2022/ranking

[2] https://competitions.codalab.org/competitions/34344

specific models with powerful performance (Wei et al., 2020; Hu and Wei, 2020). Inspired by this, our system utilizes Transformers (Vaswani et al., 2017) to learn contextual representations of the input sentence under the BERT-like architecture.

Formally, given an input token sequence $x_{i1}, ..., x_{iN}$ where x_{ij} refers to j-th token in the i-th input sample, and N is the maximum sequence length, the model learns to generate the context representation of the input token sequences:

$$\mathbf{h}_i = \text{BERT}([\text{CLS}], x_{i1}, ..., x_{iN}, [\text{SEP}]), \tag{1}$$

where [CLS] and [SEP] are special tokens, usually at the beginning and end of each sequence, respectively. $\mathbf{h}_i$ indicates the hidden representation of the i-th input sample, computed by the representation of [CLS] token in the last layer of the encoder.

3.2 PCL Detection

3.2.1 Subtask 1: Binary Classification

Subtask 1, a binary classification task, aims to predict whether or not a paragraph contains any form of PCL. After encoding, we apply a fully connected layer with the Softmax function to predict whether or not the input contains any form of PCL:

$$\hat{\mathbf{y}}_i = Softmax(\mathbf{W}\mathbf{h}_i + \mathbf{b}), \tag{2}$$

where $\mathbf{W}$ and $\mathbf{b}$ are trainable parameters. We leverage Cross-entropy loss to optimize the system. The objective function of Subtask 1 is defined as:

$$L = -\frac{1}{N}\sum_i (\mathbf{y}_i \log(\hat{\mathbf{y}}_i) + (1 - \mathbf{y}_i)\log(1 - \hat{\mathbf{y}}_i)) \tag{3}$$

where $\mathbf{y}_i$ is the ground-truth label of PCL.

3.2.2 Subtask 2: Multi-Label Classification

Subtask 2 is a multi-label classification task. Its goal is to determine which PCL categories a paragraph expresses. After encoding, we also apply a fully connected layer with the sigmoid function to predict the probability of each PCL class:

$$\hat{\mathbf{y}}_i^c = \sigma(\mathbf{W}^c\mathbf{h}_i + \mathbf{b}^c), \tag{4}$$

where σ is the sigmoid function. $\mathbf{W}^c$ and $\mathbf{b}^c$ are trainable parameters. We use Binary Cross Entropy (BCE) loss (Bengio et al., 2013) for the multi-label classification task, denoted as:

$$L = -\frac{1}{N}\sum_i \sum_{c=1}^{M} [\mathbf{y}_i^c \log(\hat{\mathbf{y}}_i^c) + (1 - \mathbf{y}_i^c)\log(1 - \hat{\mathbf{y}}_i^c)], \tag{5}$$

where M is the number of classes, $\hat{\mathbf{y}}_i^c$ indicates the predicted probability that the i-th sample belongs to the c-th class.

3.3 Fine-tuning Strategies

For discriminative fine-tuning of the model, we introduce two strategies to boost the accurate understanding of PCL context, namely grouped layer-wise learning rate decay (Grouped LLRD) and weighted random sampler (WRS).

3.3.1 Grouped LLRD

As different layers capture different types of information (Yosinski et al., 2014), they should be fine-tuned to different extents. Therefore, instead of using the same learning rate for all hidden layers of the Transformer, we tune each layer with different learning rates. Layer-wise learning rate decay (LLRD) (Howard and Ruder, 2018; Zhang et al., 2021a) is a popular fine-tuning strategy that applies higher learning rates for top layers and lower learning rates for bottom layers. Inspired by this, we group layers into different sets and apply different learning rates to each, denoted as Grouped LLRD.

Formally, we split all hidden layers of the Transformer into G sets with embeddings attached to the first set. The parameters of layers are denoted as $\{\theta^1, ..., \theta^G\}$, where θ^g refers to the g-th group. The corresponding learning rate values are denoted as $\{\eta^1, ..., \eta^G\}$, where η^g indicates the learning rate of the g-th group. To capture discriminative features, a multiplicative decay rate λ is used to change relative value of initial learning rates from adjacent groups in a controlled fashion. At time step t, the update of parameters θ is computed by:

$$\theta_t^g = \theta_{t-1}^g - \eta^g \cdot \nabla_{\theta^g} J(\theta), \tag{6}$$

where $\nabla_{\theta^g} J(\theta)$ is the gradient with regard to the model's objective function. The learning rate of the lower layer is applied as $\eta^{g-1} = \eta^g/\lambda$ during fine-tuning to decrease the learning rate group-by-group. In addition, same as LLRD, we use a learning rate that is slightly higher than the top hidden layer for the pooler head and classifier.

Under the above setting, we can capture more diverse and fine-grained linguistic features by flexibly optimizing different hidden layers of the Transformer. It can boost understanding of PCL's subtle and subjective nature.

3.3.2 Weighted Random Sampler

The PCL dataset is highly imbalanced, which causes problems for training the above models. To alleviate this imbalanced classes problem, we use a Weighted Random Sampler (WRS) to place more emphasis on the minority classes. The samples are weighted and the probability of each sample being selected is determined by its relative weight.

For both subtasks, the sampling weight of the i-th sample is computed by:

$$s_i = \begin{cases} 1/\sqrt{\kappa_p}, & \text{if it contains PCL,} \\ 1/\sqrt{\kappa_n}, & \text{otherwise,} \end{cases} \tag{7}$$

where κ_p and κ_n refer to the ratio of positive and negative examples of PCL in training data, respectively. Then, the elements are sampled based on the passed weights. It is worth noting that the number of samples is equal to the length of the training set. During training, the sampler tends to select samples from positive examples with small data volume. In this way, we can have positive and negative classes with equal probability. And the classifier can capture discriminative features from categorical distribution in an unbiased manner.

3.4 Ensemble

For the final submissions, we apply a voter-based fusion technique (Morvant et al., 2014) to ensemble several BERT-PCL models. Concretely, we train the proposed BERT-PCL with five different random seeds. Then, we select Top-3 models according to average result of k-fold cross-validation on the training data. Finally, results of the test set predicted by the three optimal models are voted to get the final submission.

4 Experimental Setup

4.1 Comparison Methods

We compare BERT-PCL and its ensembles with the following several methods:

- **Random** is based on random guessing, choosing each class/label with an equal probability.

- **BERT** (Devlin et al., 2019) is a language model pre-trained in a self-supervised fashion based on deep bidirectional transformers. We use *bert-base-uncased*[3] to initialize BERT.

- **ALBERT** (Lan et al., 2020) presents two parameter-reduction techniques to lower memory consumption and increase the training speed of BERT. We use *albert-large-v2*[3] to initialize ALBERT.

- **ERNIE 2.0** (Sun et al., 2020) is a continual pre-training framework, which builds and learns incrementally pre-training tasks through constant multi-task learning. We use *nghuyong/ernie-2.0-large-en*[3] to initialize ERNIE 2.0.

- **RoBERTa** (Liu et al., 2019) optimizes the training procedure of BERT and removes the next sentence predict objective when pre-training. We use *roberta-large*[3] to initialize RoBERTa.

4.2 Implementation Details

For both subtasks, stratified k-fold cross validation (Kohavi, 1995; Sechidis et al., 2011) is performed to split limited training data into 5 folds. We choose the optimal hyperparameter values based on the the average result of validation sets for all folds, and evaluate the performance of systems on the test data. BERT-PCL is initialized with the *roberta-large*[3] parameters, due to the nontrivial and consistent performance in both subtasks. Following Pérez-Almendros et al. (2020, 2022), the evaluation metrics are F1 over the positive class for Subtask 1 and macro-average F1 for Subtask 2.

We group layers into 3 groups, i.e., $G = 3$. The learning rate for layers in the lower, median, and higher groups as η/λ, η, and $\eta * \lambda$, respectively, where η is set to 1e-5. λ is a hyperparameter searched from $\{0.6, 1.6, 2.6, 3.6, 4.6, 5.6, 6.6\}$. For the training of BERT-PCL, the optimal value of $lambda$ is 1.6 for Subtask 1 and 3.6 for Subtask 2. The experiments are conducted with batch size of 4, maximum length of 250, and dropout rate of 0.4. The number of epochs is set to 10 and the maximum patience number of early stopping is set to 50 batches. AdamW optimizer (Kingma and Ba, 2015) is used with a weight decay of 0.01. A cosine annealing schedule (Loshchilov and Hutter, 2017) is applied to decay the learning rate, with a linear warmup for the first 10% steps.

To effectively utilize the country term and search keyword term corresponding to each paragraph in the corpus, we concatenate these terms with the original paragraph as the input sequence. In the

[3]https://huggingface.co/

	P	R	F1
Random	8.98	55.21	15.45
BERT	56.20	48.58	52.12
ALBERT	59.43	32.81	42.28
ERNIE 2.0	59.24	58.68	58.95
RoBERTa	60.65	64.67	62.60
BERT-PCL	64.31	63.09	**63.69**
Ensemble 1.0 †	64.60	65.62	65.10
Ensemble 2.0	65.20	65.62	**65.41**

Table 1: Results for the problem of detecting PCL, viewed as a binary classification problem (Subtask 1). The results are reported in terms of the precision (P), recall (R) and F1 score of the positive class. All compared pre-trained models are fine-tuned on the task dataset. † indicates the results on the official ranking.

implementation, two special token pairs (i.e., <e> and </e>) are introduced as the term boundary.

5 Result and Discussion

5.1 Overall Result

The overall results in both subtasks are summarized in Table 1 and 2. Unsurprisingly, all pre-trained models clearly outperform the random baseline. The proposed BERT-PCL and its ensembles (i.e., Ensemble 1.0 and Ensemble 2.0) consistently obtain the best performance than the comparison methods on both subtasks. Specifically, BERT-PCL gains 1.09% and 5.39% absolute improvements for Subtask 1 and 2, respectively. These results show the superiority of our models.

In Table 1, both Ensemble 1.0 and Ensemble 2.0 are fused by three optimal full BERT-PCL models with different seeds and obtain a better performance than BERT-PCL in Subtask 1. In Table 2, Ensemble 1.0 is fused by the three optimal BERT-PCL that removes WRS, since we found that it is worse when performing WRS according to weights of category label in Subtask 2. Different from it, in the post-evaluation phase, we perform WRS according to weights of positive samples (patronizing or condescending) and fuse three optimal full BERT-PCL as Ensemble 2.0. As shown in Table 2, Ensemble 2.0 obtains a better performance than BERT-PCL and Ensemble 1.0 in Subtask 2.

Then, we qualitatively analyze the performance of BERT-PCL and typical baselines on the validation set for both subtasks. The results are illustrated in Figure 1. From the figure, BERT-PCL consistently obtains the best performance on the validation set for both subtasks, which confirms again the superiority of the proposed method.

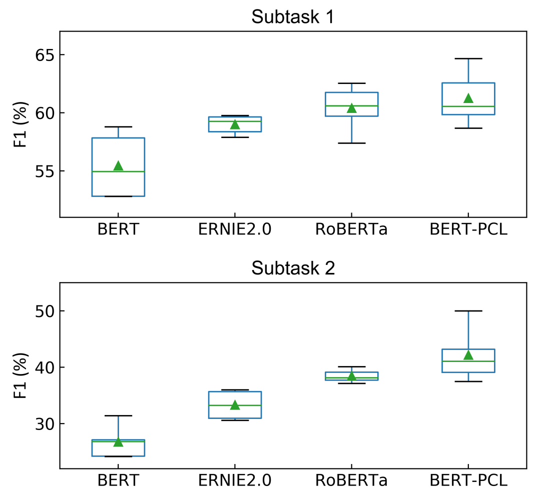


Figure 1: Results on the validation set for both subtasks. The box displays the distribution of results where the green triangle indicates the mean of results, the green line and two blue lines represent the 25%, 50%, and 75% quartiles, respectively, and black lines are the maximum and minimum values. For Subtask 1, we report F1 score of the positive class; and for Subtask 2, we list macro-average F1 score.

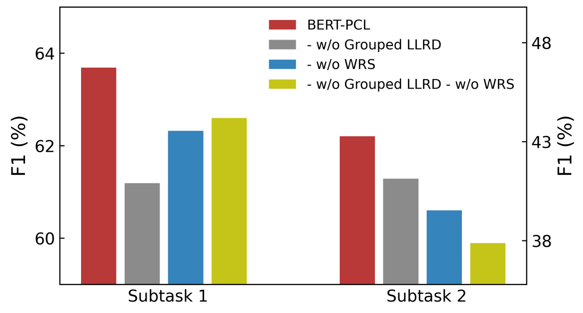


Figure 2: Results of ablation study for PCL detection. For Subtask 1, we report F1 score of the positive class; and for Subtask 2, we list macro-average F1 score.

5.2 Ablation Study

We conduct ablation studies by removing key components of BERT-PCL: 1) **- w/o Grouped LLRD** refers to removing the Grouped LLRD. 2) **- w/o WRS** refers to removing the WRS. 3) **- w/o Grouped LLRD - w/o WRS** refers to removing both Grouped LLRD and WRS, degenerated to RoBERTa.

Figure 2 shows results of ablation studies on two subtasks of PCL detection. The full model yields the best performance on both subtasks. When removing either Grouped LLRD or WRS, the results

	Random	BERT	ALBERT	ERNIE 2.0	RoBERTa	**BERT-PCL**	Ensemble 1.0 [†]	**Ensemble 2.0**
unb.	10.82	51.52	51.31	56.50	57.41	58.90	62.78	61.40
shal.	2.23	40.62	41.79	55.88	50.60	50.55	54.76	55.81
pres.	3.03	20.29	16.39	25.97	26.83	42.86	34.15	39.13
auth.	4.21	6.98	9.64	16.49	26.79	28.07	34.19	34.15
met.	1.91	8.70	9.52	28.07	40.51	40.00	33.33	43.84
comp.	5.98	42.48	40.00	45.90	48.28	49.24	50.82	49.61
merr.	1.22	0.00	0.00	0.00	14.81	33.33	8.70	35.71
Average	4.20	24.37	24.09	32.69	37.89	**43.28**	39.82	**45.66**

Table 2: Results for the problem of categorizing PCL, viewed as a paragraph-level multi-label classification problem (Subtask 2). We report the per-class F1 and macro-average F1. All compared pre-trained model are fine-tuned on the task dataset. [†] indicates the results on the official ranking. The considered seven categories are *Unbalanced power relations* (unb.), *Shallow solution* (shal.), *Presupposition* (pres.), *Authority voice* (auth.), *Metaphor* (met.), *Compassion* (comp.) and *The poorer, the merrier* (merr.).

of variants decline significantly on both subtasks. Specifically, when only removing Grouped LLRD, the model achieves 2.50% and 2.14% degradation of performance in Subtask 1 and 2, respectively. When only removing WRS, the results decline by 1.37% and 3.74% in terms of F1 scores in Subtask 1 and 2, respectively. The above results consistently indicate the effectiveness of the two components.

When removing both components, the performance also decreases on both subtasks. Note that **- w/o Grouped LLRD - w/o WRS** achieves a better F1 score of the positive class than **- w/o WRS** or **- w/o Grouped LLRD** in Subtask 1. This can be explained that ignoring Grouped LLRD limits to explore diverse features of positive samples, and further removing WRS may magnify this limitation due to the imbalanced class problem. Therefore, the model with two modules removed yields slightly better results than ablation models with only one module removed. Different from Subtask 1, Subtask 2 is a multi-label classification problem and we report the macro-average F1 score. Using Grouped LLRD can capture diverse features of each category label in positive samples, and WRS according to weights of positive and negative samples further promotes the model's attention to positive samples. Hence, removing both modules obtains the worst performance in Subtask 2.

5.3 Parameter Analysis

In this part, we explore the performance of BERT-PCL against different λ in Grouped LLRD. Bottom groups often encode more general and broad-based information, while top groups closer to the output encode information more localized and specific to the task on hand. In our model, a suitable value λ can control and balance these different layers of the

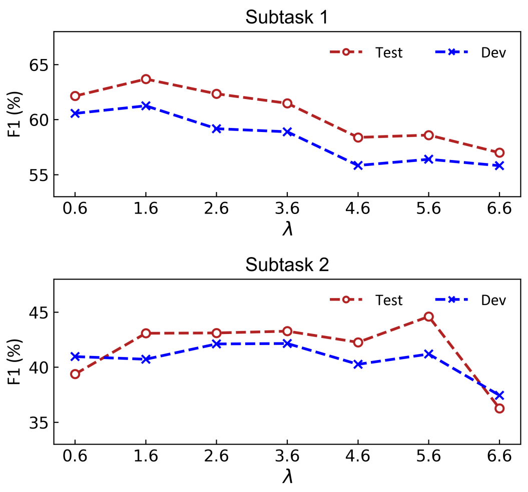


Figure 3: Results against different values of hyperparameter λ in Grouped LLRD on both test and validation sets. We report F1 score of the positive class for Subtask 1, and list macro-average F1 score for Subtask 2.

Transformer to capture different kinds of features from diverse linguistic behaviour.

The results are illustrated in Figure 3. Based on k-fold cross validation on training data, we find the local optimum of λ, 1.6 for Subtask 1 and 3.6 for Subtask 2, and the resulting model consistently performed excellently on the test set. Under the optimal setting, different layers in the model can capture more diverse and fine-grained linguistic features, enhancing the understanding of the subtle and subjective nature of PCL. However, larger λ would make the model overfit to small datasets and suffer catastrophic forgetting during fine-tuning. Hence, the performance degrades as λ increases.

It is worth noting that in when λ is up to 5.6 for Subtask 2, the model achieves suboptimal results on the validation set but performs exceptionally well on the test set. This may be because the model

No.	Para.	Gold	Pred. (BERT-PCL)	Pred. (RoBERTa)	Pred. (ERNIE 2.0)
1	*“Jesus is the Master Feminist because he championed the cause of women,” she said.*	pos.	pos.	neg.	neg.
2	*There is a saying that goes "A friend in need is a friend indeed. " This means, a good friend is the one who rescues a friend trapped in unsolved problems.*	neg.	neg.	neg.	pos.
3	*"The government is implementing several schemes that would change the economic position of poor families," she added.*	pos.	neg.	neg.	neg.
4	*Alexis and her family decided to donate more than 400 of those presents to children in need.*	neg.	pos.	pos.	pos.

Table 3: Case studies in Subtask 1: Binary Classification. The table shows four examples of paragraphs, their gold labels and predictions by three methods (BERT-PCL, RoBERTa and ERNIE 2.0). The pos. means the positive class of PCL, i.e. as instances containing PCL. Likewise, the neg. means the negatives.

No.	Para.	Gold	Pred. (BERT-PCL)	Pred. (RoBERTa)	Pred. (ERNIE 2.0)
1	*Through Gawad Kalinga, Meloto has proven to be a key player in the housing industry, helping provide decent homes and sustainable livelihood to the marginalized and homeless Filipinos.*	unb., comp.	unb., comp.	-	unb., comp.
2	*Pope Francis will visit a tiny Italian island to greet refugees and immigrants, pray for those who have lost their lives at sea and call for greater solidarity.*	unb., shal.	unb., shal.	-	-
3	*In South Africa, education is a right and not a privilege, but an unfavourable background can unconsciously infringe on this right.*	unb., pres., met.	unb., met.	unb., auth.	unb.
4	*Thankfully, while Krishna Tulasi can’t entirely escape from the trope of disabled persons with hearts of gold, it manages to do better than many previous films with disabled protagonists.*	merr.	-	-	-

Table 4: Case studies in Subtask 2: Multi-Label Classification. The table shows four examples of paragraphs, their gold labels and predictions by three methods (BERT-PCL, RoBERTa and ERNIE 2.0). The categories stand for: *Unbalanced power relations* (unb.), *Shallow solution* (shal.), *Presupposition* (pres.), *Authority voice* (auth.), *Metaphor* (met.), *Compassion* (comp.) and *The poorer, the merrier* (merr.).

overfits some redundant features of the corpus.

5.4 Case Study

Table 3 and Table 4 show several typical examples in the training set from Subtask 1 and 2, respectively. Their gold labels and predictions by BERT-PCL, RoBERTa and ERNIE 2.0 are presented in the corresponding columns.

In Table 3, the first case is correctly classified by BERT-PCL, while is misclassified by other methods. We can easily observe that this example has the characteristics of *Unbalanced power relations* and *Authority voice*, and the language expression of the latter is more subtle. Unlike other methods, BERT-PCL can capture the linguistic phenomena of PCL through a discriminative fine-tuning process, and thus detect them correctly. For the second, BERT-PCL and RoBERTa can accurately identify the positive paragraphs, using the sentence representation ability learned by the pre-trained model. The latter two examples are consistently predicted as false negatives and false positives by all methods, respectively. We notice that both paragraphs have been annotated by two human annotators as borderline PCL. Unsurprisingly, these methods also struggle to detect such cases.

As seen in Table 4, only BERT-PCL can correctly determine fine-grained PCL categories of the first two cases, which again illustrates the superiority of our method. It can be noticed that the third example has three PCL sub-categories (i.e., unb., pres., met.) with a certain internal correlation, and the gold label (i.e., merr.) of the fourth example appears too little in the training set. These phenomena increase the difficulty of identifying the two examples, which leads to wrong predicted labels. We believe that identifying multiple related sub-categories simultaneously and controlling the imbalance of positive PCL labels are urgent challenges for Subtask 2.

6 Conclusion

In this paper, we propose an advanced BERT-like model and its ensembles to accurately understand and detect patronizing and condescending language. Based on the pre-trained Transformer, we apply two fine-tuning strategies to capture discriminative features from diverse linguistic behaviour and categorical distribution. At SemEval-2022 Task 4, our system achieves 1st in Subtask 1 and 5th in Subtask 2 on the official ranking. Extensive experiments demonstrate the effectiveness and superiority of the proposed system and its strategies.

Acknowledgements

The research is supported by the Ping An Life Insurance. All the work in this paper are conducted during the SemEval-2022 Competition. Last but not least, we thank the SemEval-2022 organizers for their effort in preparing the challenge, and the reviewers for their insightful and constructive comments.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *SemEval@NAACL-HLT*, pages 54–63. Association for Computational Linguistics.

Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828.

Lilie Chouliaraki. 2006. *The spectatorship of suffering*. Sage.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Xiyang Du, Dou Hu, Meizhi Jin, Lianxin Jiang, Yang Mo, and Xiaofeng Shi. 2022. PALI-NLP at SemEval-2022 Task 6: iSarcasmEval- Fine-tuning the Pre-trained Model for Detecting Intended Sarcasm. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Howard Giles, Susan Fox, and Elisa Smith. 1993. Patronizing the elderly: Intergenerational evaluations. *Research on Language and Social Interaction*, 26(2):129–149.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL (1)*, pages 328–339. Association for Computational Linguistics.

Dou Hu and Lingwei Wei. 2020. SLK-NER: exploiting second-order lexicon knowledge for chinese NER. In *SEKE*, pages 413–417. KSI Research Inc.

Dou Hu, Lingwei Wei, Wei Zhou, Xiaoyong Huai, Jizhong Han, and Songlin Hu. 2021. A rumor detection approach based on multi-relational propagation tree. *Journal of Computer Research and Development*, 58(7):1395.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Ron Kohavi. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, pages 1137–1145. Morgan Kaufmann.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*. OpenReview.net.

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

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *ICLR (Poster)*. OpenReview.net.

Branka Drljača Margić. 2017. Communication courtesy or condescension? linguistic accommodation of native to non-native speakers of english. *Journal of English as a lingua franca*, 6(1):29–55.

Emilie Morvant, Amaury Habrard, and Stéphane Ayache. 2014. Majority vote of diverse classifiers for late fusion. In *S+SSPR*, volume 8621 of *Lecture Notes in Computer Science*, pages 153–162. Springer.

Sik Hung Ng. 2007. Language-based discrimination: Blatant and subtle forms. *Journal of Language and Social Psychology*, 26(2):106–122.

Carla Pérez-Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *ACL*, pages 5477–5490. Association for Computational Linguistics.

Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis P. Vlahavas. 2011. On the stratification of multi-label data. In *ECML/PKDD (3)*, volume 6913 of *Lecture Notes in Computer Science*, pages 145–158. Springer.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A continual pre-training framework for language understanding. In *AAAI*, pages 8968–8975. AAAI Press.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. In *EMNLP/IJCNLP (1)*, pages 3709–3717. Association for Computational Linguistics.

Lingwei Wei, Dou Hu, Wei Zhou, Xuehai Tang, Xiaodan Zhang, Xin Wang, Jizhong Han, and Songlin Hu. 2020. Hierarchical interaction networks with rethinking mechanism for document-level sentiment analysis. In *ECML/PKDD (3)*, volume 12459 of *Lecture Notes in Computer Science*, pages 633–649. Springer.

Lingwei Wei, Dou Hu, Wei Zhou, Zhaojuan Yue, and Songlin Hu. 2021. Towards propagation uncertainty: Edge-enhanced bayesian graph convolutional networks for rumor detection. In *ACL/IJCNLP (1)*, pages 3845–3854. Association for Computational Linguistics.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *SemEval@NAACL-HLT*, pages 75–86. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021a. Revisiting few-sample BERT fine-tuning. In *ICLR*. OpenReview.net.

Xueyao Zhang, Juan Cao, Xirong Li, Qiang Sheng, Lei Zhong, and Kai Shu. 2021b. Mining dual emotion for fake news detection. In *WWW*, pages 3465–3476. ACM / IW3C2.

ASRtrans at SemEval-2022 Task 4: Ensemble of Tuned Transformer-based Models for PCL Detection

Ailneni Rakshitha Rao
Indian Institute of Technology, Gandhinagar
rao_ailneni@alumni.iitgn.ac.in

Abstract

Patronizing behavior is a subtle form of bullying and when directed towards vulnerable communities, it can arise inequalities. This paper describes our system for Task 4 of SemEval-2022: Patronizing and Condescending Language Detection (PCL). We participated in both the sub-tasks and conducted extensive experiments to analyze the effects of data augmentation and loss functions used, to tackle the problem of class imbalance. We explore whether large transformer-based models can capture the intricacies associated with PCL detection. Our solution consists of an ensemble of the RoBERTa model which is further trained on external data and other language models such as XLNeT, Ernie-2.0, and BERT. We also present the results of several problem transformation techniques such as Classifier Chains, Label Powerset, and Binary relevance for multi-label classification.

1 Introduction

In this paper, we discuss various linguistic techniques used for detecting Patronizing and condescending language (PCL). This task particularly poses a new challenge in the field of NLP because of its subjectivity, subtle usage, and requirement of world knowledge. A person is said to be condescending or patronizing when he/she uses a superior tone to talk down to people or tries to raise pity by describing their situation. Even though people often use PCL with good intentions, it encourages stereotyping, discrimination and leads to greater exclusion. Therefore, it is important to devise methods that facilitate the automatic detection of PCL. SemEval 2022 Task 4 (Pérez-Almendros et al., 2022) is the first attempt to detect the usage of PCL towards vulnerable communities. It has two subtasks: In sub-task A, we need to detect if the given paragraph contains PCL or not. Sub-task B aims to classify PCL text further among potential intersecting categories.

The two main challenges faced in this task are extreme class imbalance in the dataset and the subtle nature of PCL present in the text which makes it hard, even for humans to classify it correctly. Also, the model needs to differentiate between actual news of extremely vulnerable situations from text containing PCL.

Pre-trained language models such as BERT, XLNeT, etc., have emerged as the state-of-the-art models for many NLP tasks such as text classification, machine translation, sequence tagging, etc. However, they are trained on typical day-to-day texts. PCL text is not trivial and classifying certain classes requires some level of world knowledge and commonsense reasoning. In this paper we conduct detailed experiments using the following models: RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2020), Ernie-2.0 (Sun et al., 2019), BERT-large (Devlin et al., 2019), label specific attention network (LSAN) (Xiao et al., 2019) and their ensembles for PCL detection. Additionally, we test the Classifier Chain approach for multi-label classification.

We achieved significant improvement over the baseline RoBERTa model in both the sub-tasks. We were ranked 52^{nd} with an F1 score of 0.4421 in sub-task A and 33^{rd} with an F1 score of 0.1889 in sub-task B among 81 teams. We release the code for models and experiments via GitHub [1]

The rest of the paper is organized as follows: Section 2 describes the challenge, followed by a brief literature survey. Section 3 explains the proposed approach in detail, while section 4 presents the experimental details required to reproduce the results. Results and analysis are shown in section 5. Finally, conclusions are drawn in section 7.

[1]https://github.com/rak55/ASRtrans-semeval2022

2 Background

2.1 Problem Description

SemEval 2022 Task 4: Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022) is a paragraph-level text classification problem that consists of two sub-tasks. **Task A** is a binary classification task designed to predict if the text consists of any form of PCL. In **task B**, we need to further classify PCL text among potential categories namely *unbalanced power relations*, *shallow solution*, *presupposition*, *authority voice*, *metaphor*, *compassion*, and *the poorer the merrier*.

2.2 Related Work

Hate language detection Though the detection of patronizing and condescending language has not been studied in depth in the field of NLP, extensive work has been done in several forms of harmful language detection such as Automated hate speech detection (Davidson et al., 2017), rumor propagation (Gorrell et al., 2019), fake news detection (Conroy et al., 2015), and trust-worthiness prediction (Barrón-Cedeño et al., 2018). However, recently (Wang and Potts, 2019) introduced a labeled dataset named *TalkDown* derived from Reddit communication threads for modeling condescension. (Sap et al., 2020) introduced *Social Bias Inference* corpus to study the unbalanced power relations present in the condescending language.

Multi-label text classification There are mainly three different techniques to solve a multi-label text classification problem: Binary Relevance, Classifier Chains (Dembczyński et al., 2010) and Label Powerset method (Boutell et al., 2004). Binary Relevance treats each class independently and ignores label dependence. The Label Powerset method considers each combination of labels as a distinct class, thereby transforming a multi-label classification problem into a single-label problem. (Chen et al., 2007) propose document transformation by assigning label weights based on label entropy. (Alvares-Cherman et al., 2012) tries to incorporate label dependency into the Binary Relevance method. ReliefF and Information Gain are combined with Binary Relevance and Label Powerset approaches in (Spolaôr et al., 2013) to evaluate the importance of each label. (Wang et al., 2017) show that regularising the model during the training phase and using support inference during prediction along with F-optimizer improves the F1 score of the multi-label problem.

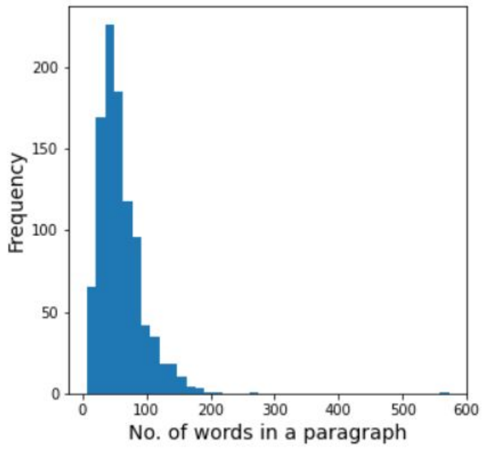


Figure 1: Frequency distribution of the text length.

Label	No. of samples
Unbalanced power relations	230
Shallow solution	716
Presupposition	196
Authority voice	224
Metaphor	469
Compassion	197
The poorer, the merrier	40

Table 1: Distribution of training data for Task B.

Transfer Learning Pre-trained transformer-based language models such as BERT, RoBERTa, etc., often outperform many traditional models trained from scratch. This success can be attributed to their rich contextual embeddings. Therefore, these models are used for many downstream tasks. (Li and Xiao, 2020) use SpanBERT for the detection of propaganda techniques in news articles. (Ranasinghe and Hettiarachchi, 2020) use BERT-based multilingual models for offensive language identification in social media.

3 System Overview

3.1 Data

The *Don't Patronize Me!* dataset (Pérez-Almendros et al., 2020) provided as training data for both the sub-tasks consists of paragraphs extracted from the News on Web (NoW) corpus. The distribution of different labels in the dataset is shown in table 1. Each training example in sub-task A consists of a *doc-id*, *keyword*, *country-code*, *paragraph*, and *label*. The text was retrieved from the news of 20 English-speaking countries based on 10 keywords belonging to vulnerable communities like disabled, homeless, etc. The frequency distribution of text

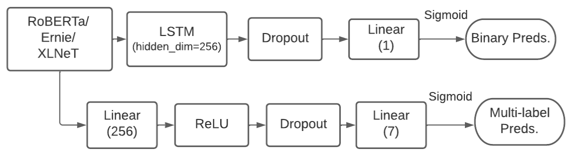


Figure 2: Flow chart for multi-task learning.

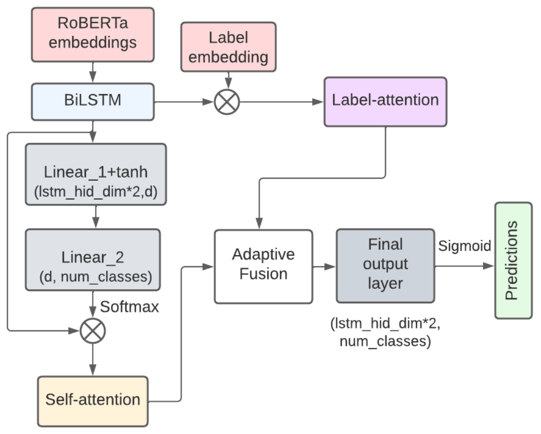


Figure 3: Architecture of LSAN.

length is shown in figure 1. The dataset for task B contains span-level annotation of PCL among seven categories mentioned earlier.

3.2 Transformer-based models

Our approach is to train several pre-trained language models including RoBERTa, Ernie-2.0, XLNet, and BERT-large individually and in an ensemble with different problem transformation approaches for both the sub-tasks. We conduct experiments with two different settings: Multi-task and single-task learning. The model used for multi-task learning is described in figure 2.

3.2.1 Using External data

A RoBERTa model is further trained using the HuggingFace library with metaphor and condescension datasets since the language and content present are similar to our training dataset. This improves the accuracy of the contextual embeddings. We extracted around 7000 text instances for training RoBERTa from the talk-down corpus (Wang and Potts, 2019), MOH corpus (Mohammad et al., 2016), VUA dataset (Steen et al., 2010) and Trofi dataset (Birke and Sarkar, 2006). We call this trained model as tuned RoBERTa (tRoBERTa) for the entirety of this paper.

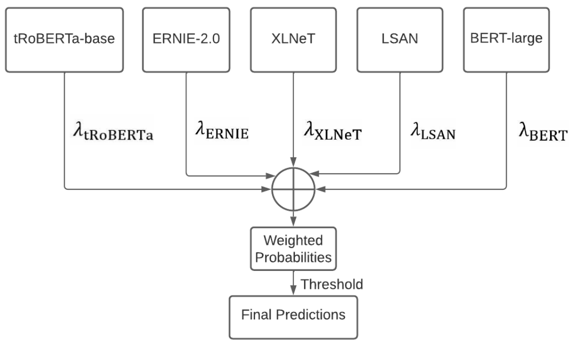


Figure 4: **Weighted-Average Ensemble**. $\lambda_{RoBERTa}$, λ_{ERNIE}, λ_{XLNeT}, λ_{LSAN}, and λ_{BERT} represent the weights of the respective models.

3.3 Label-Specific Attention Network

For sub-task B, in conjunction with transformer-based models, we used LSAN (Label-Specific Attention Network) (Xiao et al., 2019) for multi-label classification. LSAN tries to determine the label-related text from the given paragraph. It has two parts: self-attention and label-attention as shown in the figure 3. Self-attention aims to calculate the contribution of each word to a particular label. While it takes into account the context of the given text, the semantic meaning of labels themselves is captured by label-attention. The importance of these two mechanisms is determined by two trainable fully connected layers. Finally, an MLP layer with Sigmoid activation is used to get the final output.

3.4 Ensembles

Large Language models differ in the training procedures and the datasets on which they are trained. Hence, they may focus on different aspects of the input text even though they give comparable results. Therefore, it is a good practice to combine the results of these language models to get accurate word / sentence embeddings. There are several ways to combine them: we can concatenate embeddings of different models and project them to a low dimensional space for prediction, but this will require high computational power. Instead, we can tune different language models (tRoBERTa, ERNIE, XLNeT, LSAN and BERT) independently on the entire dataset and later combine their predictions as shown in the figure 4. Final results are obtained by taking a weighted average of the predictions. In this case, the weights are obtained by

Model	**Task B**							
	UPR	**SS**	**PS**	**AV**	**M**	**C**	**PM**	**Avg. F1**
Baseline	0.3535	0	0.1667	0	0	0.2087	0	0.1041
Ours	0.186	0.0875	0.0826	0.198	0.1324	0.2784	0.3571	0.1889
Ours*	0.1413	0.1706	0.0594	0.2086	0.2297	0.2874	0.238	0.1904

Table 2: Comparison of test results our models with baseline RoBERTa model for sub-task B. UPR: Unbalanced power relations, SS: Shallow solution, PS: Presupposition, AV: Authority voice, M: Metaphor, C: Compassion, PM: The poorer the merrier.

Model	**Task A**		
	Precision	**Recall**	**F1 score**
Baseline	0.3935	0.653	0.4911
Ours	0.3558	0.5836	0.4421
Ours*	0.5389	0.5678	0.5530

Table 3: Comparison of test results our models with baseline RoBERTa model for sub-task A.

[*] Modified System after submission and not submitted in the task.

grid search on the validation dataset. Another way to combine the predictions is the Voting Ensemble method, where the class predicted by the majority of the models is considered as the final output. We tested both approaches and found that the weighted average method yields better results than the voting ensemble method.

3.5 Classifier Chains

The classifier Chains approach connects binary classifiers in a chain such that the output of one classifier is treated as the input feature for the subsequent classifier. One of the factors that influence the performance of classifier chains is the sequence of labels used for training. The sequence of labels can be decided based on various approaches such as easiest-to-predict labels, most frequent labels first, etc. We tested the path 1–> 2 –> 6–> 7–> 4–> 5–> 3 based on the first approach. In the end we used an ensemble of the paths, 1–> 2–> 3–> 4–> 5–> 6–> 7 (P1) and 1–> 2–> 6–> 7–> 4–> 5–> 3 (P2).

4 Experimental setup

We used Pytorch (Paszke et al., 2019) and HuggingFace library (Wolf et al., 2019) for training and inference. All the models are trained on Google Colab. AdamW (Loshchilov and Hutter, 2019) optimizer with a learning rate of 2e-5 is used for training all transformer-based models and Adam is used for training all the other models. The maximum length of the text is limited to 200. We chose a batch size of 32 for training all the models.

4.1 Text preprocessing

We cleaned the text by removing punctuation, special characters, URLs, etc. We removed the contractions by mapping them to regular text and annotated the sentences with a [CLS] token before passing them into transformer-based models. We did not remove the stopwords as it deteriorated performance. We augmented the data based on contextual augmentation proposed in (Kobayashi, 2018) and back-translation technique.

4.2 Loss functions

We trained our model for sub-task A with binary cross-entropy loss, whereas for sub-task B we used focal loss. Focal loss (Lin et al., 2020) is a form of cross-entropy loss, but it is dynamically scaled. It is computed as:

$$FL(p_t) = (1 - p_t)^{\gamma} \log(p_t)$$

By setting $\gamma > 0$, we are reducing the relative loss for easy, well-classified examples ($p_t > 0.5$). For prediction, we used a threshold of 0.35.

4.3 Training details

We split the original training data into train and development sets with 80% and 20% of the data respectively. For Sub-task A, we used *Stratified-ShuffleSplit* option from *sklearn* library to split the dataset whereas for Sub-task B we used *iterative stratification* of order 1 from *skmultilearn* library. In the case of multi-task models, we used early stopping criteria on dev set independently for each task.

5 Results and Analysis

Our model for sub-task A consists of an ensemble of tuned RoBERTa (tRoBERTa), XLNet, and Ernie-2.0 models trained with binary cross-entropy

Model	Task A	Task B
Ernie-2.0 (ST)	0.535	0.498
tRoBERTa (ST)	0.537	0.503
XLNeT (ST)	0.539	0.500
LSAN	-	0.493
BERT-large	-	0.497
Avg. Ensemble (ST)	0.538	0.502
tRoBERTa (MT)	0.541	0.505
ULMFiT (ST)	0.510	0.487
tRoBERTa (CC with P1)	-	0.548
tRoBERTa (CC with P2)	-	0.550

Table 4: F1 score (Dev) of the other major models for both the sub-tasks. ST stands for single-task models, MT stands for multi-task models, CC stands for Classifier Chains with paths P1, P2 and Avg. Ensemble is the weighted average ensemble method.

Model	Task A	Task B
HAHNN (CNN)	0.530	0.493
C-BiLSTM	0.500	0.485
USE + BiLSTM	0.520	0.485
SVM (word n-grams)	0.486	0.469
SVM (ELMO)	0.492	0.475
SVM (char n-grams)	0.485	0.462
SVM (composite)	0.489	0.471
LR (ELMO)	0.485	0.465
RF (ELMO)	0.484	0.470

Table 5: Results of RNN-based and ML models.

loss. Model for sub-task B consists of a weighted average ensemble of tRoBERTa, XLNet, Ernie-2.0, BERT-large, and LSAN trained with focal binary cross-entropy explained in section 4.2. We also tested the Classifier Chain approach with tuned RoBERTa in post-evaluation phase. For this approach, we tested two sequences of labels P1, P2 (described in 3.5) and their ensemble. The official results of the models for sub-task A and sub-task B along with the baseline results on the test dataset are summarized in table 3 and 2 respectively. Apart from these models, the results of other major transformer-based models are shown in table 4.

Besides transformer-based models, we also tested Hierarchical Attentional Hybrid Neural Networks for Document Classification described in (Abreu et al., 2019), C-BiLSTM model, and a BiLSTM-attention model with USE (Cer et al., 2018) sentence embeddings. In the C-BiLSTM model, we apply convolutional and max-over-time pooling layers on the word embeddings and pass them through a bi-LSTM layer with an attention mechanism. This approach is the combination of work used in (Wang et al., 2016) and (Yang et al., 2016). We implemented traditional machine learning models like Support Vector Machine (SVM), Logistic Regression (LR), and Random Forests (RF). We used TF-IDF on words (both unigrams and bigrams), TF-IDF on character n-grams (1-5 characters), ELMO embeddings (Peters et al., 2018) and the composite features utilized in (Anzovino et al., 2018) as features for the ML models. The composite features are a combination of features based on the adjective count, length of the text, n-grams, POS tags, and doc2vec (Le and Mikolov, 2014). The results of the above RNN-based and ML models are summarized in table 5.

The comparison among three different problem transformation approaches to Multi-label classification is shown in table 6. The Label Powerset (LP) method gives the lowest F1 score as it doesn't perform well with unseen label combinations not covered in the training dataset. Classifier Chain performed better than Binary relevance as it takes label correlations into account. Incremental analysis of our system is shown in table 7. This analysis shows the importance of further training RoBERTa and focal loss.

Data augmentation is widely used to generate slightly variant larger datasets from the existing smaller ones. Since one recurring issue among all the models we trained is overfitting, three types of data augmentation techniques are tested to address this issue. We applied contextual augmentation for labeled sentences as proposed in (Kobayashi, 2018). In this method, we replace the words in a sentence with words predicted by a bi-directional language model. We also tested the back-translation approach proposed in (Sennrich et al., 2016) and easy data augmentation (EDA) techniques described in (Wei and Zou, 2019). We observed that while back-translation and contextual augmentation slightly improved the performance of the model, the use of EDA degraded it. We conclude this is because of a contextual mismatch in the text generated with EDA. Since random deletion is one of the techniques used in EDA, it may have led to the deletion of the keywords like *'them'*, *'us'*, *'poor'* etc., resulting in bad performance.

As the training dataset provided for this task is small and extremely imbalanced, we focused on analyzing the effects of data augmentation and choice

Approach	F1 Macro
Label Powerset	0.479
Classifier Chain	0.506
Binary Relevance	0.492

Table 6: Results of various problem transformation approaches on Dev set.

System	Precision	Recall	F1
RoBERTa	0.529	0.515	0.521
+ Tuned RoBERTa	0.539	0.521	0.530
+ Focal loss	0.550	0.525	0.537
+ Ensemble	0.557	0.530	0.543

Table 7: Sub-task A (Dev): Incremental analysis of our system.

of the loss function in this paper. F1 score for the *the poorer the merrier* class has greatly improved even though it is the least represented class in the entire dataset. Also, the F1 score of the *metaphor* class has improved. This improvement is the result of tuned RoBERTa which is trained on metaphor datasets. *Shallow solution* and *presupposition* are the classes with the lowest F1 score. This is because they require some form of world knowledge. Surprisingly, even though *unbalanced power relations* is the easiest class to predict, our model seemed to struggle with it.

6 Conclusion and Future Work

We have presented the results of various experiments using pre-trained language models such as RoBERTa, XLNeT, Ernie-2.0, BERT-large, and their ensembles for the detection of patronizing and condescending language. For the sake of comparison, we also presented experimental results of several RNN-based and traditional machine learning models with different features such as character n-grams, ELMO embeddings, etc. We also conducted experiments with different problem transformation approaches like Classifier Chains for multi-label classification. Since the dataset for this task is imbalanced, we also tested the effect of several data augmentation techniques and loss functions. We have achieved sizeable improvements over the baseline model by task-specific training and using techniques to mitigate class imbalance. In future work, we plan to explore other forms of Classifier Chains to more effectively model label dependence and hierarchy.

References

Jader Abreu, Luis Fred, David Macêdo, and Cleber Zanchettin. 2019. Hierarchical attentional hybrid neural networks for document classification. *Lecture Notes in Computer Science*, page 396–402.

Everton Alvares-Cherman, Jean Metz, and Maria Carolina Monard. 2012. Incorporating label dependency into the binary relevance framework for multi-label classification. *Expert Syst. Appl.*, 39(2):1647–1655.

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *Natural Language Processing and Information Systems*, pages 57–64, Cham. Springer International Publishing.

Alberto Barrón-Cedeño, T. Elsayed, Reem Suwaileh, Lluís Màrquez i Villodre, Pepa Atanasova, Wajdi Zaghouani, Spas Kyuchukov, Giovanni Da San Martino, and Preslav Nakov. 2018. Overview of the clef-2018 checkthat! lab on automatic identification and verification of political claims. task 2: Factuality. In *CLEF*.

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 329–336, Trento, Italy. Association for Computational Linguistics.

Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. 2004. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Weizhu Chen, Jun Yan, Benyu Zhang, Zheng Chen, and Qiang Yang. 2007. Document transformation for multi-label feature selection in text categorization. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 451–456.

Niall J. Conroy, Victoria L. Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. In *Proceedings of the 78th ASISamp;T Annual Meeting: Information Science with Impact: Research in and for the Community*, ASIST '15, USA. American Society for Information Science.

Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *ICWSM*.

Krzysztof Dembczyński, Weiwei Cheng, and Eyke Hüllermeier. 2010. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 279–286, Madison, WI, USA. Omnipress.

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

Genevieve Gorrell, Elena Kochkina, Maria Liakata, Ahmet Aker, Arkaitz Zubiaga, Kalina Bontcheva, and Leon Derczynski. 2019. SemEval-2019 task 7: RumourEval, determining rumour veracity and support for rumours. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 845–854, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *ArXiv*, abs/1805.06201.

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

Jinfen Li and Lu Xiao. 2020. syrapropa at semeval-2020 task 11: Bert-based models design for propagandistic technique and span detection.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327.

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

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

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33, Berlin, Germany. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

Tharindu Ranasinghe and Hansi Hettiarachchi. 2020. BRUMS at SemEval-2020 task 12: Transformer based multilingual offensive language identification in social media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1906–1915, Barcelona (online). International Committee for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Newton Spolaôr, Everton Alvares Cherman, Maria Carolina Monard, and Huei Diana Lee. 2013. A comparison of multi-label feature selection methods using the problem transformation approach. *Electronic Notes in Theoretical Computer Science*, 292:135–151. Proceedings of the XXXVIII Latin American Conference in Informatics (CLEI).

Gerard J. Steen, Aletta G. Dorst, J. Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A Method for Linguistic Metaphor Identification: From MIP to MIPVU*. John Benjamins.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2019. Ernie 2.0: A continual pre-training framework for language understanding.

Bingyu Wang, Cheng Li, Virgil Pavlu, and Javed A. Aslam. 2017. Regularizing model complexity and label structure for multi-label text classification. *CoRR*, abs/1705.00740.

Jin Wang, Liang-Chih Yu, K. Robert Lai, and Xuejie Zhang. 2016. Dimensional sentiment analysis using a regional CNN-LSTM model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 225–230, Berlin, Germany. Association for Computational Linguistics.

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Lin Xiao, Xin Huang, Boli Chen, and Liping Jing. 2019. Label-specific document representation for multi-label text classification. In *EMNLP*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

LastResort at SemEval-2022 Task 4: Towards Patronizing and Condescending Language Detection using Pre-trained Transformer Based Model Ensembles

Samyak Agrawal **Radhika Mamidi**
International Institute of Information Technology Hyderabad
samyak.agrawal@research.iiit.ac.in,
radhika.mamidi@iiit.ac.in

Abstract

This paper presents our solutions for Task4 at SemEval2022: Patronizing and Condescending Language Detection. This shared task contains two sub-tasks. The first sub-task is a binary classification task whose goal is to predict whether a given paragraph contains any form of patronising or condescending language(PCL). For the second sub-task, given a paragraph, we have to find which PCL categories express the condescension. Here we have a total of 7 overlapping sub-categories for PCL. Our proposed solution uses BERT based ensembled models with hard voting and techniques applied to take care of class imbalances. Our paper describes the system architecture of the submitted solution and other experiments that we conducted. Our best performing models achieve an F1 score of 59.4 and 15.7 on sub-tasks 1 and 2 respectively.

1 Introduction

Patronizing and condescending attitude in language generally denotes the writer's sense of superiority over others. If someone is patronizing or condescending, it means what they write/say is accompanied by a sense of pity or compassion. Often, usage of PCL is relatively unconscious, and the intent of the writer is not to hurt a particular group or person they are referring to. So while being harmless in its intention. Usage of PCL still poses a risk of harming vulnerable people or groups by stereotyping them or normalizing specific behaviour towards them.

Task4 at SemEval-2022 (Pérez-Almendros et al. (2022)), Patronizing and Condescending Language Detection provides two sub-tasks. The goal of subtask1 is to identify if the given paragraph contains PCL. The goal of sub-task2 is to determine which subcategory of PCL expresses the condescension. The seven subcategories are Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion and The poorer, the merrier. A given paragraph can show instances of multiple subcategories.

We experimented with multiple transformer-based models. We used focal loss and Weighted Random Sampling to address the class imbalance; we also tried out ensembling models with hard voting, which improved the accuracy over the baseline models for both the sub-tasks.

The paper is structured as follows: Section 2: describes the dataset and related work. Section 3: describes our system and model architecture. Section 4 has information regarding the dataset size and splits with libraries used. Section 5 discusses the findings from our experiments, and section 6 concludes our paper.

2 Background

There has been work done to detect potentially harmful forms of language. Liu et al. (2019a) used BERT and LSTM based models to detect offensive language in the dataset, Offensive Language Identification Dataset (OLID) provided by Zampieri et al. (2019) at SemEval 2019. Indurthi et al. (2019) used InferSent (Conneau et al. (2018)) semantic sentence representations to detect Hate Speech against Immigrants and Women in the dataset provided by Basile et al. (2019) at SemEval 2019.

PCL's subtle and often unrealised nature makes its detection an arduous task for humans and Artificial Intelligence systems alike. There has been some recent work done when it comes to addressing PCL. Wang and Potts (2019) presents a dataset of social media messages annotated for condescending acts in context.

2.1 Dataset and Task Description

SemEval2022 Task 4 provides the Don't Patronize Me! dataset (Pérez-Almendros et al. (2020)). The dataset contains 10469 paragraphs. We divide the data into training and validation datasets. The paragraphs in the dataset are annotated for PCL.

Paragraph	Label
Call to restore hope for homeless through inquiry	1
farooqui said women 's groups were demanding fast-track courts to deal with rape and other crimes against women .	0

Table 1: Example for Sub-Task1

Paragraph	Label
the word of god is truth that 's living and able to penetrate human souls (heb. 4:12) . consider how powerful scripture is : it can change hearts , save lives from eternal condemnation , and give hope to the hopeless	Unbalanced power relations , Compassion
these poor ladies are definitely going through some traumatic issues right now , and i am asking that they come forward so that i help them ? together with women parliamentarians - to be able to heal.	Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Compassion

Table 2: Example Of Sub-Categories of PCL for Sub-Task2

The paragraphs marked positively for PCL are then annotated for seven different categories of PCL. The dataset has paragraphs in the English language and was collected by the News on Web (NoW) corpus. They queried the corpus for paragraphs using ten keywords related to vulnerable communities widely covered by the media and from the 20 English speaking countries in the corpus. Detailed information of the dataset can be found in the task description paper (Pérez-Almendros et al., 2019).

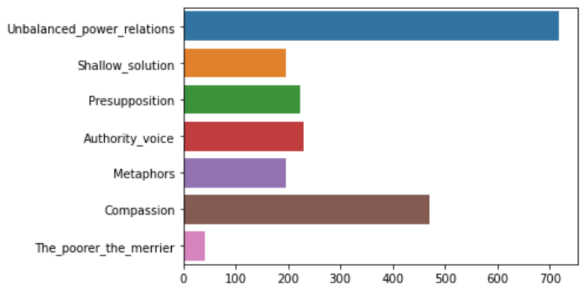


Figure 1: Frequencies of PCL subcategories

2.1.1 Sub-task1

The first sub-task is a binary classification task where given a paragraph, we have to classify whether it contains PCL. The annotation in the original paper has labels from 0 to 4. The paragraphs marked 0 and 1 are marked negative for PCL, while those marked 2 and above are marked positively. Table 1 shows positive and negative PCL examples from the dataset for sub-task 1.

2.1.2 Sub-task2

The second sub-task is a multi-label classification problem where given a paragraph, we have to classify whether it belongs to one or many of the seven subcategories of PCL. The subcategories are Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier. Table 2 shows examples for paragraphs marked for different categories of PCL.

3 System-Overview

For our solution, we have relied on using pre-trained transformer-based models like RoBERTa (Liu et al. (2019b)) which robustly optimizes the original Bidirectional Encoder Representations from Transformers(BERT) Devlin et al. (2019). It is pre-trained on much larger datasets, bigger batches and employs dynamic masking wherein a masking pattern is generated every time a sequence is fed to the model.

We also experimented with the newer Decoding-Enhanced BERT with Disentangled AttentionV3 (He et al. (2021a)), which is an improved version of the original DeBERTa (He et al. (2021b)). It leverages ELECTRA style (Clark et al. (2020)) pre-training by replacing DeBERTa's original mask language modelling (MLM) with a more sample-efficient pre-training task, replaced token detection (RTD), where the model is trained as a discriminator to predict whether a token in the corrupted input is either original or has been replaced by a generator.

As the dataset is highly imbalanced, we use two different ways to deal with the imbalance, focal loss and Weighted Random Sampling.

Model	UNB_POW	SHAL	PRES	AUTH	MET	COMP	POOR_MERR	AVG
RoBERTa $_{\text{sep}}$	19.4	15.9	9.3	5.3	13.9	11.3	9.1	12.0
RoBERTa $_{\text{ens}}{}^{*}$	15.8	24.8	10.0	**9.3**	16	11.2	14.8	14.6
DeBERTa $_{\text{ens}}$	15.7	24.8	10.0	9.2	16	11.3	14.9	14.6
RoBERTa $_{\text{WRS\_sep}}{}^{*}$	23.2	16.3	9.7	8.0	14.7	10.1	9.5	13
DeBERTa $_{\text{WRS\_sep}}$	16.3	24.9	10.3	9.1	10.0	11.6	13.8	13.7
RoBERTa $_{\text{focal\_sep}}$	10.9	**25.0**	8.1	8.8	**22.9**	**16.2**	**17.7**	**15.7**
DeBERTa $_{\text{focal\_sep}}$	15.7	24.8	10.0	9.2	16	11.2	14.8	14.5
RoBERTa $_{\text{baseline}}$	**35.35**	0.0	**16.7**	0.0	0.0	20.8	0.0	10.4

Table 3: F1 scores: Sub-Task2

3.0.1 Focal Loss

Focal loss (Lin et al. (2018)) is an improved version of Cross-Entropy Loss that tries to handle the class imbalance problem by assigning more weights to hard or easily mis-classified examples and down-weight easy examples. It results in the reduction of the contribution of easy examples. It also makes so that there is more emphasis on correcting misclassified examples.

$$L = \begin{cases} \alpha(1-p)^{\gamma}\log(p) & \text{if y} = 1 \\ (1-\alpha)p^{\gamma}\log(1-p) & \text{otherwise} \end{cases} \tag{1}$$

where p is model prediction and y is the ground truth label; α and γ are hyper-parameters, α is used to control the loss weight of positive and negative samples, and γ is used to scale the loss of difficult and easy samples. The values we take for α is 0.25 and γ is 2.0 which is the default values used in the original paper.

3.0.2 Weighted Random Sampling

Data sampling provides a way to transform a training dataset to better balance the class distribution. Data sampling techniques are helpful in cases of data imbalance as the class distribution is skewed, resulting in the models predicting the dominant class more while learning to ignore the classes with very few samples.

We use Weighted Random Sampling (WRS), which samples items from our set such that the probability of sampling item i is proportional to a given weight w_i which is equal to the class weight for the label of the i'th item.

$$w_i = 1/n_i \tag{2}$$

Here n_i is the number of items in the dataset with label i.

3.1 Sub-task1

The first sub-task is a binary classification task. We use our pre-trained BERT based transformers for this task. We experiment with either using Focal Loss or Weighted Random Sampling to deal with imbalanced data. We pass the output of the transformer model through a fully connected layer; we add a Tanh activation function with a dropout layer before passing it through our final fully connected layer, which gives us the output. We also train an ensemble of models using 5-fold cross-validation and use hard voting method to decide the final labels and combine it with Weighted Random Sampling for dealing with class imbalance

3.2 Sub-task2

For the second sub-task, we have a multi-label classification problem. We experiment with treating it as multiple binary classification tasks where for each label to be predicted, we train a separate classifier(sep). Even here, we experiment with focal loss and Weighted Random Sampling to deal with imbalanced classes.

We also train an ensemble of multi-label classifiers using 5-fold cross-validation and hard voting to decide on the final labels. As not all labels are imbalanced, we decided to use binary cross-entropy as our loss function for ensemble models for this task. We add weights to the positive samples in the loss function as done by researchers at (Gupta et al., 2021) to address the classes which do have imbalances. The formula is given below:

$$\begin{aligned} \ell(\mathbf{x},\mathbf{y}) &= -\frac{1}{Nd}\sum_{n=1}^{N}\sum_{k=1}^{d}\left[p^k y_n^k \log x_n^k + (1-y_n^k)\log(1-x_n^k)\right] \\ p^k &= \frac{1}{f^k}(|K| - f^k) \end{aligned} \tag{3}$$

Where N is the batch size, n index denotes n^{th} batch element, d is the number of classes, f stands for a vector of class absolute frequencies calculated on the train set, $\mathbf{x}$ is the output vector from the last Sigmoid layer, $\mathbf{y}$ is a vector of multi-hot encoded ground truth labels and $|K|$ is the size of the train set.

Model	Precision	Recall	F1
RoBERTa $_{\text{focal}}$	64.6	45.4	53.3
DeBERTa $_{\text{focal}}$	**68.3**	34.7	46.0
RoBERTa $_{\text{WRS}}$*	51.5	59.9	55.4
DeBERTa $_{\text{WRS}}$	50.1	65.6	56.8
RoBERTa $_{\text{ens\_WRS}}$	53.2	**67.1**	**59.4**
DeBERTa $_{\text{ens\_WRS}}$	56.2	59.6	57.9
RoBERTa $_{\text{baseline}}$	39.35	65.3	49.1

Table 4: Results: Sub-Task1

4 Experimental setup

Parameter	sub-task1	sub-task2
Dropout	0.3	0.3
BatchSize	8	8
Epochs	5	8
Learning Rate	1e-05	1e-05
Optimizer	Adam	Adam

Table 5: Hyperparameters

The dataset contains 10469 paragraphs about potentially vulnerable social groups. 9476 examples were marked negatively for PCL, while 993 were marked positively for PCL. For the second sub-task, only the 993 examples were used for training as they were marked positively for PCL.

80% of the dataset was used for training while the rest was used for validation. We only use the organisers' test set for testing out our final models. Hyper Parameters used are mentioned in table 5. Not much time was spent on hyperparameter tuning as using other previously mentioned techniques and different models gave better and more varied results.

The primary evaluation metrics used is the F1 scores. For sub-task1, precision and recall scores are also given. For sub-task2, we have individual F1 scores for each PCL subcategory along with the average F1-score. We use huggingface [1] library for our transformer models implemented in PyTorch [2].

[1]Transformers,v4.16.2,`https://huggingface.co/docs/transformers/index`

[2]PyTorch, v1.10.2, `https://pytorch.org/`

The models were trained on 4 GeForce RTX 2080 Ti GPUs.

5 Results And Discussion

The results from all our experiments conducted for sub-task1 and sub-task2 can be seen in Tables 4 and 3, respectively. The models submitted in the evaluation phase are marked * in the tables, but we have shown results from all our experiments. We experimented with several models and techniques during the development and evaluation phases. We use the F1 score to judge our models, which is also the official metric. We ranked 23rd on the first task and 36th on the second task on our submitted models. We achieved better results on other models for both tasks, and the results are shown in their tables, respectively.

We see that all methods, namely focal loss, Weighted Random Sampling(WRS) and ensembling performed better than the baseline model. The 5-fold cross-validation, hard voting ensemble model with WRS achieves the best F1-score and Recall score for sub-task1, more than the models where only WRS is applied.

For the second task, we see the best average score from RoBERTa model trained separately (sep) for each label with focal loss to achieve the best average F1 score. Focal loss performs poorly on Unbalanced power relations, which has the highest number of positive samples (716 out of 993) and performs better on imbalanced labels having a lower number of positive samples like Metaphors and Poorer The Merrier having 197 and 40 positive samples out of 993 respectively.

6 Conclusion

This paper presents and describes our solution system for the SemEval2022 Task4: Towards Patronizing and Condescending Language Detection. We have applied BERT based pre-trained language models RoBERTa and DeBERTa with hard voting ensembling techniques along with techniques to deal with imbalanced datasets like focal loss and Weighted Random Sampling. Our submitted solutions scored F1 scores of 0.5539 and 0.1456 for the two sub-tasks, respectively.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel

Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

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

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2018. Supervised learning of universal sentence representations from natural language inference data.

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

Kshitij Gupta, Devansh Gautam, and Radhika Mamidi. 2021. Volta at SemEval-2021 task 6: Towards detecting persuasive texts and images using textual and multimodal ensemble. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1075–1081, Online. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-enhanced bert with disentangled attention.

Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. 2019. FERMI at SemEval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 70–74, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2018. Focal loss for dense object detection.

Ping Liu, Wen Li, and Liang Zou. 2019a. NULI at SemEval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 87–91, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2019. Cardiff University at SemEval-2019 task 4: Linguistic features for hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 929–933, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Felix&Julia at SemEval-2022 Task 4: Patronizing and Condescending Language Detection

Felix Herrmann & Julia Krebs
University of Regensburg
Regensburg, Germany
felix-georg.herrmann@stud.uni-regensburg.de, julia.krebs@stud.uni-regensburg.de

Abstract

This paper describes the authors' submission to the SemEval-2022 task 4: Patronizing and Condescending Language (PCL) Detection. The aim of the task is the detection and classification of PCL in an annotated dataset. The authors of this paper worked on two different models with finetuned hyperparameters focusing on number of epochs, training batch size, evaluation batch size, gradient accumulation steps and learning rate. The authors submitted one RoBERTa model and one DistilBERT model. Both systems performed better than the random and RoBERTA baseline given by the task organizers. The RoBERTA model finetuned by the authors performed better in both subtasks than the DistilBERT model.

1 Introduction

With the rise of social media, online hate speech has skyrocketed and has posed a problem for social media platforms: How can the giant number of messages on social platforms be surveilled, so that hateful comments can be reported and deleted? The answer to this is hate speech detection, a discipline within Natural Language Processing that has become increasingly popular and successful in recent years.

However, apart from hate speech, there is also other harmful language that should be studied. This is what the research by Perez-Almendros, Espinosa-Anke and Schokaert dives into. They collected an annotated dataset that focuses on a type of harmful language that is not so easily detected: patronizing and condescending language (PCL). The researchers describe PCL as follows: "An entity engages in PCL when its language use shows a superior attitude towards others or depicts them in a compassionate way." (Perez-Almendros et. al., 2020)

One of the difficulties in detecting PCL as opposed to openly hateful speech, is that persons who use PCL often do not intend to do harm, but instead want to support the groups that they name. PCL is usually aimed at vulnerable communities which makes the detection of PCL even more important.

The SemEval 2022 task 4 competition is based on this research by Perez-Almendros et. al: the detection of PCL. In two subtasks, one a binary classification and one a multi-label classification, the participants of the SemEval competition were tasked to submit up to two models.

The authors of this paper participated in both subtasks and propose a finetuning based approach on a pre-trained RoBERTa language model. Two models were submitted to the task organisers, one RoBERTa model and one DistilBERT model. The RoBERTa model performed better in both subtasks.[1]

In this paper, the authors will first describe the task set out by the SemEval 2022 competition in sections 2. In section 3, the authors explain the relevance of related work to this topic. Section 4 gives an overview over the BERT model. In the subsequent section, the two pre-trained models are presented. In the section experimental setup, the two different finetuned models are described in detail. Section 7 presents the results of the competition and section 8 the conclusion.

[1] The code can be found here: https://github.com/julia-ecrevisse/SemEval2022

2 Task description

This research paper contributes to task 4 of the SemEval 2022 competition[2]: PCL detection. The starting point of the research task is the paper "Don't Patronize Me! An annotated Dataset with Patronizing and Condescending Language Towards Vulnerable Communities" (Perez-Almendros et al., 2020). The researchers provide an annotated dataset with paragraphs taken out of news articles from English speaking countries[3] where vulnerable communities are mentioned. The following communities are included: disabled, homeless, hopeless, immigrant, in need, migrant, poor families, refugee, vulnerable and women. The dataset includes a total of 10,637 paragraphs extracted from the News on Web corpus. 3,554 of the paragraphs had been labeled as PCL by the annotators.

The aim of the SemEval task is 1) to identify which paragraphs include PCL and 2) if the paragraph includes PCL which category or categories it belongs to. The two subtasks are described as follows:

Subtask 1: Binary classification. Given a paragraph, a system must predict whether or not it contains any form of PCL.

Subtask 2: Multi-label classification. Given a paragraph, a system must identify which PCL categories express the condescension. There are seven different categories: Unbalanced power relations, shallow solution, presupposition, authority voice, metaphor, compassion and the poorer, the merrier.

3 Related Work

While other areas related to hate speech and hateful language have been a focus of NLP research in recent years, the research into PCL is still limited. However, there are a few researchers that have delved into this discipline from different angles.

Wang and Potts (2019) discuss condescending language and its detection in their research and also point out that high quality data for this kind of language detection is still limited. They introduce the dataset "Talkdown" which is a data set from the social media platform Reddit. Their approach concentrated on BERT baseline models and the researchers concluded that condescending language is highly connected to context (Wang and Potts, 2019).

Taking a different angle, Sap et al. (2020) write about the social and power implications of language. They claim that by using language with, e.g. social bias, stereotypes and prejudices are reinforced. Their collected data stems from different social media sites. They state in their results that previously successful models can categorize social bias relatively well, but they have difficulties classifying the social bias frames which were developed by the authors (Sap et al., 2020).

BERT

The state of the art in natural language processing is significantly characterized by architecture-based transformer-encoder models called BERT (Bidirectional Encoder Representation from Transformers) (Peters et al., 2018).

BERT's architecture relies on a two-part training process, a pretraining using unlabeled text corpora and a subsequent test run using labeled data (Develin et al, 2019). Here, BERT models do not use any decoder layers (Rothman, 2021) and fully rely on the encoder structures developed by Devlin et al. (2018). Here, these models are divided into base and large.

In order not to have to perform the training process, which is time-consuming, for each new task, especially pre-trained models are made available on platforms such as Huggingface. These can be adapted to the respective requirements by means of fine-tuning.

This previous research into hate speech and hateful language as well as the research into BERT model provides the basis of this paper. The challenge of condescending or patronizing language as opposed to hate speech is that condescending or patronizing language is often more difficult to detect as it is often not on purpose or not as obvious. Still, the steps taken by previous authors show a way on how to tackle these challenges.

4 System overview of pretrained models

In the paper "Don't Patronize me!" which is the starting point of the SemEval Task, the researchers

[2] https://sites.google.com/view/pcl-detection-semeval2022/home?authuser=0

[3] News on Web Corpus: https://www.english-corpora.org/now/

point out that an NLP model with BERT achieves the best results. More specifically RoBERTa performs slightly better than DistilBERT and BERT-base (Perez-Almendros et. al., 2020).

4.1 RoBERTa model

The RoBERTa model stands for Robustly Optimized BERT Pretraining Approach and is based on the basic architecture of BERT, which, however, has been insufficiently trained for the subtasks to be handled. RoBERTa increases performance by improving the pretraining processes. To generate this progress, the number of pretraining transformers is increased. An example of this is the omission of WordPiece tokenization (Song et al., 2021) and the associated structuring at the byte-level byte pair encoding level (Rothman 2021).

For both subtasks, the RoBERTa base model was used, which consists of 12 encoders. In contrast, the RoBERTa large consists of 24 (Nester, 2022). The decisive factor for the choice of the model was the improved modification of the hyperparameters in the fine-tuning, which were our primary focus. In addition, the large amount of data used to pre-train RoBERTa was a reason for the choice. From this, we aimed to improve the generalization of the responding capability compared to the conventional BERT model (Delobelle et al., 2020).

Apart from the RoBERTa baseline, another RoBERTa model from the transformers library was tried (Devlin et al, 2019): *xlm-roberta-base* (Huggingface), with the following results: Subtask 1 F1 score: 0.466, Subtask 2 average F1 score: 2: 0.110. As this baseline performed worse than the original baseline, the authors decided to continue with the original RoBERTa baseline.

4.2 DistilBERT model

DistilBERT is often called the "faster and cheaper" version of BERT. Using up massive amounts of data is expensive and also wasteful. In their research Sanh, Debut, Chaumond and Wolf describe that they were able to "reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster" (Sanh et. al., 2020). That is why it makes sense to see if there are promising results with a DistilBERT model also for this SemEval task.

Two different DistilBERT models from transformers were tried for the two subtasks: *distilbert-base-uncased* and *distilbert-base-uncased-finetuned-sst-2-english*. Both were first tried as a baseline. Comparing both models, the *distilbert-base-uncased* showed better results for the baseline, especially for subtask 2 (see table 1),

Baseline	***distilbert-base-uncased***	***distilbert-base-uncased-finetuned-sst-2-english***
Subtask 1 F1 score	0.488	0.433
Subtask 2 average F1 score	0.121	0.060

Table 1: DistilBERT baselines F1 scores

that is why it was decided to continue with this model in the following fintetuning stage. For the multilabel classification of subtask 2 the most promising predictions of the *distilbert-base-uncased* baseline were in the categories unbalanced power relations, presupposition, and compassion.

The *distilbert-base-uncased-finetuned-sst-2-english* model only showed a good F1 score for the category unbalanced power relations, whereas all other categories had a F1 score of 0.0.

5 Experimental setup

Fine-tuning was performed on Google Colab. The authors used the train and test set as it was provided by the task organizers.

5.1 RoBERTA model

Subtask 1

The experimental setup used was the RoBERTa baseline, which was provided by the Semeval 2022 team for the task. For fine-tuning, 9 different hyperparameters were run in different combinations with the model. For this, the F1 score was set as a benchmark and depending on the development, the hyperparameter was pushed to its limit. Using the example of the epochs, a steady improvement of the F1 score could be observed up to level 5, before it deteriorated again. If such a limit of a parameter was reached, it could be set as default and the tuning could be supplemented by another one. This resulted in a combination of 5 epochs, a learning rate of 3e-5, an evaluation batch size of 32, a training batch size of 16 and gradient accumulation steps of 2.

In addition, the weight_decay, adam_epsilon, max_grad_norm and the warmup_steps were implemented at different levels. However, these led to a deterioration of the precision and the F1 score in all variants. However, it is worth mentioning that there was a significant improvement of the recall by using warmup_steps= 3500. Without further illuminating this direction, a recall of 0.804 was achieved, which cannot yet be called a maximum.

Subtask 2

The findings from Subtask 1 were to be applied to multi-labeling. For this purpose, both the RoBERTa-baseline model and the same hyperparameters were adopted. Again, the different hyperparameters were pushed to maximum improvement with the goal of obtaining the highest possible mean value of the individual F1 scores of the labels. Epochs could be increased to 20 until the peak was reached. All other efforts to obtain an optimized result away from the epochs or in combination with them were unsuccessful. The mean value reached its possible optimum after 20 epochs with a result of 0.258.

5.2 DistilBERT model

For the DistilBERT model, three hyperparameters were tried by the authors: number of trained epochs, the evaluation batch size and the training batch size. The number of trained epochs has been adapted from 1 to 20 epochs, while for the evaluation and training batch size 16, 32 and 64 was used respectively.

	Subtask 1	Subtask 2
Number of trained epochs	5	20
Evaluation batch size	32	64
Training batch size	32	16

Table 2: Hyperparameters used for DistilBERT model

The best results were achieved with the hyperparameters described in Table 2.

6 Results

An overview of the results can be found in table 3 and 4.

	RoBERTa model	DistilBERT model	Baseline RoBERTa
Precision	0.401	0.357	0.394
Recall	0.773	0.640	0.653
F1	0.528	0.459	0.491

Table 3: Results subtask 1

F1 scores	RoBERTa model	DistilBERT model	RoBERTa baseline
unbalanced power relations	0.366	0.352	0.354
shallow solution	0.351	0.345	0
presupposition	0.176	0.2	0.167
authority voice	0.221	0.163	0
metaphor	0.211	0.095	0
compassion	0.285	0.271	0.209
the poorer, the merrier	0.167	0.0	0
average	0.254	0.204	0.104

Table 4: Results for subtask 2

6.1 RoBERTa model

In Subtask 1 as well as in Subtask 2 significant improvements could be achieved compared to the baselines. This is reflected in the results of the test data set as well as the final data set. The finetuning improved the initial value (F1: 0.483) of the RoBERTa baseline in Subtask 1 by about 0.05 to 0.547. A significant increase of about 0,10 (0.441 to 0.350) was observed in the Precision section. In the Competition a similar value was achieved with, so that the robustness of the system and its configuration is given. An overfitting could not be determined. The results were ranked 31st with a Precision of 0.401, a Recall of 0.773 and the F1 0.528.

In Subtask 2, RoBERTa nearly doubled the mean F1 score from baseline 0.134 to 0.258. On the official leaderboard, an almost identical value of 0.2536 was achieved, so that here, as in Subtask 1, no overfitting prevailed and the robustness of the system was confirmed. The RoBERTA model performed best in the categories “unbalanced power relations” and “shallow solution” (see table 4). The worst results were reached for the categories “presupposition” and “the poorer the merrier”.

6.2 DistilBERT model

A comparable scenario occurred when running the DistilBERT model with the test data. Compared to the Random as well as the RoBERTa Baseline, the results significantly improved in both subtasks. In the comparison, however, the Random Baseline is used first. An increase from 0.174 (Random) to 0.512 could be achieved by applying the above mentioned hyperparameters. The RoBERTA baseline was thus also exceeded by approximately 0,30. However, in the official results, the score was not repeated and was 0.459 (see table 3).

In Subtask 2, the Random Baseline result with the test data of 0.055 for the mean of the F1 scores was almost tripled to 0.146. However, compared to the RoBERTa baseline of 0.134, only a small improvement was observed.

Surprisingly, the model performed much better with the official data and thus achieved its maximum value of 0.203. Also, for the DistilBERT model, the categories with the best results were “unbalanced power relations” and “shallow solutions” (see table 4). The DistilBERT model performed worse than the RoBERTA model in most categories, except “presupposition”. The system performed worst for the category “the poorer, the merrier” compared with the other categories.

7 Conclusion

The purpose of this work was to identify PCL in texts using NLE. For this purpose, existing text classification models were used and adapted to the task by fine tuning. The goal was to achieve the highest possible F1 scores by the model, which is equivalent to the percentage detection of condescending terms. A complete detection of all these terms was not achieved, but in the binary classification with RoBERTa a score above 0.52 could be obtained. The multi classification was the bigger challenge and could only finish with a score around 0.250.

Despite these results, a basis, if not an improvement, has been created for future work. The hyperparameters which were used can already be set to default in the next works at the beginning and thus increase the speed of the development process for researches in PCL detection by BERT models. More research could also be done into the different categories of PCL. There were categories that consistently performed better than others (unbalanced power relations and shallow solution), while the systems had more difficulties with other categories (especially the poorer, the merrier).

There are still a lot of directions that the research into PCL detection can continue. As stated above, PCL detection in NLE is an emerging field that would benefit from further research. The task is more difficult than traditional hate speech detection as PCL is often more subtle. However, PCL can contribute to repeating and furthering stereotypes and discrimination, especially among vulnerable communities and therefore the research into PCL detection systems is a vital endeavor.

Acknowledgments

We would like to thank our professor Dr. Udo Kruschwitz at the University of Regensburg for encouraging us to take part in the SemEval 2022 evaluation and supporting us during the process. We would also like to thank our study colleagues Aenne and Amela with whom we exchanged ideas and approaches throughout the timeframe of the SemEval 2022 competition.

References

Delobelle P., Winters T., and Berendt B.. 2020. RobBERT: a Dutch RoBERTa-based Language Model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3255–3265, Online. Association for Computational Linguistics

Devlin J., Chang M., Lee K., and Toutanova K.. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nester, J.. 2022. Rassistische Sprache mit BERT erkennen: Eine Untersuchung am Beispiel deutscher Plenarprotokolle. Universität zu Köln.

Perez-Almendros, C., Espinosa-Anke, L., Schockaert, S.. 2020. Don’t Patronize Me! An Annotated Dataset with Patronizing and Condescending

Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics, pages 5891–5902*, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Peters M., Neumann M., Iyyer M., Gardner M., Clark C., Lee K., and Zettlemoyer L.. 2018. Deep contextualized word representations. In *NAACL 2018*.

Rothman, D.. 2021. Transformers for natural language processing: build innovative deep neural network architectures for NLP with Python, Pytorch, TensorFlow, BERT, RoBERTa, and more. Birmingham, Mumbai 2021

Sanh V., Debut L., Chaumond J., Wolf T.. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108.

Sap M., Gabriel S., Qin L., Jurafsky D, Smith N., and Choi Y.. 2020. Social bias frames: Reasoning about social and power implications of language. arXiv preprint arXiv:1911.03891.

Song X., Salcianu A., Song Y., Dopson D., and Zhou D.. 2021. Fast WordPiece Tokenization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2089–2103, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wang Z., Potts C.. 2019. Talkdown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing.*

MS@IW at SemEval-2022 Task 4: Patronising and Condescending Language Detection with Synthetically Generated Data

Selina Meyer, Maximilian Schmidhuber and **Udo Kruschwitz**
Chair for Information Science
University of Regensburg, Germany
(selina.meyer, udo.kruschwitz)@ur.de
maximilian.schmidhuber@stud.uni-regensburg.de

Abstract

In this description paper we outline the system architecture submitted to Task 4, Subtask 1 at SemEval-2022. We leverage the generative power of state-of-the-art generative pretrained transformer models to increase training set size and remedy class imbalance issues. Our best submitted system is trained on a synthetically enhanced dataset with 10.3 times as many positive samples as the original dataset and reaches an F1 score of 50.62%, which is 10 percentage points higher than our initial system trained on an undersampled version of the original dataset. We explore possible reasons for the comparably low score in the overall task ranking and report on experiments conducted during the post-evaluation phase.

1 Introduction

Task 4 of SemEval-2022 focuses on the detection of patronising and condescending language (PCL) in news (Pérez-Almendros et al., 2022). PCL in popular media and news sources is detrimental to an emancipated and equal society, as it is usually targeted towards minorities and socially disadvantaged communities, often in an unsuccessful attempt to show solidarity (Perez Almendros et al., 2020). PCL has the potential to strengthen existing stereotypes by representing minorities either as passive entities to be pitied and supported, thus taking away their agency and focusing on their vulnerabilities or praising members of vulnerable groups for everyday achievements simply because of their background (Nolan and Mikami, 2013). In contrast to hate speech, PCL is usually subtle, well intentioned, and free of discriminatory phrases or racial slurs, which makes it an interesting Natural Language Processing (NLP) problem.

In other domains with more discriminatory classes such as hate speech detection, generative models have recently become increasingly popular and successful as a tool to increase classification performance (Wullach et al., 2021; Anaby-Tavor et al., 2020). In our contribution to the shared task, we explored to what extent this approach is feasible for the presented use case, where classification of a text sample is less distinct and often relies on world knowledge (Perez Almendros et al., 2020). The dataset provided for the task was fairly small, with less than 10% of the data belonging to the positive class. We thus enhanced the original dataset in two ways for our system runs:

- balancing the dataset by generating only PCL samples
- increasing overall dataset size, by generating an equal amount of PCL and non-patronizing (nPCL) samples

We generally followed the approach used by Wullach et al. (2021) and initially fine-tuned a BERT classifier on the original dataset. We then fine-tuned GPT-3 (Brown et al., 2020) and generated samples of PCL and nPCL which were classified using our fine-tuned system. Samples for which the BERT classification did not correspond to the intended output were discarded. We then fine-tuned a new BERT instance with the modified dataset PCL*enhanced* including the synthetic data. Although our system only ranked middle field in the competition, both classifiers trained on the modified datasets improve our initial classifier trained on the original dataset by multiple percentage points. We conclude that this approach does add value to classification, even in cases where the distinction between the positive and the negative class relies on subtleties. The code described in the following as well as the synthetic data used for the modification of the original dataset is available on Github[1].

[1] https://github.com/khaliso/MS-IW-at-SemEval-2022-Task-4

Text	Class
Meanwhile "throughout this island, the high level of suicide is terrible and terrifying. "As Christians" we can give hope, where a person feels only darkness and hopelessnes," he said	PCL
As the house prices go up, so do rents , and the pohara poor families ca n't afford to live. Those who own houses, and are only just making it through, will be rated out of their homes	nPCL

Table 1: Examples of PCL and nPCL in the DPM.

2 Background

We participated in Subtask 1 of the competition, which entailed the binary classification of news paragraphs as either patronizing or not patronizing. Basis for the task was the *Don't Patronize Me! dataset* (DPM) (Perez Almendros et al., 2020), which contains 10,469 paragraphs of annotated data from 20 English news sources. While all paragraphs include references to potentially vulnerable groups, only 993 are examples of patronising speech. The dataset included meta-information about the country each paragraph was published in, an article id, a keyword indicating which vulnerable group is addressed, and a label ranging from 1 to 4, where 0 and 1 are treated as non-patronizing and 2 to 4 as patronizing. The task organizers define PCL as often unconscious, subtle and subjective ways in which the speaker conveys a superiority "concealed behind a friendly or compassionate approach towards the situation of vulnerable communities" (Perez Almendros et al., 2020). They explicitly exclude hate speech and discriminatory speech from PCL, making it harder to be identified not only by NLP-systems, but also by humans. We include examples of both classes in Table 1.

Transformer-based generative models such as GPT (Radford et al., 2018) and its successors have become prevalent in various NLP tasks. For instance, Liu et al. (2021a) explored the idea of synthetically constructing benchmark datasets to concur with existing benchmarks such as SQuAD, while Zhang et al. (2018) showed that a fine-tuned GPT model can accurately mimic the personal conversation style of an individual, leading to improvements in the Persona-Chat dataset.

Another increasingly popular use case is the generation of data on tasks with small labeled corpora to synthetically increase dataset size in order to train better performing classifiers. Dekker and van der Goot (2020) used synthetic data for lexical normalisation, while other researchers employed such data to train question answering models (Puri et al., 2020). Even in maths, researchers have proposed ways of creating synthetic theorems (Firoiu et al., 2021). Wullach et al. (2021) used GPT-2 (Radford et al., 2019) for their approach to hate speech detection. Their datasets were small to medium sized (6-53k labelled examples) and highly unbalanced, with as little as 1-6k hate speech samples per dataset. They created three mixed datasets containing 10k, 80k and 240k synthetic samples respectively, as well as 80% of the original datasets. The classification models trained on the largest created dataset outperformed those trained on the smaller datasets in most cases. Anaby-Tavor et al. (2020) generated data using GPT and improved sentence-level topic classification on three datasets, ranging from 4.2k to 17k entries. Wullach et al. (2021) and Anaby-Tavor et al. (2020) fine-tuned the respective GPT models on relatively small datasets, and find statistically significant improvements on classifier performance through incorporating synthetic data in the datasets used for fine-tuning classifiers.

While GPT and GPT-2 were trained on 117M and 1.5B parameters respectively, GPT-3 models were trained on up to 175B parameters (Radford et al., 2018, 2019; Brown et al., 2020). As it has been shown that an increase in model size systematically leads to improvements in text synthesis as well as common downstream tasks (Brown et al., 2020), GPT-3 is likely to produce higher quality and more natural sounding data than its predecessors. We thus expect GPT-3 generated data to have an even greater impact on performance in intricate language classification tasks such as PCL detection. We know of only few other research teams which used GPT-3 in their experiments, for instance to search for more suitable prompts for Natural Language Understanding (NLU) tasks (Liu et al., 2021b) or using prompts for few-shot generation (Yoo et al., 2021). Both achieved strong results on classification benchmarks.

While using foundation models for data generation has the potential to increase the power of language models and mitigate the data scarcity prob-

Dataset	PCL	nPCL	% PCL	$F1_{pos}$	Pre_{pos}	Rec_{pos}
DPM	993	9476	9.5	–	–	–
$DPM_{undersampled}$	804	804	50	40.74%	27.2%	**81.07%**
$DPM_{enhanced}$	10242	16937	37.7	**50.62%**	51.15%	50.10%
$DPM_{enhancedPos}$	7880	7580	49	46.76%	**54.39%**	41.01%
Official Baseline RoBERTa	–	–	–	*49.11%*	*39.36%*	*65.30%*
Post-Evaluation						
$DPM_{enhancedUnfiltered}$	24984	31886	43.93%	42.28%	43.62%	41.01%
$DPM_{enhancedPosUnfiltered}$	24984	7580	76.72%	44.07%	43.07%	45.11%

Table 2: Overview of the datasets used for fine-tuning as compared to the original dataset and test classification metrics.

lem prevalent in many NLP fields (Budzianowski and Vulić, 2019), this also bears potential risks not yet fully explored. For instance, past research showed that GPT-3 is biased in some cases, and that its defects are inherited by downstream models (Bommasani et al., 2021). Similarly, Bender et al. (2021) note, that the widespread application of foundation models carries a cost - both monetary and ethically. Thus, this approach's ethical implications should be investigated more thoroughly in future work.

3 System Overview

To generate the synthetic data, we used GPT-3's Curie model. Curie has about 13B unique parameters, while Davinci has about 175B. Although Davinci performs significantly better on a number of NLP tasks than Curie, we chose Curie, as it is more financially viable than the larger model, while retaining a comparatively strong performance (Brown et al., 2020).

For fine-tuning, we split the dataset into PCL and nPCL data and modified it to meet the API's requirements. As the API requires a prompt-completion pairing, the prompt was set to be empty (") and the completion contained the data sample. Afterwards, two GPT-3 Curie instances were fine-tuned on the PCL and nPCL data, respectively. We thus created two models, one to generate PCL and one for nPCL phrases. Following Wullach et al. (2021), we called the models with an empty (") prompt in the pipeline for synthetic data generation and the default parameters. We set *max_tokens* to the rounded mean length of the samples in the original dataset (60 for PCL and 54 for nPCL). With each iteration, we generated the maximum number samples (128), resulting in a total of 24.321 synthetic phrases by the nPCL model and 24.197 by the PCL model.

Like (Wullach et al., 2021), we classified all synthetic samples after generating the data. We used an initial baseline classifier and discarded all samples where the intended and predicted class did not match. Due to the high class imbalance of the original dataset, we randomly undersampled the negative class to the size of positive samples for training of the baseline classifier. We fine-tuned $BERT_{base}$-cased (Devlin et al., 2018) across three epochs using a learning rate of 1e-5 on the undersampled dataset. Since the synthetically generated data consisted solely of text for each label, we did not use any of the meta-information or context provided in the dataset and fine-tuned solely on text and labels. In the future, it might be useful to take meta-information into account for text generation.

39% of the generated PCL samples were classified as such by the baseline classifier, whereas 85,5% of generated nPCL samples were classified as nPCL. We explain this with the much larger sample size of nPCL in the DPM allowing the GPT-3 pipeline to generate better suited data. Based on the predictions, we created two enhanced datasets: For $DPM_{enhanced}$, we added a similar amount of synthetic PCL (9448) and nPCL (9357) samples to the DPM. For $DPM_{enhancedPos}$, we added 7086 PCL samples to balance the original dataset. For a comparison of sample sizes and share of PCL in the DPM and the different datasets used for fine-tuning see Table 2. On each of the enhanced datasets, we trained a $BERT_{base}$-cased instance the same way as our initial classifier. We submitted the classifier trained on $DPM_{enhancedPos}$ for our first and the classifier trained on $DPM_{enhanced}$ for the second run.

Intended	Text	Pred
coherent samples		
PCL	*so gao becomes emotional as he reflects on the thousands of homeless children he has come across during his decades long football career – most of them growing up without a father figure in their lives. "the kids today may be our future But there is no future for the kids today if we don't have*	PCL
PCL	*English and humanities teacher blowsy dilworth decided some kids in her georgian village needed more than a pack of cards to play baseball with – they needed an ancestral field. so she quested across the border to find some land for her students, and this week she opened a playfield on that*	nPCL
nPCL	*understandably , many sri lankans look at india with wariness, if not hostility. foster father pair of us destroyed an Eldorado of a country. thousands of families were made homeless and live on the streets today. on november medium*	PCL
nPCL	*africa has the largest block of 2017 retirements sufferance among all regions , with recent precedent of expenses course and after-inheritance taxed deaths , show disclosures by top investment funds in the united states . on the whole , fund seniors are think about leaving equities*	nPCL
incoherent samples		
PCL	*Subject : Crying Monkey Fortunetelling video 1 'sunday 's focus is on a widower , otis reigns , who recites a fortune to his 11 children while they weep , a performance that has attracted millions of views online . producer and director rebeca Ramirez says she*	PCL
PCL	*Crazy Horse 3 is aNATIVEpi agt sanctioned 51 majorityhare partnership firm jointly owned and managed by a group of indian stipendiaries and based in vancouver , b. c. agt Crazy Horse 3 is an eyaculofemoral orifice created for the purpose of*	nPCL
nPCL	*policy to homeseekers , students and the vulnerable........................transparency and public control of thebiologist!!!!!!!!!!*	nPCL
nPCL	*seems like coast is in need of some life. you could say that again about their women's Water Polo team. the t Vernons Wyr Kangas athletes recent 4ANPerformers cabinet hardwood men's schools100 result in need of some inspiring coast women*	PCL

Table 3: Examples of patronizing and non-patronizing generated data and its classification with the baseline classifier. Samples where intention and prediction matched were used for DPM*enhanced* and DPM*enhancedPos*, regardless of whether they are coherent or not. All synthetically generated data is available on github.

4 Results and Discussion

The evaluation metric used for ranking in the task was F1 over the positive class. Our baseline classifier reached an F1-score of 40.74% on the test set provided by the task organisers after the end of the competition's evaluation phase. Although it had a high recall of over 80%, precision was very low, leading to a suboptimal F1-score. The classifier trained on $\text{DPM}_{\text{enhanced}}$ scored almost 10% higher than the initial classifier, but had neither the highest recall, nor the highest precision of the three classifiers trained before the post-evaluation phase. This was suprising, as we initially expected the classifier trained on $\text{DPM}_{\text{enhancedPos}}$, which was the larger balanced dataset out of the three, to perform best. This leads to the assumption that with synthetic data, sheer amount might be more important than balancing out the dataset.

Although in the official task scoring, our system trained on $\text{DPM}_{\text{enhanced}}$ ranked in place 41 of 78 and surpassed the official baseline (fine-tuned RoBERTa) by only about 1%, we note that using both synthetically enhanced datasets led to a boost in performance compared to our initial classifier. This might seem surprising, especially considering the low performance of the initial classifier used to filter the GPT-generated data. In the post-evaluation phase, we repeated the experiments from our two system runs without previous filtering of the GPT-output, to explore the role of the initial classifier in our system's performance. Neither $\text{DPM}_{\text{enhancedUnfiltered}}$ nor $\text{DPM}_{\text{enhancedPosUnfiltered}}$ led to better performance than $\text{DPM}_{\text{enhanced}}$. Thus, using a baseline classifier for filtering seems to be the most sensible option when working with synthetic data, regardless of its performance strength. We report on detailed classification results in Table 2. Since our baseline system did not perform very well in terms of classification, future work should first and foremost focus on improving it. The baseline system forms the basis of our approach and classification errors at this stage are likely to significantly lower the usefulness of the synthetic data.

We also looked at some of the synthetic data generated by GPT-3. Both for PCL and for nPCL, the generated samples were not always coherent on a semantic level and the occurrence of incoherent text appeared to be more common in the nPCL condition. However, it seems like coherence did not impact classification, as in both cases incoherent synthetic samples could be found in the final dataset (see Table 3).

We also found a lot of text in languages other than English, possibly because of the small size of the dataset in comparison to the vast amount of training data used to create GPT-3. We expect that filtering out such samples would increase performance further. In addition, basic data-cleaning of the synthetic data before classification might be in order. Both of this could potentially be achieved by only using data samples for which a confidence score above a certain percentage (i.e. 70%) is returned in classification. Another approach might be using an unrelated dataset to filter out all synthetic data unrelated to the task at hand. In the context of PCL detection, this could help discard generated data that is not related to vulnerable groups.

The approach of using an empty prompt (") while fine-tuning the models is debatable, because the prompt is such a powerful tool (Yoo et al., 2021; Liu et al., 2021b) and should probably be utilized. A possible approach would be to train a single model on both PCL and nPCL data, and put PCL/nPCL information in each samples' prompt. The currently unused meta-information of the dataset could also be incorporated, possibly causing additional improvements in the quality of the generated data.

5 Conclusion

We described our system submitted to Task 4, Subtask 1 of SemEval-2022. Although the system's performance did not score highly on the overall leaderboard, ranking 41st place, incorporating synthetic data in the original training set still boosted performance by up to 10% compared to our initial baseline system, which leads to the assumption that pairing this approach with more sophisticated classification systems has some potential to increase classification performance significantly. We derive some lessons learned from the presented experiments as follows:

- Using a baseline classifier to filter the synthetic data after generation seems to be essential.
- The size of the additional data seems to be more important to increase performance than balancing the data.
- Further data cleaning and filtering might be necessary to improve classification performance.
- Synthetic data leads to better performance, even if it includes a lot of incoherent samples and the baseline classifier has low performance.

In the future, we plan to improve the baseline classifier and explore different data cleaning and filtering techniques, such as using confidence scores returned by the classifier for our data selection, using unrelated datasets to filter whether a data sample fits in the task-specific domain or making use of prompts during GPT-3 fine-tuning and data generation. Exploring other augmentation strategies such as back-translation or synonym replacement of either the original data or the generated samples might further increase classification performance.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.

Kelly Dekker and Rob van der Goot. 2020. Synthetic data for english lexical normalization: How close can we get to manually annotated data? In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6300–6309.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Vlad Firoiu, Eser Aygun, Ankit Anand, Zafarali Ahmed, Xavier Glorot, Laurent Orseau, Lei Zhang, Doina Precup, and Shibl Mourad. 2021. Training a first-order theorem prover from synthetic data. *arXiv preprint arXiv:2103.03798*.

Nelson F Liu, Tony Lee, Robin Jia, and Percy Liang. 2021a. Can small and synthetic benchmarks drive modeling innovation? a retrospective study of question answering modeling approaches. *arXiv preprint arXiv:2102.01065*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

David Nolan and Akina Mikami. 2013. 'the things that we have to do': Ethics and instrumentality in humanitarian communication. *Global Media and Communication*, 9(1):53–70.

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

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

Tomer Wullach, Amir Adler, and Einat Minkov. 2021. Fight fire with fire: Fine-tuning hate detectors using large samples of generated hate speech. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4699–4705.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Team LEGO at SemEval-2022 Task 4: Machine Learning Methods for PCL Detection

Abhishek Kumar Singh
GLA University, India
sabhishek.kumar166@gmail.com

Abstract

In this paper, we present our submission to the SemEval 2022 - Task 4 on Patronizing and Condescending Language (PCL) detection. We approach this problem as a traditional text classification problem with machine learning (ML) methods. We experiment and investigate the use of various ML algorithms for detecting PCL in news articles. Our best methodology achieves an F1- Score of 0.39 for subtask1 with a rank of 63 out of 80, and F1-score of 0.082 for subtask2 with a rank of 41 out of 48 on the blind dataset provided in the shared task.

1 Introduction

The explosion of social media in recent years also enables increasing the number of patronizing and condescending language (PCL). Patronizing and condescending language depicts apparently kind or helpful behavior but betraying a feeling of superiority on others. Previously, harmful behavior in language for example, hate speech (Fortuna and Nunes, 2018), offensive language (Razavi et al., 2010), fake news (Oshikawa et al., 2018), rumor propagation or misinformation (Zhou and Zafarani, 2020), and many others has been widely studied in NLP, PCL has been a neglected area of study until very recently.

Identifying PCL is hard even for humans because it is subjective and subtle. For instance, one might find condescending something which another person might consider an objective portrayal of a situation or some people might not see the harm in describing how those in a privileged position donate their remainings to those who need them. Also, we would expect a member of a so-called vulnerable community to feel more patronised than one person who does not belong to such group while reading how others refer to them.

The goal of SemEval 2022-Task 4 is to design a system to detect whether or not the text contains any form of PCL and furthermore, identify which PCL category expresses the condescension. The organizers provided two datasets, one annotated based on the intensity of PCL and other with the PCL categories. We approach this problem using various machine learning approaches using the linguistics features.

Class	Nb of Samples
PCL	9476
Non-PCL	993

Table 1: Dataset Statistics Task 1

The structure for the rest of the paper is as follows. Section 2 describes a background about the dataset. Section 3 describes the experimental setup of our experiments. It involves pre-processing, feature engineering, implementation details for all the respective ML models. Section 4 describes the results and discussion for both the subtask. And lastly, in Section 5, we concluded the paper and suggest ideas for future research.

2 Dataset

The dataset (Perez-Almendros et al., 2020) provided for this challenge was collected from the News on Web (NoW) corpus (Davies, 2013). For task1, we are provided with 10469 text paragraphs. Each paragraph instance in the dataset is provided with paragraph-level label, vulnerable community-info which includes (disabled, homeless, hopeless, immigrant, in-need, migrant, poor families, refugee, vulnerable and women), and along country of origin. There are 5 classes (0-4) based on the intensity of PCL. For task2, we are provided with 993 paragraphs. Each paragraph instance in the dataset is provided with keyword, country of origin, span-text, category label, and number of agreeing annotators. The labels comprise of 7 classes: *Unbalanced Power relations*(unb) , *Shallow solution*(shall), *Presupposition*(Pres), *Authority Voice*(Auth), *Metaphor*(Meta), *Compas-*

Class	Nb of Samples
Unbalanced power (unb)	716
Shallow solutions (shall)	196
Presupposition (Pres)	224
Authority Voice (Auth)	230
Metaphor (Meta)	197
Compassion (Comp)	469
The poorer,the merrier (Poorer)	40

Table 2: Dataset Statistics Task 2

Model	Val-F1	Test-F1
SVM	**0.91**	0.053
Logistic	0.86	0.390
SGD	0.90	0.058
MLP	0.89	0.300
AdaBoost	0.89	0.280
Ensemble	-	0.340
Roberta-baseline	-	**0.491**

Table 3: F1- score for ML Models for task1

sion(Comp), *The poorer,the merrier*(poorer). For our experiments, we perform 80-20 data split with random state 0 for both the tasks to train the models for all experimental setup.

3 Experiment Setup

3.1 Pre-processing

Task1 is a binary text classification. The dataset is annotated from 0 to 4 on the basis of PCL intensity in the text. We further re-label the dataset instances using the intensity score where 0,1 referred to Non-PCL text and (2-4) referred to PCL text. (Ref . Table 1)

Task2 is multi-label classification. Each dataset instance is annotated with different PCL category labels and the text span reflecting the PCL label is provided respectively. Many paragraph instances were annotated for more than one category of PCL over a different span of text. (Ref . Table 2)

For our experiments, we remove stopwords by using NLTK(Natural Language Toolkit) library and other non-ascii symbols from the text before performing feature engineering.

3.2 Feature Engineering

Count Vectorizer Feature extraction (Vectorization) on text, encodes the text as integers or floating point values for using as input in machine Learning algorithms. Scikit-learn's CountVectorizer is used to convert a collection of text documents to a vector of term/token counts.

Term Frequency- Inverse Document Frequency (TF-IDF) We use Sklearn TF-IDF, which is an approach to quantify words in a set of documents by computing a score for each word to signify its importance in the document or corpus.

TF-IDF = Term Frequency (TF) * Inverse Document Frequency (IDF)

TF is the ratio the frequency of a word in a document and the frequency with the total number of words in the document whereas, document frequency (DF) is the normalized count of documents in which the term is present. Inverse Document Frequency is the inverse of the document frequency which measures the informativeness of a term in the document. We used the features generated on the entire corpus and the feature length was 20244.

3.3 Models

Support Vector Machine (Burges, 1998) is an effective technique for classifying high dimensional data. It learns the optimal hyperplane that separates training examples from different classes by maximizing the classification margin. Each row vector of the word-document matrix represents the vectorization of text that are mapped to a latent semantic space in this module by LSA vector space model, to generate representation vectors and further classify them. We perform Principal Component Analysis (PCA) (`number of components`=500) to perform dimension reduction over text features before inputting to this model.

Logistic Regression (Cramer, 2002) is a classification algorithm used to solve binary and multi-label classification. The logistic regression classifier uses the weighted combination of the input features and passes them through a sigmoid function.

Stochastic Gradient Descent (Ruder, 2016) is an iterative algorithm that starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function. We perform feature scaling before inputting the features to the model.

AdaBoost (Freund and Schapire, 1997) Adaptive Boosting is very popular boosting technique that combines multiple local weak classifiers into

Model	Unb	shall	Pres	Auth	Meta	Comp	poorer	Average F1
SVM	0.87	0.2	0.27	0.16	0.05	0.75	0	0.32
Logistic	0.84	0.56	0.53	0.37	0.27	0.76	0	**0.47**
SGD	0.84	0.25	0.35	0.08	0.05	0.68	0	0.32
MLP	0.84	0.42	0.47	0.27	0.26	0.71	0	0.42
AdaBoost	0.82	0.3	0.36	0.25	0.28	0.6	0	0.37

Table 4: F1-Score for ML Models for task2 on validation set

Model	Unb	shall	Pres	Auth	Meta	Comp	poorer	Average F1
SVM	0.11	0.18	0.02	0.094	0.10	0.10	0	0.089
Logistic	0.13	0.14	0.06	0.08	0.04	0.099	0	0.082
SGD	0.11	0.14	0.029	0.046	0.023	0.068	0	0.060
MLP	0.13	0.12	0.029	0.045	0.028	0.092	0	0.063
AdaBoost	0.12	0.089	0.027	0.039	0.054	0.101	0.045	0.068
Ensemble	0.1180	0.2050	0.0192	0.0643	0.0645	0.1018	0	0.082
Roberta-baseline	0.3535	0	0.1667	0	0	0.2087	0	**0.104**

Table 5: F1-Score for ML Models for task2 on test set

a single strong classifier. It can be used to significantly reduce the error of any learning algorithm that consistently generates classifiers whose performance is a little better than random guessing. We initiate the model with `number of estimators` = 400, `learning rate` = 1, and `base classifier` = DecisionTreeClassifier with `criterion` ='entropy'.

Multilayer Perceptron (MLP) (Rosenblatt, 1961) is a classical type of neural network. They are composed of one or more layers of neurons. Data is fed to the input layer, there may be one or more hidden layers providing levels of abstraction, and predictions are made on the output layer, also called the visible layer. MLPs are suitable for classification prediction problems because they are known to be capable of modelling complex functions. We used number of `hidden layers`= 30(for task1),and 1000 (for task2), and `maximum iteration` = 2000 to initiate the MLP model. Activation functioned used for the hidden layer is by default i.e ReLU (Nair and Hinton, 2010).

3.4 Ensemble

For our submitted system, after predictions were extracted from different models, we calculate the ensemble (Kuncheva and Whitaker, 2003) using the mode to find the most frequently occurring label. In the presence of a tie-breaker scenario, we select the label predicted by the best performing model.

3.5 Implementation

For all the models, we used Scikit learn library for our (Pedregosa et al., 2011) implementation. For task1, our validation set has 2094 examples and for task2, we have 199 examples from the training data according to the initial data split. All the models were initiated with `class_weight` ='balanced' setting , `maximum iteration`=1000. For the rest of the hyper-parameters we use the default setting otherwise specified in their sections. The github repository containing all the details of our experiments is made publicly available[1].

4 Results & Discussion

We evaluate and report the performance of different models on our validation set and the blind-test set (Table 3, 4, 5). We can see that logistic regression model performs the best among all the models with TF-IDF features as the input for task1 and SVM works better than of LR classfier for task2.

We see a steep drop in the performance of model on the test-set. This can be attributed to the difference distribution in the training and testing data which reduces the effectiveness of the TF-IDF features. The imbalance in the dataset also can be a reason for the model to perform badly on the 'Non-PCL' class in task1 (1) and 'poorer' class in task2 (1) . The machine learning models are not effective compared to the roberta-baseline as we see that

[1]https://github.com/Abhi-020/PCL

Gold	**Paragraph**	**Pred**
non-pcl	On the other hand,in Europe and North America,educated and young Muslims are surprisingly found to be **vulnerable** to such extremism	*pcl*
pcl	Many celebrities wore blue ribbons to support the American Civil Liberties Union, which is seeking to shed light on the plight of young **immigrants** facing the potential of being deported .	*non-pcl*

Table 6: Examples of incorrect predictions made by Logistic Regression Classifier in Task 1.

Gold	**Paragraph**	**Pred**
unb	Fast food employee who fed **disabled** man becomes internet sensation	*non-unb*
non-unb	When I was born , this was a nightmare town for **disabled** children , he said to me then	*unb*
shall	After a big casino win , Mario Balotelli gave a **homeless** man? 1,000 (PA)	*non-shall*
non-shall	I rather donate to the less privilege in the church or **homeless** than to pour a cup of water into Nigeria sea of wealth so that the thieves can grab my little contribution .	*shall*
Pres	Once again the stateless Rohingya are on the run – **homeless** and increasingly hopeless .	*non-Pres*
non-Press	Antidote for **hopelessness** Pulitzer Prize-winning journalist Roy Gutman , author of How We Missed the Story , argued that journalism in conflict zones provides change-makers and hope as an antidote for **hopelessness** .	*Press*
Auth	Every family which qualifies for the program should be covered . Every child in **poor families** must be placed and kept in school , and they should enjoy health and nutrition assistance , Romualdez said	*non-Auth*
non-Auth	The government is implementing several schemes would change the economic position of **poor families** , " she added	*Auth*
Meta	It is the supreme task of this generation to give **hope** to the **hopeless** strength to the weak and protection to the defenceless	*non-Meta*
non-Meta	They discounted and denied every conceivable poll which, showed Jonathan losing the election ,preaching that Nigerians wanted continuity ,not the change the opposition advocated . he people of Nigeria were portrayed as somehow loving their poverty and insecurity , their darkness and weakness , **hopelessness** and joblessness.	*Meta*
Comp	Today , **homeless** women are still searching for the same thing . A place to sleep and be safe .	*non-Comp*
non-Comp	Housing Minister Grant Shapps added :'The plight of **homeless** people should be on our minds all year round - not just at Christmas . families and be symbols of hope and possibility , of never giving up .	*Comp*
Poorer	A lot of my disabled patients over the years have gained strength and hope from me when they see that I also have a **disability** , but that I 'm coping . Sometimes the biggest gift I can give other people with **disabilities** is to show them that you can get a job .	*non-Poorer*
non-Poorer	One of her proudest achievements as an MP is challenging how the **disabled** are treated She became the first **disability** issues spokesperson and later minister .	*Poorer*

Table 7: Examples of incorrect predictions made by Logistic Regression Classifier in Task 2.

static text features are less helpful in the detection of PCL.

The table 6, 7 contains examples from the task1 and task2 validation set respectively, where the model failed to label the paragraph instances correctly. We can see that, the presence of vulnerable community keywords (Highlighted Table 6,7) often confuses the model leading it to mislabel the instances. We observe that TF-IDF features are not able to capture contextual information as they rely only on the presence of the word indicators . We believe that this is the reason behind the inefficiency of the ML models.

5 Conclusion

This paper presents our study of machine learning models for the binary and multi-label text classification on the PCL detection shared task. We find that tf-idf features can be effective in cases where train and testing data are from the same distribution but it may fail otherwise. For future work, we plan to experiment with contextual embeddings from BERT, and other transformer-based models. We also would like to look into bootstrapping and data augmentation techniques to solve the class imbalance problem more effectively.

References

Christopher JC Burges. 1998. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167.

Jan Salomon Cramer. 2002. The origins of logistic regression.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30.

Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.

Ludmila I Kuncheva and Christopher J Whitaker. 2003. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Icml*.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2018. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Carla Perez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Amir H Razavi, Diana Inkpen, Sasha Uritsky, and Stan Matwin. 2010. Offensive language detection using multi-level classification. In *Canadian Conference on Artificial Intelligence*, pages 16–27. Springer.

Frank Rosenblatt. 1961. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY.

Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Xinyi Zhou and Reza Zafarani. 2020. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Computing Surveys (CSUR)*, 53(5):1–40.

RNRE-NLP at SemEval-2022 Task 4: Machine Learning Approaches to Detect Patronizing and Condescending Language

Rylan Yang
The Harker School
San Jose, CA
23rylany@students.harker.org

Ethan A. Chi
Stanford University
Stanford, CA
ethanchi@stanford.edu

Nathan A. Chi
De Anza College
Cupertino, CA
chinathan@student.deanza.edu

Abstract

An understanding of patronizing and condescending language detection is an important part of identifying and addressing discrimination and prejudice in various forms of communication. In this paper, we investigate several methods for detecting patronizing and condescending language in short statements as part of SemEval-2022 Task 4. For Task 1a, we investigate applying both lightweight (tree-based and linear) machine learning classification models and fine-tuned pre-trained large language models. Our final system achieves an F1-score of 0.4321, recall-score of 0.5016, and a precision-score of 0.3795 (ranked 53 / 78) on Task 1a.

1 Introduction

Patronizing and Condescending Language (PCL) is characterized by expressions that reveal a sense of compassion or superiority toward others. Research suggests that PCL can perpetuate–and even veil–discrimination toward vulnerable groups (Ng, 2007). To make matters worse, its presence is often more subtle than other offensive language (Mendelsohn et al., 2020).

Detecting PCL is a challenging task for humans annotators–and the task becomes even trickier for artificial systems. Given the varied nature of condescension, current NLP models struggle to accurately detect PCL. Part of the issue is defining what patronizing and condescending language is, exactly–what one reader considers condescending might be deemed perfectly respectful by another.

SemEval-2022 Task 4 attempts to address some of these issues (Pérez-Almendros et al., 2022). Pérez et al. classify PCL into seven distinct categories: *unbalanced power relations*, *shallow solutions*, *presupposition*, *authority voice*, *metaphor*, *compassion*, and *the poorer, the merrier*.

Task 1a seeks to determine whether the sequence of text contains any form of patronizing or condescending language. Task 1b seeks to identify the PCL category that corresponds to the sequence of patronizing or condescending text. Overall, developing systems that perform well on these tasks—-that are capable of flagging condescending language–is a critical step toward reducing discrimination toward minority groups in media. We investigate various lightweight models to determine whether such models trainable on an extremely small compute budget could effectively identify PCL, as well as larger pre-trained transformer models to identify whether performance improves as models increase in size and complexity.

2 Dataset

For Task 1a, we train and validate our models on the SemEval-2022 Task 4 training set (Pérez-Almendros et al., 2020). Each paragraph has been annotated by two annotators on a Likert-type scale from 0 to 2 as shown in Table 1. The scores from each annotator are summed together: an overall score of 0 signifies that both annotators gave scores of 0, 1 that just one annotator gave a score of 1, and 2-4 that any higher score given by both annotators was summed together. A summary of the PCL status based on the two annotators' scores is shown in Table 2.

We did not investigate Task 1b.

Rating	Description
0	no presence of PCL
1	borderline PCL
2	contains PCL

Table 1: Likert scale for annotators to describe PCL status.

2.1 Train-test split

The dataset has a total of 14366 examples, split 10469–3897 between training and testing sets. The testing set was not provided until the last phase of the competition, so we created our own validation

Sum	Description
0-1	Not a PCL paragraph
2-4	A PCL paragraph

Table 2: Summary of PCL status based on two annotators' scores.

set using a 75/25 train/validation split. For this reason, our train set has 7851 examples, and our validation set has 2618 examples. In our paper, all "validation-set" performance is reported on this internal held-out set.

3 Methods

3.1 Systems Overview for Task 1a

The aim of this task is to classify a given sequence of text as patronizing and condescending or not. We implement the following lightweight machine learning classifiers in **Scikit-learn** (Pedregosa et al., 2011):

- **Logistic Regression** is a supervised classification algorithm that employs a logistic function to categorize data into discrete classes. (LaValley, 2008)
- **Support Vector Machine** is a supervised classification algorithm that maps data points in a hyperplane to maximize the width of the gaps between two or more categories. (Gold and Sollich, 2003)
- **Random Forest** is an supervised learning technique that utilizes random bagging of different bootstrap samples of data to create a prediction from uncorrelated trees that is more accurate than any one tree. (Liaw et al., 2002)
- **Multi-layer Perceptron** is a feed-forward neural network with an input layer, an output layer, and any number of hidden layers. (Gardner and Dorling, 1998)
- **Gradient Boosting** is a greedy additive algorithm that sequentially ensembles an number of weak learners (typically decision trees) (Natekin and Knoll, 2013).
- **AdaBoost (Adaptive Boosting)** is a form of gradient boosting that adds weights to each subsequent weak learner (also typically decision trees) with incorrectly classified samples until either all data points have correctly classified or the maximum iteration has been reached. (Hatwell et al., 2020)

Ensemble We also experiment with a Voting-Classifier ensemble which incorporates one Logistic Regression, one Random Forest, and one Gaussian-hybrid models. Our models were averaged with equal weights.

We also experiment with the following pre-trained language models to try and effectively classify the presence of PCL in sentences:

- **BERT** is a pre-trained masked language model. We use BERT-cased, BERT-Large-cased, BERT-uncased, and BERT-Large-uncased in our experiments. (Devlin et al., 2018)
- **RoBERTa** is an optimized BERT model that utilizes the same architecture but various changes such as larger mini-batches and learning rates. We use RoBERTa and RoBERTa-Large. (Liu et al., 2019)

3.2 Experimental Setup

Normalization We investigate standardizing the dataset (implemented with the **Scikit-learn** *StandardScaler* preprocessing function) for the lightweight models.

Pre-Trained Models Regarding the large pre-trained models, we trained with binary cross-entropy loss for 5 epochs, using a learning rate of 1×10^{-5} and batch sizes of 8 (BERT and RoBERTA-base) and 3 (BERT and RoBERTa-Large).

4 Results

The official evaluation set performances for our classifiers are listed in Table 3, while the unofficial validation set performances for our Scikit-learn and Transformer-based models are listed in Table 4.

For Task 1a (patronizing and condescending language binary classification), we submitted our two highest-performing lightweight models (Support Vector Machine and Random Forest models). Due to error, we did not submit our BERT model. Overall, we ranked 53rd out of 78 on this task, achieving a positive-class F1 score of 0.4321.

Model	positive-class F1 (1a)	recall-score (1a)	precision-score (1a)
Support Vector Machine	**0.4321**	**0.5016**	**0.3795**
Random Forest	0.3310	0.3691	0.3000

Table 3: Official validation set performance of our lightweight models on Task 1a (binary classification).

Model	Features	positive-class F1 (1a)	Accuracy (1a)	Normalize
Logistic Regression	GloVe	0.37	0.76	False
Logistic Regression	GloVe	0.37	0.76	True
Support Vector Machine	GloVe	0.37	0.73	False
Support Vector Machine	**GloVe**	**0.48**	**0.89**	**True**
Random Forest	GloVe	0.38	0.87	False
Random Forest	GloVe	0.39	0.86	True
Multi-layer Perceptron	GloVe	0.40	0.90	False
Multi-layer Perceptron	GloVe	0.34	0.88	True
AdaBoost	GloVe	0.31	0.90	False
AdaBoost	GloVe	0.31	0.90	True
VotingClassifier Ensemble	GloVe	0.42	0.87	False
VotingClassifier Ensemble	GloVe	0.42	0.87	True
RoBERTa-base	—	0.54	0.92	False
RoBERTa-large	—	**0.55**	**0.92**	**False**
BERT-cased	—	0.55	0.91	False
BERT-uncased	—	0.51	0.91	False
BERT-large-cased	—	0.56	0.93	False
BERT-large-uncased	—	0.53	0.92	False

Table 4: Unofficial validation set performances of candidate models on Task 1a (binary classification). For this task, the highest-performing lighweight models are the Support Vector Machine model and the Random Forest model, and the highest-preforming pre-trained models are BERT-cased and RoBERTa-large.

5 Conclusion

We have proposed lightweight and pre-trained systems that are able to detect PCL in text.

We find that reasonably lightweight models such as Support Vector Machine and Random Forest are effective at predicting the level of patronizing and condescending language. However, we note that ensembling these models together does not improve performance.

Additionally, normalizing the dataset had little effect for most models—-and in the case of the Multi-layer Perceptron model actually returned a lower positive-class F1 score. However, it substantially increased the F1 score with the Support Vector Machine model from 0.37 to 0.48.

Finally, we find that fine-tuning large pre-trained models like BERT and RoBERTa achieves results at least as accurate as lightweight models–if not better.

An area of interest for future work may be further experimentation with ensembles of lightweight models, as well as inquiries into adversarial and contrastive learning to improve overall accuracy.

Overall, our results show that both lightweight and fine-tuned models can achieve reasonable results at detecting patronizing and condescending language in human channels of communication.

6 Acknowledgements

The authors would like to thank Ryan A. Chi for their support and guidance throughout this project. We would also like to acknowledge Google Collaboratory for their free services.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Matt W Gardner and SR Dorling. 1998. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636.

Carl Gold and Peter Sollich. 2003. Model selection for support vector machine classification. *Neurocomputing*, 55(1-2):221–249.

Julian Hatwell, Mohamed Medhat Gaber, and R Muhammad Atif Azad. 2020. Ada-whips: explaining adaboost classification with applications in the health sciences. *BMC Medical Informatics and Decision Making*, 20(1):1–25.

Michael P LaValley. 2008. Logistic regression. *Circulation*, 117(18):2395–2399.

Andy Liaw, Matthew Wiener, et al. 2002. Classification and regression by randomforest. *R news*, 2(3):18–22.

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

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A framework for the computational linguistic analysis of dehumanization. *Frontiers in artificial intelligence*, 3:55.

Alexey Natekin and Alois Knoll. 2013. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21.

Sik Hung Ng. 2007. Language-based discrimination: Blatant and subtle forms. *Journal of Language and Social Psychology*, 26(2):106–122.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Model	Hyperparameter	Task 1a
Logistic Regression	solver	lbfgs
	penalty	none
	class weight	balanced
Support Vector Machine	class weight	0:1, 1:13
	degree	poly
Random Forest	max depth	10
	n estimators	100
	class weight	balanced
	min samples leaf	10
Multi-layer Perceptron	hidden layer sizes	(100, 100)
	α	0.01
	β	0.2
	learning rate	adaptive
AdaBoost	learning rate	1.0
	n estimators	50

Table 5: Hyperparameters for lightweight supervised models.

Sentence	label
" Anja Ringgren Loven I ca n't find a word to describe how I feel for you .... May God almighty keep blessing you and always give you strength and sound health to continue your good work ..... You gave hope to the hopeless ! ! ! ! Have so much respect for you .. Stay Blessed my good fellow ... " says one commenter on Facebook. "God bless you and your mission . Glad to see Hope (and all the children) growing up loved , well fed , happy , having fun , and going to school , " says another .	4
We 're living in times of absolute insanity, as I 'm pretty sure most people are aware . For a while , waking up every day to check the news seemed to carry with it the same feeling of panic and dread that action heroes probably face when they 're trying to decide whether to cut the blue or green wire on a ticking bomb – except the bomb 's instructions long ago burned in a fire and imminent catastrophe seems the likeliest outcome . It 's hard to stay that on-edge for that long , though , so it 's natural for people to become inured to this constant chaos , to slump into a malaise of hopelessness and pessimism .	0

Table 6: Examples that are considered patronizing and condescending and those not considered patronizing and condescending, respectively

UTSA NLP at SemEval-2022 Task 4: An Exploration of Simple Ensembles of Transformers, Convolutional, and Recurrent Neural Networks

Xingmeng Zhao and **Anthony Rios**
Department of Information Systems and Cyber Security
University of Texas at San Antonio
San Antonio, TX
{xingmeng.zhao, anthony.rios}@utsa.edu

Abstract

The act of appearing kind or helpful via the use of but having a feeling of superiority condescending and patronizing language can have have serious mental health implications to those that experience it. Thus, detecting this condescending and patronizing language online can be useful for online moderation systems. Thus, in this manuscript, we describe the system developed by Team UTSA SemEval-2022 Task 4, Detecting Patronizing and Condescending Language. Our approach explores the use of several deep learning architectures including RoBERTa, convolutions neural networks, and Bidirectional Long Short-Term Memory Networks. Furthermore, we explore simple and effective methods to create ensembles of neural network models. Overall, we experimented with several ensemble models and found that the a simple combination of five RoBERTa models achieved an F-score of .6441 on the development dataset and .5745 on the final test dataset. Finally, we also performed a comprehensive error analysis to better understand the limitations of the model and provide ideas for further research.

1 Introduction

Patronizing and condescending language (PCL) generally appears as an act to hold a superior attitude, resulting in language that "talks down" to others. For instance, PCL may describe someone in a power position as the potential "savior" of a vulnerable community (e.g., "*A donation of one dollar can save a life*"), masquerading a sense of superiority as compassion. There has been recent research suggesting that PCL can have adverse effects on the mental health of individuals (Giles et al., 1993; Shaw and Gordon, 2021), particularly in the context of ageism. While there has been substantial research on PCL in various contexts (Huckin, 2002; Komrad, 1983; Giles et al., 1993; Shaw and Gordon, 2021), unfortunately, there have been few efforts to develop PCL detectors in the field of Natural Language Processing (NLP). Hence, this paper describes our team's (UTSA NLP) contributions on the SemEval-2022 Task 4 (Pérez-Almendros et al., 2022) that introduced a new dataset for detecting PCL language.

NLP has investigated a broad spectrum of problematic language usages, such as hate speech (Vidgen et al., 2021), sarcasm language (Bamman and Smith, 2015), fake news (Hu et al., 2021), and the spread of rumors and disinformation. However, PCL has only recently been explored in the NLP community (**?**). To alleviate this issue, SemEval-2022 Task 4 expanded on the work by **?**, releasing a large PCL dataset for two PCL subtasks. Subtask 1 focuses on detecting the presence of PCL in news stories. PCL detection consists of a variety of sub-problems, for instance, identifying the exact PCL type expressed post (if any). There are multiple technical challenges for identifying PCL. For instance, accurate models must handle imbalanced data (most news stories do not contain PCL) and complex semantic understanding to relate shallow solutions for helping vulnerable populations. For instance, **?** describe "Shallow Solutions" as a type of PCL, e.g., "*Raise money to combat homelessness by curling up in sleeping bags for one night*". Nevertheless, for a model to understand this is an example of PCL, it would need to understand that "curling up in sleeping bags for one night" is unlikely to help the general problem of homelessness. Hence, we hypothesize that different models will learn to detect different types of PCL with varying accuracy; thus, combining multiple methods can result in better performance than a single method.

Overall, this paper describes our system for Task 1. Specifically, we evaluate multiple combined methods to handle PCL's complex nature better than a single method. Hence, for our methodology, we trained a RoBERTa model and two traditional deep learning models (a Convolutional

Neural Network and a Long Short-Term Memory Network) for comparison. In addition, we experimented with different model hyperparameters, random seeds, thresholds, and pre-trained word embeddings using the performance on the validation set to assess model variants. Finally, we evaluate multiple simple, yet effective, methods of combining the neural network models in an ensemble.

2 Background

Based on the work of **?**, the SemEval Task 4 dataset contains 10,637 news stories about vulnerable people published in 20 English-speaking countries, with a novel PCL taxonomy consisting of three top-level categories (The savior, The expert, and The poet) and seven low-level PCL categories describing the different types of condescension (Perez Almendros et al., 2020). Thet task contains two subtasks: binary classification (subtask 1) and multi-label classification (subtask 2). The binary classification for subtask 1 annotated the data with one of two categories: PCL and Not PCL. Subtask 2, the multi-label classification task, categorizes the news stories into a subset of of seven different PCL categories: unbalanced power relations, authoritative voice, shallow solution, presumption, compassion, metaphor, and the pooer, the merrier. A complete description of each category can be found in **?**.

PCL has been studied in a wide array of contexts, from sociolinguistics to healthcare (Huckin, 2002; Komrad, 1983; Giles et al., 1993; Shaw and Gordon, 2021). However, much of the prior work has focused on interviews and general qualitative methods. Thus, automated PCL detection models can provide social scientists with tools to understand the impact of PCL at scale. For instance, PCL models would allow linguistics to understand the implicit language actions related to condescension and aid social scientists in researching the link between condescension and other characteristics like gender or socioeconomic status because these superior attitudes and discourse of pity can routinize discrimination and make it less visible(Ng, 2007). However, much of the research on harmful language in NLP has concentrated on the explicit, offensive, and apparent phenomena like false news identification, trustworthiness prediction and fact-checking, modeling offensive language, both generic and community-specific (Vidgen et al., 2021; Zampieri et al., 2019; Schmidt and Wiegand, 2019); or how rumors spread (Ma et al., 2017). Recently, some work on condescending language has begun to surface. For example, based on the challenge that condescension is often undetectable from isolated discourse because it depends on discourse and social context, Wang and Potts (2019) introduces the task of modeling the phenomenon of condescension in direct communication from an NLP perspective and developing a dataset with annotated social media messages. Likewise, **?** also trained various baseline models to examine how existing NLP approaches perform in this task. Although they observe that recognizing PCL is achievable, it is still difficult. Hence, the work by **?** formed the basis of SemEval Task 4.

3 System Description

Overall, we developed an ensemble model strategy for the PCL challenge. Specifically, we evaluated three individual methods: RoBERTa, Convolutional Neural Networks, and Long Short-Term Memory Networks. Furthermore, we experimented with various ensemble combinations. Approaching the task we conduct multiple experiments with a variety neural network architectures using Convolutional Neural Networks (CNN) (Kim, 2014), Bi-directional long short term memory (BiLSTM) (Huang et al., 2015), and the pre-trained transformer-based model, RoBERTa (Liu et al., 2020). Each model and ensemble method is described in the section below.

CNN. We use the CNN model introduced by Kim (2014). Intuitively, the CNN model learns to extract predictive n-grams from the text. For the CNN architecture, we used filter sizes that span two, three, and four words. For the activation functions, we used ReLU (Glorot et al., 2011). Furthermore, we only needed two filters for each filter size [1]. Between the max-pooled outputs from the convolutional layer and the the full-connected output layer, we use dropout with the probability set to 0.5 during training. The final fully-connected output layer uses a Softmax activation and outputs class probabilities for PCL or Not PCL. The model was trained with the Adam optimizer (Kingma and Ba, 2015). Furthermore, we trained the CNN models with various learning rates randomly selected from 1e-4 to 1e-3 for a maximum of 35 epochs.

[1]We experimented with filter normal filter sizes from 100 to 300, but two seemed to perform just as well. We hypothesize this is because of the small number of PCL examples in the dataset.

BiLSTM. While CNNs only extract informative n-grams from text, recurrent neural networks (RNNs) are able to capture long term dependencies between words. For our RNN method, we use a Bidirectional Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997), specifically we use a variant introduced by Graves (2012). For the hyper-parameters, we did not use dropout, trained for a maximum of 35 epochs, and used variety hidden layer sizes (128, 256, and 512). The models were trained with the Adam optimizer (Kingma and Ba, 2015). Furthermore, we trained the BiLSTM models with various learning rates randomly selected from 1e-4 to 1e-3.

RoBERTa. In our study we used a variant of BERT (Kenton and Toutanova, 2019), namely RoBERTa (Liu et al., 2020) model, which is lighter and faster. Specifically, we use the roberta-base variant in the HuggingFace package (Wolf et al., 2019). We trained the RoBERTa model for 20 epochs with a mini-batch size set to 8 with the Adam optimizer. The learning rate was initially set to 2e-5 (other hyper-parameters same as (Liu et al., 2020)) and the adjusted stepwise linear decay was used to modify the learning rate through training, with step sizes of two and three used. Moreover, we used the last layer's CLS token which is passed to a final softmax layer. The model was check-pointed after each epoch, and the best version was chosen using the validation data.

Pre-trained Word Embeddings. For the CNN and BiLSTM models, we compare the following pre-trained word embeddings: Word2Vec vectors trained on Google News corpus (Mikolov et al., 2013), GloVe vectors trained on Wikipedia2014 and Gigaword5 corpus (GLoVe-Word) and Twitter (GLoVe-Twitter) corpora (Pennington et al., 2014), and FastText vectors trained on CommonCrawl corpora (Bojanowski et al., 2017).

Ensemble Model. There has been a wide array of research showing that ensembles of deep learning models have are useful for boosting model performance (Allen-Zhu and Li, 2020; Peng et al., 2018). We built different ensemble models by taking an unweighted average of the probability outputs of each of the independently trained models. This includes models trained with different hyperparameters, e.g., hidden state size for the BiLSTM models, different learning rates, and different random seeds. For the CNN and BiLSTM, each model was trained on four different pre-trained word vectors described above. The CNN and LSTM models were also trained on four different random seeds, for each combination of word embedding and learning rate. The RoBERTa model was trained with eleven random seeds and a number of two different step-wise learning rate schedulers using step sizes of two and three. Overall, we trained a total of 46 different models for the PCL task. Next, we experimented with two methods of model averaging (i.e., an ensemble): Ensemble 1 and Ensemble 2.

Model	Embedding	Seed	LR	HL
CNN	FastText	99	0.002	NA
LSTM	Glove_Twitter	99	0.002	128

Table 1: The hyperparameters for the best CNN and LSTM models found using random search. We report random seed (Seed), learning rate (LR), and pretrained embeddings (Embedding), and hidden layer size (HL) in this table.

First, for **Ensemble 1**, we simply average the top five model instances—which resulted in different RoBERTa models trained with different random seeds and learning step settings, i.e. step sizes of two with random seed of three; step size of three with random seed of zero, two, four, or seven. Second, for **Ensemble 2**, we experimented with taking the top five models combined with the top two CNN and BiLSTM models, and the hyperparameters (e.g., learning rate and embedding size) of the best performing models are shown in Table 1.

4 Dataset, Experimental Setup, and Training Details

For subtask 1, we use the train dataset provided by the PCL organizers. We choose the best epoch and the best hyperparameters using performance assessed in terms of F1-score on this development set based on the random search (Bergstra and Bengio, 2012). We saved the best epoch and best hyperparameters for each model variant. For evaluation, we use the provided test and validation datasets released by the organizers. We implemented our models on four GPUs using PyTorch (Paszke et al., 2019) to train binary classifiers for PCL. We use Cross-Entropy Loss in all our experiments as the loss function. We ran the experiments on a server using a GPU CUDA Version: 11.4. We selected the epoch based on the F1 score on the development set to save the best version of each model.

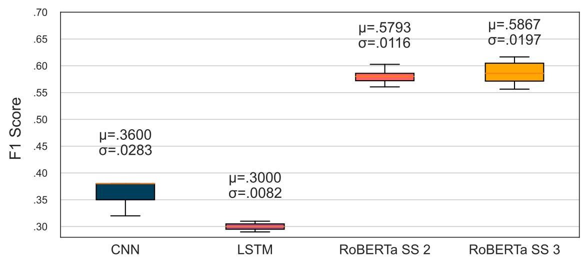


Figure 1: F1 score distributions for the best CNN, LSTM and RoBERTa models with the same hyperparameters, but different random seed values on the development dataset. SS stands for the learning rate step size.

To build a basis for comparison, all models were trained using the training data provided by the task organizers and evaluated against the provided validation dataset. The best-performing models were then submitted for evaluation against the test dataset during the task evaluation period. The training method was repeated three times for CNN and LSTM and eleven times for Roberta, each with a new random seed. This is because changing the random seed used in fine-tuning RoBERTa models can provide significantly different outcomes, even if the models are similar in terms of hyper-parameters (Dodge et al., 2020). The best-performing hyperparameters in each model were saved for the remainder of the ensemble study. We examined three random seeds (17, 42, and 99) for the best CNN and LSTM models, with standard deviations .0283 and .0082, respectively. We also evaluated ten random seeds for the RoBERTa model with step sizes of 2 and 3, yielding standard deviations for variance of .0116 and .0197, respectively. The distribution of each model's performance for different random seeds is shown in Figure 1.

We attempted to build a robust ensemble classifier with softmax output aggregation. For the ensemble model, the default threshold for interpreting probabilities as class labels is 0.5, but due to the imbalanced classification problem, we adjust the optimal threshold range from 0.1 to 0.9 when converting probabilities to class labels. We found the optimal probability threshold of CNN, LSTM, and RoBERTa that resulted in the best F1 score on the validation dataset were .1, .35, and .35, respectively.An optimal threshold was also chosen for the ensemble model, which was found to be .35.

		AVG P.	AVG R.	AVG F1
RoBERTa	step_size = 2	.5948	.5738	.5826
	step_size = 3	.6006	.5916	.5952
CNN	GoogleNews	.2542	.7085	.3733
	FastText	.2738	.5729	.3653
	Glove_Word	.2253	.4640	.3031
	Glove_Twitter	.2070	.6549	.3132
BiLSTM	GoogleNews	.3250	.4087	.3609
	FastText	.3659	.4020	.3810
	Glove_Word	.3525	.4271	.3831
	Glove_Twitter	.3745	.3953	.3821

Table 2: Average precision (AVG P.), average recall (AVG R.), and average F1 (AVG F1) for each model.

	step_size	seed	Prec.	Rec.	F1
		Development Results			
RoBERTa	3	0	.6103	.6533	.6311
		4	.6263	.5980	.6118
		2	.6029	.6181	.6104
		7	.6277	.5930	.6098
	2	0	.5980	.6131	.6055
Ensemble 1	—	—	**.6215**	**.6683**	**.6441**
Ensemble 2	—	—	.6093	.6583	.6328
		Test Results			
Ensemble 1	—	—	.5412	.5804	.5601
Ensemble 2	—	—	**.5599**	**.5899**	**.5745**

Table 3: Individual models in the best ensemble, and overall ensemble performance on the development and test datasets.

5 Results

Table 2 shows the average recall, precision, and F1 score. The scores are averaged across the different random seeds and hyperparameters used to train the models. Overall, we notice that the RoBERTa model outperforms both the CNN and BiLSTM models by more than 20%. For the CNN model, we find that the GoogleNews word embeddings perform best. However, for the BiLSTM model, we find that the model performs similarly across all pretrained embeddings, with the Glove Word embeddings slightly outperforming others.

In Table 3 we report the results of the two ensemble models: Ensemble 1 (only RoBERTa Models) and Ensemble 2 (Combining RoBERTa with the CNN and RNN models). On the development set, we find that that a single RoBERTa model achieves an F1 of .6311, with the next best four models achieving an F1 of around .61. The F1 of Ensemble 1 improves on the best RoBERTa models result

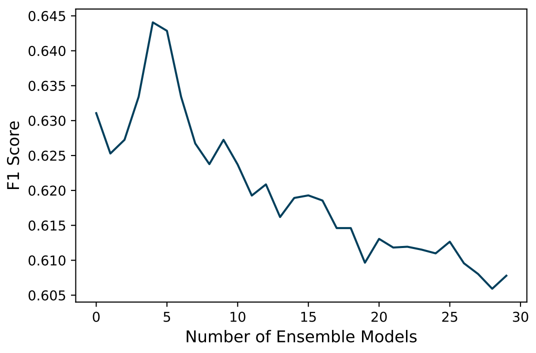


Figure 2: F1 score for different sized ensembles on the development dataset.

	FPs	FNs	Total PCL	Total
homeless	15	9	29	212
poor-families	13	17	38	190
women	3	8	14	233
in-need	15	2	33	226
immigrant	0	4	7	218
hopeless	15	9	26	217
vulnerable	6	4	20	209
migrant	2	3	5	207
disabled	4	7	14	194
refugee	8	3	13	188

Table 4: Summary of the false positives and false negatives found in each of the then PCL types.

by more than 1%. Ensemble 2 only improves on the Best F1 by .1%. However on the final test set, we find that the differences is not meaningful, with Ensemble 2 slightly outperforming Ensemble 1 (.56 vs. .57).

Next, in Figure 2 we report the results of averaging different number of models in our ensemble. More specifically, we evaluate averaging the best two models, best three models, and best N models, for N up to an ensemble of 30 different models. The model is chosen based on the top N performing models across all model types and random seeds. Overall, we see that initially the result of an ensemble of size 1 (i.e., only using the best RoBERTa model) has an F1 of around .63. However, that slowly increases beyond .64 at around the top five models. After that, the results slow decrease. Overall, we find that while a few models with varying performance improves the results. The more inaccurate models slowly outweigh the best performing model, thus decreasing the overall results. However, we find that the results stabilize around 0.61. Finally, in Table 4 we measure the number of False Positives and False Negatives for each of the main keywords identified in the PCL dataset (e.g., they keywords are provided by the organizers indicating a vulnerable group). The model produced 66 false negative predictions and 81 false positives predictions in total, but most of false positive errors are come from homeless,in-need, poor families, and hopeless. And the false negative error occur more frequently among the homeless, woman, immigrant, migrant and disabled topic.

5.1 False Positives and False Negatives

In addition to an exploration of the observation results, we perform an error analysis by manually comparing the true labels and predictions of Ensemble 1. First, for False Positives, we analyze an example related to "hopelessness/homelessness":

FP Example:"The City Without Drugs organisation is still active , as is their YouTube channel . It features hundreds of videos of drug addicts being dragged half-conscious through the street , their faces not blurred , or confessing their alleged worthlessness , their hopelessness , their shame."

This paragraphs is predicted as PCL, but the ground-truth is Not PCL. This example indicates that the Ensemble incorrectly identifies sentences as PCL when they contain many PCL-related words that may be related to PCL-like text (e.g., related to hopelessness), even when the text is not directly indicating a feeling of superiority. Another false positive example is from the "homeless" topics:

FP Example:"Viral photo helping fund homeless kid , his dog."

We can see that the entity of this sentence is a single individual. This paragraph is recognized as PCL by the system, maybe because the PCL system believes it contains the shallow solution (i.e., viral photo). However, it neglected the fact that financing a specific homeless child may be realistic, i.e. it may not be a shallow solution for a single person. Perez Almendros et al. (2020) also mention that shallow solutions are also often overlooked by RoBERTa, where recognizing shallow solutions in the text requires external knowledge of the situation and the needs of those affected. Thus, a large number of false positive results are generated by misidentifying the entities and the relationship between patronizing and condescending language. Next, we look at a False Negative:

FN Example: "Charity plans to forgo parking so homeless can have gym and medical centre."

Here we find another issue with shallow solutions. Specifically, the model is not able to associate the proposed procedure as not being a method of really addressing homelessness. Specifically, the PCL system unaware that "forgoing parking" is not a complete solution to help homeless people, which is a simple and superficial philanthropic effort that is unlikely to make a significant difference on vulnerable communities. The second example of a false negative concerns presuppositions. People need to decide whether the assumption made is reasonable or not for this type of PCL. We found for the political topic, like immigrant and migrant topics, there are many preconceived assumptions in this kind of news. For example, in the following situation, the author assumes Filipino families are poor and need assistance based on stereotypes.

FN Example:"But if the Supreme Court gives a favorable decision for the president , his immigration program would immediately take effect , changing the lives of eligible Filipino families and other immigrants."

This error suggests that the model is incapable of understanding complex relationships between vulnerable communities and ideas. A future interesting research avenue would explore methods for incorporating relevant knowledge bases, similar to recent work on common sense generation (Xing et al., 2021), into transformer models to address these errors.

6 Conclusion

In this paper, we have presented our submission for the PCL detection system submitted to the SemEval-2022 Task 4. Our team are focus on the subtask1 to identify whether the paragraphs contain the PCL or not. We proposed several ensemble models that leverages pre-trained word vectors and three different deep learning architectures. In future efforts, we plan to further improve our model by incorporating structured knowledge bases.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1947697.

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *CoRR*, abs/2012.09816.

David Bamman and Noah Smith. 2015. Contextualized sarcasm detection on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 9, pages 574–577.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).

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

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *CoRR*, abs/2002.06305.

Howard Giles, Susan Fox, and Elisa Smith. 1993. Patronizing the elderly: Intergenerational evaluations. *Research on Language and Social Interaction*, 26(2):129–149.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings.

Alex Graves. 2012. *Supervised sequence labelling with recurrent neural networks*, volume 385. Springer.

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

Linmei Hu, Tianchi Yang, Luhao Zhang, Wanjun Zhong, Duyu Tang, Chuan Shi, Nan Duan, and Ming Zhou. 2021. Compare to the knowledge: Graph neural fake news detection with external knowledge. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 754–763.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Mark S Komrad. 1983. A defence of medical paternalism: maximising patients' autonomy. *Journal of medical ethics*, 9(1):38–44.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 708–717.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Sik Hung Ng. 2007. Language-based discrimination: Blatant and subtle forms. *Journal of Language and Social Psychology*, 26(2):106–122.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Yifan Peng, Anthony Rios, Ramakanth Kavuluru, and Zhiyong Lu. 2018. Extracting chemical–protein relations with ensembles of svm and deep learning models. *Database*, 2018.

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

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Anna Schmidt and Michael Wiegand. 2019. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, April 3, 2017, Valencia, Spain*, pages 1–10. Association for Computational Linguistics.

Clarissa A Shaw and Jean K Gordon. 2021. Understanding elderspeak: An evolutionary concept analysis. *Innovation in aging*, 5(3):igab023.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682.

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yiran Xing, Zai Shi, Zhao Meng, Gerhard Lakemeyer, Yunpu Ma, and Roger Wattenhofer. 2021. Km-bart: Knowledge enhanced multimodal bart for visual commonsense generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 525–535.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1415–1420.

AliEdalat at SemEval-2022 Task 4: Patronizing and Condescending Language Detection using Fine-tuned Language Models, BERT+BiGRU, and Ensemble Models

Ali Edalat, Yadollah Yaghoobzadeh, Behnam Bahrak
School of Electrical and Computer Engineering, College of Engineering
University of Tehran, Tehran, Iran
{ali.edalat,y.yaghoobzadeh,bahrak}@ut.ac.ir

Abstract

This paper presents the AliEdalat team's methodology and results in SemEval-2022 Task 4: Patronizing and Condescending Language (PCL) Detection. This task aims to detect the presence of PCL and PCL categories in text in order to prevent further discrimination against vulnerable communities. We use an ensemble of three basic models to detect the presence of PCL: fine-tuned bigbird, fine-tuned mpnet, and BERT+BiGRU. The ensemble model performs worse than the baseline due to overfitting and achieves an F1-score of 0.3031. We offer another solution to resolve the submitted model's problem. We consider the different categories of PCL separately. To detect each category of PCL, we act like a PCL detector. Instead of BERT+BiGRU, we use fine-tuned roberta in the models. In PCL category detection, our model outperforms the baseline model and achieves an F1-score of 0.2531. We also present new models for detecting two categories of PCL that outperform the submitted models.

1 Introduction

Increasing internet access rates and the development of a diverse range of online forums have allowed people around the world to engage in a tremendous range of topics. This has been accompanied by an increase in unhealthy online texts whose negative effects on people have been significant. One type of such unhealthy texts is a text with patronizing and condescending language (PCL). When a person's language expresses a superior attitude towards others or describes their situation in a benevolent way that creates a sense of pity, the person has used this type of language. In the media, vulnerable communities seem to be a good target for this type of language. However, this type of language can normalize discrimination. We believe that unfair treatment of vulnerable groups leads to greater deprivation and inequality for these groups. Therefore, recognizing the existence of this type of language and its variations is important and can prevent these problems.

So far, significant work has been done on modeling the language that deliberately and openly undermines others, such as offensive language or hate speech, but little has been done on the language of humiliation and pity. This language of humiliation and pity is used in the media subtly and indirectly and different from other types of unhealthy languages. The special focus on the language of humiliation and compassion for vulnerable communities has been noted only in the work of Pérez-Almendros et al., (2020). In this work, a dataset to identify this type of language is presented, but no significant work has been done in designing a model to classify this type of text.

Unhealthy text papers usually focus on obvious and aggressive phenomena such as detecting fake news, fact-checking, modeling offensive language, and spreading rumors. There has been a few work on PCL recently. Wang and Potts (2019) introduced compassion modeling in direct communication from the perspective of natural language analysis. They created and tagged a dataset with social media messages. Sap et al. (2019) Discussed the specific uses of language and power, especially the unbalanced power relations often present in degrading treatment, and the social consequences of these applications. Unfair treatment of disadvantaged groups was also examined as an example of these cases. Price et al. (2020) Provided datasets for classifying unhealthy speech on social media. They provided fine-grained classifications for all kinds of unhealthy writings, one of which was PCL.

Of course, the use of this type of writing is not limited to weak groups in society. There is still a need to design a model to detect such language towards vulnerable communities. PCL is a toxic language that implicitly has a negative impact on public opinion. There are tasks that generally identify toxic language that can be used to provide an answer to this problem. For example Lees et al. (2021) proposed the use of a fine-tuned BERT model to detect veiled toxicity.

To design a model to detect this language in vulnerable communities, we participated in SemEval 2022 task 4 competition (Pérez-Almendros et al., 2022). This paper describes the models we provide for detecting PCL. The contest data is taken from the work of Pérez-Almendros et al., (2020).

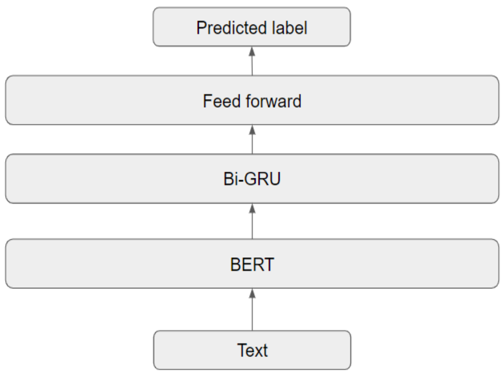


Figure 1 The general structure of BERT+BiGRU model

To detect the presence of PCL, we present an ensemble model. This model consists of three basic models: fine-tuned bigbird, fine-tuned mpnet and BERT + BiGRU. The models are combined with the weighted average. This model cannot perform better than the baseline because our model is overfitted. In this paper, we present a solution to this model's problem and create a new model that can achieve an F1-score of 0.5505. To identify existing PCL categories, we consider the categories separately. To detect the presence of any category, we act like detecting the presence of PCL. The basic models for making ensemble models are fine-tuned bigbird, fine-tuned mpnet and fine-tuned roberta. The model can outperform the baseline model. In addition to this model, in two categories, we improve the diagnostic model and build a new model. This model can achieve an F1-score of 0.3160. The statement of contributions is given below.

To balance the data set, we used a different method than the data set providers. Instead of using the sampling method (We sample twice the number of PCL data from non-PCL data), we used a combination of the sampling method and EDA (Wei et al., 2019). And in compassion and metaphor diagnosis, we used a set of related articles for balancing. We paid attention to the medium and high lengths of the texts and used language models with the ability to summarize long texts. We used the ensemble model for classification to help reduce the bias caused by the data set imbalance.

2 Models

In this section, we describe how to detect the presence of PCL in the text and how to detect the type of PCL in the text. We describe the models used for these diagnoses.

2.1 Subtask1: PCL detection

Recognizing the presence of PCL in the text is a two-class classification problem. To do this, we use a model that is an ensemble of three basic models. Our base models are fine-tuned bigbird, fine-tuned mpnet, and BERT+BiGRU model. We use BERT language model (Devlin et al., 2018) for this classification to prepare the BERT+BiGRU model. We also fine-tune the Big Bird language model (Zaheer et al., 2020) and the MPNet language model (Song et al., 2020) to prepare the fine-tuned bigbird and fine-tuned mpnet models.

The texts are taken from the news. For this reason, there is medium to long texts in the data, and to address this issue, we used two language models, mpnet and bigbird, to create our model. This allows the model for long texts to extract the information needed for classification. Details about the length of the texts are given in the section 4.1 .

Each of the basic models learn separately on the training data. We then combine the results of the models to predict the text class using the weighted average to generate our model prediction. To combine the models, we use the probability that the text has PCL. Each model predicts this probability for each text. First, we use the weighted average of the probabilities predicted by these two models. We use a weight of 0.4 for the mpnet model prediction and a weight of 0.6 for the bigbird model prediction. Then, we use the weighted average to combine the BERT+BiGRU model prediction and the average prediction of the previous two models. The weight of the BERT+BiGRU model in this average is equal to 0.3 and the weight of the combination of the two previous models is equal to 0.7.

2.2 BERT+BiGRU model

The model consists of three layers. The first layer of the model applies BERT. In this layer, we give the cleaned text to the language model and get the embedding of text tokens. Then we give tokens' embedding to the Bi-GRU layer. The output of the Bi-GRU layer is then given to the feed forward layer to predict the input class. The general structure of this model is shown in Figure 1. To clear the text, we remove the HTML tags, URLs, Mentions and Emojis in the text. More details of this model are given in Section 3.1 .

2.3 Fine-tuned other Language Models

To fine-tune these language models, we use a two-layer model. In the first layer, the language model takes the input text and creates a display for the entire text. The classifier token embedding is used to display the entire text. In the second layer, we predict the label using a feed forward network. In these models we do not clean the input text. Figure 2 shows the general structure of the model to fine-tune the language model.

2.4 Subtask2: PCL categories detection

The PCL categories detection problem is a multi-label classification. Given a paragraph, a system must identify which PCL categories (if any) appear in the paragraph. The problem is, a text can have multiple categories at the same time.

To solve this problem, we detect the presence of each category in the text separately from the other categories. That is, we create a separate model to identify each category. Each model solves a binary classification problem. This model determines whether the text has the desired PCL category or not.

To identify the "Unbalanced Power Relations" category in the text, we use an ensemble of two basic models. We use fine-tuned bigbird and fine-tuned mpnet as basic models. We use a weighted average to combine the two models. On this ensemble, the bigbird model weighs 0.7 and the other model weighs 0.3.

To identify the "Shallow Solution" category in the text, we also use an ensemble of two basic models. We use fine-tuned roberta and fine-tuned mpnet as basic models. We use a weighted average to combine the two models. On this ensemble, the roberta model weighs 0.7 and the other model weighs 0.3. We fine-tune the RoBERTa language model (Liu et al., 2019) for this classification to prepare the fine-tuned roberta model.

To identify the "Presupposition" category in the text, we use an ensemble of two basic models. We use fine-tuned bigbird and fine-tuned mpnet as basic models. We use a weighted average to combine the two models. On this ensemble, the bigbird model weighs 0.7 and the other model weighs 0.3 and the sum of the weights is one. In bigbird for this category, the error weight for class with "Presupposition" is 4 times that of class without "Presupposition".

To identify the "Authority Voice" and "Metaphor" categories in the text, the model structure is similar to the "Presupposition" detection model in the text. The only difference between the detection models of these categories is in the weights of the base models to create the ensemble model. In the weighted average for the "Authority Voice" category, the weights of the bigbird model and the mpnet model are 0.5 and 0.5. For the "Metaphor" category, the weight of these models are 0.6 and 0.4, respectively.

To identify the "Compassion" category in the text, we use an ensemble of three basic models. First, we combine the results of the two basic models with the weighted average. These basic models are fine-tuned bigbird and fine-tuned mpnet. We use a weight of 0.4 for the bigbird model and a weight of 0.6 for the other model. Then we combine the result of combining the previous two models with the prediction of the fine-tuned roberta model. We use a weighted average with a weight of 0.1 for the roberta model and we set the weight of the combination of the previous two models to 0.9.

We also use fine-tuned roberta to identify the "The Poorer The Merrier" category in the text.

Task 1 and Task 2 share the same input paragraphs and have different labels respectively. The reason we chose Task 1 fine-tuning models is the same as the reason for using Task 2 models. In addition to the Task 1 models, we also used the RoBERTa model for Task 2, which is the base model presented in the competition. In each category, all of these models are trained for classification, and we presented the best possible combination of these models as the final model. To determine the weights for creating the ensemble model, the performance of the constituent models has been considered. The model with better performance has more weight in the ensemble model.

3 Experimental Setup[1]

In this section, the structural details of the base models are given. All models are trained on the GPU of google colab[2] in normal account mode.

3.1 BERT+BiGRU model

In the Bi-GRU layer of this model, we use two layers. Set the dimension of the hidden layer vector to 256. The direction of one GRU (Chung et al., 2014) is the positive direction of the input sequence (from left to right), and the other is the reverse direction of the input sequence (from right to left). When feature extraction is performed on the input sequence, the GRUs in the two directions do not share the state. The state transition rules of GRU follow the transition occurrence between the same states. However, at the same moment, the output results of the GRUs in the two directions are spliced as the output of the entire Bi-GRU layer. We apply dropout to the output of this layer with a probability of 0.25.

We output the Bi-GRU layer result to the feed forward network. This feed forward network consists of a hidden layer with 256 neurons. We also set the maximum number of input tokens to 512. In the learning process of this model, we use 5 epochs. In learning phase, the error weight for PCL class is 2 times that of non-PCL class.

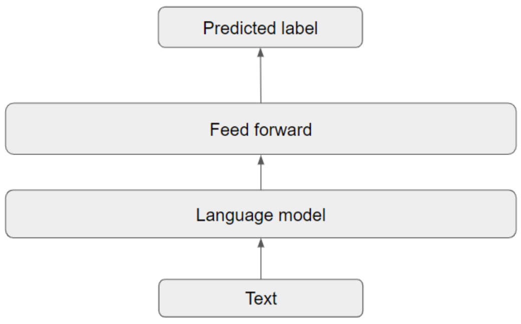


Figure 2 general structure of our model for fine-tuning the language model

3.2 Fine-tuned other Language Models

In the learning process of this model, we use 1 epoch. To fine-tune these language models, we use the ClassificationModel in the simpletransformers[3] library. The weight of the error in predicting the sample of class with label 1 can be different from the class with label 0.

4 Results and Analysis

4.1 Dataset

We use the SemEval 2022 task 4 dataset. We have three sets of training, evaluation, and testing in the competition data. The training dataset is imbalanced for both sub-tasks. Task 1 and Task 2 share the same input paragraphs and have different labels respectively. The maximum, mean, and median length of training texts are 5518, 294, and 258. Length means the number of characters. The maximum, mean, and median number of words in training texts are 911, 53, and 45.

To solve the problem of class imbalance in the learning process, we use the augmentation methods provided for toxic texts (Juuti et al., 2020). Among these methods, we use the EDA method for all binary classification problems. In some cases, we use other relevant datasets to increase minority class data.

In Task 1, we consider a constant difference of 4900 sample between the data number of the two classes. For classification, we paste the paragraph text, the keyword corresponding to the text, and the full name of the country associated with it, and consider it as the text for the classification. To reduce the data difference between the two classes, which is more than 4900, we add the texts of the Task 2 dataset that are not in the Task 1 dataset. We also add the first 100 texts of the talkdown dataset (Wang and Potts, 2019) to the collection. Wang and Potts (2019) introduced compassion modeling in direct communication from the perspective of natural language analysis. They created and tagged talkdown dataset with social media messages. Compassion is a type of PCL. For this reason, the use of compassion data helps detect the presence of PCL. All texts with a PCL type are added. We do not add the text itself, but we use the modified text

[1] Our code can be found at: https://github.com/AliEdalat/SemEval-2022-task-4-PCL-detection.git
[2] https://colab.research.google.com/

[3] https://simpletransformers.ai/docs/classification-models/

by substituting several words with the same meaning by using WordNet (Miller, 1995). We fill the rest of the difference between the two classes with two EDA methods. One way is to use modified texts that have PCL, by replacing some words with their synonyms in WordNet. Another way is to use modified texts that have PCL, by replacing some words with their nearest neighbours in Glove (Pennington et al., 2014) embedding space. Glove is a pre-trained word embedding that is trained on Twitter data.

For each category in Task 2, except for the "Metaphor" and "Compassion" categories, as in Task 1, we balance the classes. There are only two differences. We consider the difference between the two classes to be 5900 and do not use any other datasets to generate data. For the "Metaphor" category, the difference between the two categories is 5900. We get help from 1200 datapoints from vumc (Mu et al., 2019) dataset for balancing. Like the first task, we do not use this data itself and modify the text using WordNet. We fill in the rest of the gap like the other categories. For the "Compassion" category, we act like the "Metaphor" category. The only difference is the use of talkdown dataset instead of vumc.

With these methods, we prepare training data. To fine-tune non-BERT language models, we sample twice the amount of class with label 1 data from class with label 0. To predict the test data, we add the data that corresponds to the problem, with label 1 from the evaluation data to the training data.

4.2 Evaluation Metrics

We use competition metrics to evaluate system. For each binary classification, we use F1 score for label 1 as the evaluation metric. For Task 2, we use the mean F1 score for all categories as the evaluation metric.

4.3 Results

The results of the proposed model for Task 1 are given in the Table 2. In addition to the results of our model, the results of the baseline model of the competition are also included. The baseline model is fine-tuned roberta. This model uses sampling to balance the training dataset. In addition to our model, our model without using BERT+BiGRU in creating ensemble model is included ("(BigBird, MPNet)" shows this model in Table 2).

As you can see, our model did not perform well in the test and performed worse than the baseline due to overfit. But our model outperformed the baseline in evaluation. The reason for this overfit was the addition of the BERT+BiGRU model. As can be seen, if we remove BERT+BiGRU from the model, the model performs better in testing and evaluation than the baseline. BERT+BiGRU has contributed a little to the performance of the model in evaluation, but it has overfitted our model. So, the best model to solve this problem is “(BigBird, MPNet)”. Using this model, we achieved an F1-score of 0.5505.

Model	Eval F1	Test F1
(BigBird, MPNet, BERT+BiGRU) submitted system	0.5965	0.3031
(BigBird, MPNet)	0.5789	0.5505
Baseline	0.4829	0.4911

Table 2 The results of the proposed model for Task 1.

Category	Model	Eval f1	Test f1
Metaphor	**Our, submitted system (MPNet, BigBird 1:4)**	0.4557	0.1345
	PCM (MPNet, BigBird)	-	0.2947
Compassion	**Our, submitted system (BigBird, MPNet, RoBERTa)**	0.5565	0.1129
	PCM (BigBird, MPNet)	-	0.3932

Table 1 results of our models in the "Compassion" and "Metaphor" categories.

The performance of our model for Task 2 in evaluation and testing is given in the Table 3. The model presented in the Models section is called "Our Model" is listed in Table 3. The "Problem-free model in Compassion and Metaphor (PCM)" model is similar to our model, except that for the

"Compassion" category, we remove the fine-tuned roberta base model from its detection model. For the "Metaphor" category, we also set the weight of the class 1 detection error equal to the weight of the other class. As can be seen, our two models performed better than the baseline. The "PCM" model performs better than the "Our" model. Unfortunately, we did not use the "PCM" model in this competition. The reason for not using more models in two tasks was the restriction on uploading answers in the contest. Our F1-score in Task 2 of this competition was 0.2531. If we used the "PCM" model, our F1-score would be 0.3160.

The results of our two models in the "Compassion" and "Metaphor" categories are shown in the Table 1. BigBird 1:4 means that the two classes zero and one weigh one and four in the fine-tuned bigbird, respectively.

Model	Eval f1	Test f1
Our Model (submitted system)	0.3677	0.2531
Problem-free model in Compassion and Metaphor (PCM)	-	0.3160
Baseline	0.1340	0.1041

Table 3 The performance of our model for Task 2.

As can be seen, the performance of "Our Model" for the two classes in the test phase was very different from our performance in the evaluation phase. The performance of the "PCM" model in the test phase was much better than the Our Model. The difference in the performance of "Our Model" in the two stages of evaluation and testing in the "Compassion" category was due to the use of the fine-tuned model roberta as the base model. Using this model has caused overfit. The reason for the difference in performance in the "Metaphor" category was due to the different weight of the class with metaphor error compared to the class without metaphor in the bigbird model. This different weight has created a bias for our model as a whole. As can be seen, by solving these problems, the "PCM" model was able to perform better than “Our Model”

5 Conclusion

In this paper, we presented models for two tasks. For Task 1, we presented an ensemble model consisting of three basic models. We reviewed the results of this model in the competition. We examined the weaknesses of this model and presented another model with a similar structure that performed better on the test data. For Task 2, we considered identifying each category separately from the other categories. We provided a model to identify each category. We examined the result of our prediction based on these models in the competition and identified weaknesses. By solving these cases, we changed the classification model of the two categories. We were able to come up with a new prediction for test data that would have a better result than our original model. Using our second model resulted in better ranks in the testing phase. In future work for the second task, the categories can be considered related. This is because some categories have a common concept. The extracted features according to other categories, can be used to classify a category. In future work, other ways can be proposed to solve the problem of class imbalance. For example, constructor models can be used to create text for a class on a conditional basis.

References

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Zaheer, M., Guruganesh, G., Dubey, K.A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L. and Ahmed, A., 2020. Big bird: Transformers for longer sequences. Advances in Neural Information Processing Systems, 33, pp.17283-17297.

Song, K., Tan, X., Qin, T., Lu, J. and Liu, T.Y., 2020. Mpnet: Masked and permuted pre-training for language understanding. Advances in Neural Information Processing Systems, 33, pp.16857-16867.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V., 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

Juuti, M., Gröndahl, T., Flanagan, A. and Asokan, N., 2020. A little goes a long way: Improving toxic language classification despite data scarcity. Findings of the Association for Computational Linguistics: EMNLP 2020,.

Wei, J. and Zou, K., 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),.

Mu, J., Yannakoudakis, H. en Shutova, E. (2019) “Learning outside the box: Discourse-level features improve metaphor identification”, in Proceedings of the 2019 Conference of the North. Proceedings of the 2019 Conference of the North, Stroudsburg, PA, USA: Associa-tion for Computational Linguistics. doi: 10.18653/v1/n19-1059.

Pérez-Almendros, C., Espinosa-Anke, L. en Schockaert, S. (2020) “Don’t Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities”, in Proceedings of the 28th International Conference on Computational Linguistics, bll 5891–5902.

Wang, Z. and Potts, C., 2019. TalkDown: A Corpus for Condescension Detection in Context. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),.

Sap, M., Gabriel, S., Qin, L., Jurafsky, D., Smith, N.A. and Choi, Y., 2019. Social bias frames: Reasoning about social and power implications of language. arXiv preprint arXiv:1911.03891.

Price, I., Gifford-Moore, J., Flemming, J., Musker, S., Roichman, M., Sylvain, G., Thain, N., Dixon, L. and Sorensen, J., 2020. Six Attributes of Unhealthy Conversations. Proceedings of the Fourth Workshop on Online Abuse and Harms,.

Pérez-Almendros, C., Espinosa-Anke, L. en Schockaert, S. (2022) “SemEval-2022 Task 4: Patronizing and Condescending Language Detection”, in Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics.

Lees, A., Borkan, D., Kivlichan, I., Nario, J. and Goyal, T., 2021, April. Capturing Covertly Toxic Speech via Crowdsourcing. In Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing (pp. 14-20).

Miller, G.A., 1995. WordNet: a lexical database for English. Communications of the ACM, 38(11), pp.39-41.

Pennington, J., Socher, R. and Manning, C.D., 2014, October. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

Tesla at SemEval-2022 Task 4: Patronizing and Condescending Language Detection using Transformer-based Models with Data Augmentation

Sahil Manoj Bhatt **Manish Shrivastava**
Language Technologies Research Center (LTRC)
International Institute of Information Technology, Hyderabad
sahil.bhatt@research.iiit.ac.in
m.shrivastava@iiit.ac.in

Abstract

This paper describes our system for Task 4 of SemEval 2022: Patronizing and Condescending Language (PCL) Detection. For sub-task 1, where the objective is to classify a text as PCL or non-PCL, we use a T5 Model fine-tuned on the dataset. For sub-task 2, which is a multi-label classification problem, we use a RoBERTa model fine-tuned on the dataset. Given that the key challenge in this task is classification on an imbalanced dataset, our models rely on an augmented dataset that we generate using paraphrasing. We found that these two models yield the best results out of all the other approaches we tried.

1 Introduction

Detecting the presence of patronizing and condescending elements in text is an important task for NLP because of the social impact it has. PCL is characterised by a superior attitude towards others, or a manner of speech that seems to portray others in a pitying way. It is different from other problems in the field of text classification such as detection of hate speech or abusive comments because it is not necessarily done on purpose. Having an automated system that is capable of understanding and classifying language that contains PCL elements would be the first step towards making people and entities, such as media publications, aware of the kind of language they use when talking about vulnerable communities, and as a result, prevent discrimination, stereotypes and harm that could potentially arise from the use of such language.
The PCL Detection task at SemEval-2022 (Pérez-Almendros et al., 2022) aims to solve this problem by exploring different systems that are capable of detecting features that indicate and categorize PCL in the *Don't Patronize Me!* dataset (Pérez-Almendros et al., 2020). Sub-task 1 has been formulated as a binary classification problem, where the goal is to identify whether a given text falls under the category of PCL or not. Sub-task 2 is a multi-label classification problem, where a given text is either free of PCL or belongs to one or more of the seven PCL categories described in the dataset provided.
This paper describes the system developed by team Tesla for SemEval-2022 Task 4. One of the key challenges of this task is the unequal class distribution across the dataset for both sub-tasks.
In this paper, we introduce a system to detect PCL using i) a T5 (Raffel et al., 2019) model for subtask-1 and ii) a RoBERTa (Liu et al., 2019) model for sub-task 2. For our final submission, we use these two models finetuned on an augmented dataset, which we create by generating paraphrases of the sentences belonging to the minority classes in the dataset. Our system ranked 51st out of 78 teams in sub-task 1 and 27th out of 49 teams in sub-task 2, as shown in the leaderboard[1]. Our system for sub-task 2 outperforms the RoBERTa baseline, obtaining an average F1-score of 0.2445 versus 0.1041 for the baseline. All of our code is made publicly available on Github[2].

2 Task Description

The PCL detection task provides participants with the Don't Patronize Me! dataset (Pérez-Almendros et al., 2020), which consists of more than 10,000 paragraphs taken from English-language news stories across 20 different countries. Each of these paragraphs has been annotated to indicate the presence of PCL, and the dataset for sub-task 2 is further annotated with a category label from different classes proposed. These classes are focused on PCL towards vulnerable communities.
The datasets for both the sub-tasks are not balanced. The number of non-PCL examples is nearly 10

[1]https://sites.google.com/view/pcl-detection-semeval2022/ranking
[2]https://github.com/bhattsahil1/pcl_task

times more than the number of PCL instances in the dataset for sub-task 1. Similarly, the seven classes are not equally represented in the dataset for sub-task 2, and the number of examples that do not belong to any category exceeds those that belong to at least one category by a significant amount.

3 Related Work

Transformer-based approaches such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019), T5 (Raffel et al., 2019), etc. have shown an impressive performance on a wide range of NLP tasks, including text classification. They have been found to yield good results on text classification that deal with problems such as detection of hate speech (Basile et al., 2019) and offensive language (Zampieri et al., 2019).
The problem of condescending language detection is explored in the TalkDown Dataset (Wang and Potts, 2019), where the authors released an annotated Reddit corpus of condescending linguistic acts in context, along with a BERT model fine-tuned on the dataset as the baseline.
Data augmentation has been widely used across various machine learning tasks, particularly in those tasks where data is scarce, such as computer vision problems in the medical domain (Sundaram and Hulkund, 2021), (Sandfort et al., 2019). These augmentation approaches allow for a larger and more diverse training dataset.
With respect to text data augmentation, (Bayer et al., 2021) discusses various approaches, both in the data space and feature space. Techniques such as synonym-replacement (Wei and Zou, 2019), embedding replacement (Wang and Yang, 2015), SMOTE (Chawla et al., 2002), generative methods (Yu et al., 2016), (Radford et al., 2018) etc. have been studied for data augmentation.
Many recent works have also used back-translation (Corbeil and Ghadivel, 2020) and paraphrasing as a way to increase training data, such as data in the areas of dialogue-generation (Gao et al., 2020).

4 System Description

4.1 Data

The dataset for sub-task 1 consists of 10469 examples, containing the paragraph ID, article ID, keyword, country, paragraph text, binary PCL label, and original annotator label (on a scale of 0-4). The labels 2,3 and 4 are considered as examples containing PCL, whereas those having labels 0 and 1 are negative (non-PCL) examples. The distribution can be seen in Table 1. The dataset in sub-task 2 follows a similar format and consists of 9368 training examples. Here the label is a one-hot encoding of the seven classes to which the text may or may not belong. These classes are: *unbalanced power relations*, *shallow solution*, *presupposition*, *authority voice*, *metaphors*, *compassion* and *the poorer the merrier*. The major challenge that both these datasets pose is the lack of positive samples. The dataset, such as the one seen in Table 1, has a positive to negative class ratio of nearly 1:10. This presents difficulties in learning features that characterize PCL since many of the existing techniques don't perform well when there is class imbalance (Madabushi et al., 2020), which is noticeable if the training and test data are dissimilar.

Class	**Instances**
PCL	993
Non-PCL	9476

Table 1: Class distribution for sub-task 1 dataset

Class	**Instances**
Unbalanced power relations (unb)	1290
Shallow solution (sha)	356
Presupposition (pre)	386
Authority voice (aut)	422
Metaphors (met)	342
Compassion (com)	832
The poorer the merrier (the)	69
(None of the seven classes)	7581

Table 2: Class distribution for sub-task 2 dataset

4.2 Data Augmentation

One of the most common problems that are faced in classification problems is the lack of data across different classes. This is an even bigger problem in the case of text classification problems, since generating new samples is not a trivial task.
Undersampling the majority class would lead to a loss of negative samples and under-utilization of the given data, hence the approach we take is on the lines of oversampling minority classes to augment data. However, instead of directly oversampling minority samples, we use a T5 model[3] finetuned for the task of paraphrase generation on the PAWS

[3]https://huggingface.co/Vamsi/T5_Paraphrase_Paws

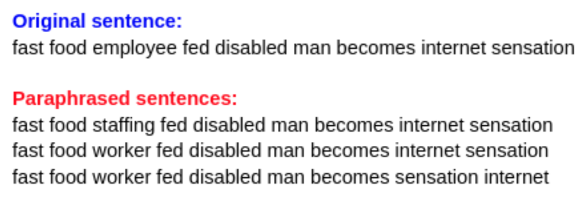
Class	Instances
PCL	8758
Non-PCL	9476

Table 3: Class distribution for sub-task 1 dataset after augmentation

Class	Instances
Unbalanced power relations (unb)	5114
Shallow solution (sha)	1413
Presupposition (pre)	1532
Authority voice (aut)	1682
Metaphors (met)	1361
Compassion (com)	3307
The poorer the merrier (the)	274
(None of the seven classes)	7581

Table 4: Class distribution for sub-task 2 dataset after augmentation

dataset (Zhang et al., 2019). The idea is to generate samples that are similar to the original text, but not the same, as we would like to avoid overfitting that could result from simple oversampling.
We use top-k sampling, in combination with top-p sampling, setting the values of k=120 and p=0.95. For sub-task 1, we generate a maximum of 8 samples for each of the paragraphs belonging to the PCL class. For sub-task 2, we generate a maximum of 3 samples for every sentence that belongs to at least one of the seven class labels. We finally get 18234 examples for sub-task 1 and 14667 examples for sub-task 2.


Original sentence:
fast food employee fed disabled man becomes internet sensation

Paraphrased sentences:
fast food staffing fed disabled man becomes internet sensation
fast food worker fed disabled man becomes internet sensation
fast food worker fed disabled man becomes sensation internet

Figure 1: An example of the paraphrases generated for one of the sentences in the dataset (after pre-processing).

4.3 Pre-processing

We remove punctuation and numbers from our text, and convert each of our sentences to lowercase before using them to generate paraphrases. We remove stop-words from the sentences after carrying out data augmentation.

5 Models

We fine-tune a range of pretrained models, described below, given their good performance on a wide range of NLP tasks. We use the transformer model implementations provided by `simpletransformers`[4] with the default hyperparameters.
BERT (Devlin et al., 2019): We try out the BERT$_{\text{BASE}}$ (uncased) model, which consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.
RoBERTa (Liu et al., 2019): We try out the RoBERTa$_{\text{BASE}}$ model. Similar to BERT$_{\text{BASE}}$, RoBERTa$_{\text{BASE}}$ also consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.
T5 (Raffel et al., 2019): We use the Text-to-Text Transfer Transformer (T5) released by the authors. We use the T5$_{\text{BASE}}$ model, which has about 220 million parameters, nearly twice the number of parameters in BERT$_{\text{BASE}}$.
We test all three models for sub-task 1, and for sub-task 2 we only try out BERT$_{\text{BASE}}$ and RoBERTa$_{\text{BASE}}$.

6 Experiments

6.1 Implementation details

For both the sub-tasks, we concatenate the paragraph text and the keyword associated with each sample, to explore if certain keywords have an effect on the sample being classified as PCL or not.
We train each of the models with two different conditions - using a dataset without paraphrased instances (original dataset) and a dataset with the original and the paraphrased sentences (augmented dataset). We train them for a single epoch only, to avoid overfitting since the data available is less.
We use an 80:20 train-dev split, preserving the class ratio. We use this to evaluate our model performance in sub-task 1. For sub-task 2 too, we use an 80:20 split for training and evaluating the model.
For our final submissions, we train the models for both sub-task 1 and sub-task 2 on the entire dataset.

6.2 Metrics

We report the F1-score, Precision and Recall for sub-task 1. For sub-task 2, we report the F1-score across each of the classes, along with the average score.

[4] https://simpletransformers.ai/

7 Results

Before submitting the final models, we evaluated each of the models' performances on validation sets.

Table 5 presents a comparison between all the different models tried out in sub-task 1. The $T5_{BASE}$ model yields the best F1-score among models trained on the original dataset, and the model also seems to perform decently well on the augmented dataset. The results presented in the table for models trained on the augmented dataset for both the sub-tasks are high since the validation set also contains paraphrases of sentences it might have already seen in the training set. Nevertheless, the relative scores between models trained on the augmented dataset are still a good indicator of their expected performance.

Table 6 discusses the results obtained using $BERT_{BASE}$ and $RoBERTa_{BASE}$. The average F1-score reported on the validation set does not change significantly across the four models used, however, the F1-scores for individual classes are significantly different when comparing the models trained on the original dataset versus models trained on the augmented one.

We present the results of our final submissions in Table 7 and 8.

8 Discussion and Conclusion

The $T5_{BASE}$ models that we submitted for sub-task 1 do not perform well on the test set. One reason for this could be that the $T5_{BASE}$ should have been trained for longer than one epoch. Another reason could be that BERT-based approaches (BERT, RoBERTa) might be better suited for this task. In addition to this, the results of $T5_{BASE}$ fine-tuned on the augmented dataset are not very good either, which could be due to overfitting that results from seeing many paraphrased instances of the same sentence during training. A manual inspection of the generated paraphrases also reveals that their quality needs improvement, since many of the generated paraphrases do not differ much from each other, and at times the generation gets reduced to simple oversampling.

The results of sub-task 2 are encouraging and we can see a significant improvement in F1-scores across almost all classes when we consider the $RoBERTa_{BASE}$ model trained on the augmented dataset.

This confirms that augmenting the dataset through paraphrasing does have a positive effect on model performance. Ensuring dissimilarity in the generated paraphrases, choosing an ideal number of paraphrases to generate, and using other techniques to handle imbalanced data such as cost-sensitive learning or augmentation through other methods (or a combination of them), might yield better results.

We thus, see that identifying PCL and categorizing its occurrences is feasible despite its subjective nature, and that transformer-based approaches are capable of doing this.

Acknowledgments

We thank the organisers of this task for their effort, and the anonymous reviewers for their feedback and insightful comments.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2021. A survey on data augmentation for text classification.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Jean-Philippe Corbeil and Hadi Abdi Ghadivel. 2020. BET: A backtranslation approach for easy data augmentation in transformer-based paraphrase identification context. *CoRR*, abs/2009.12452.

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

Silin Gao, Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Paraphrase augmented task-oriented dialog generation.

Model (Sub-task 1)	Metrics (validation set results)		
	Recall	Precision	F1-score
BERT base (original dataset)	0.578	0.789	0.609
RoBERTa base (original dataset)	0.500	0.452	0.475
T5 base (original dataset)	0.694	0.703	0.699
BERT base (augmented dataset)	0.939	0.938	0.938
RoBERTa base (augmented dataset)	0.878	0.879	0.877
T5 base (augmented dataset)	0.870	0.879	0.866

Table 5: Sub-task 1: The results mentioned here are for the validation set from an 80:20 training-validation split of the dataset (with and without augmentation). Do note that the validation set for the original dataset and augmented dataset is different, which is why we are only looking at the relative performance of models on the augmented dataset and not comparing it with models trained on the original dataset.

Model (Sub-task 2)	F1 scores (validation set results)							
	unb	sha	pre	aut	met	com	the	avg
BERT base (original dataset)	0.404	0	0	0	0	0.118	0	0.075
RoBERTa base (original dataset)	0.485	0	0	0	0	0.105	0	0.084
BERT base (augmented dataset)	0.196	0.025	0.079	0.044	0.027	0.116	0	0.070
RoBERTa base (augmented dataset)	0.187	0.039	0.081	0	0.031	0.135	0	0.067

Table 6: Sub-task 2 : The results mentioned here are for the validation set from an 80:20 training-validation split of the dataset (with and without augmentation). The classes mentioned here are abbreviations of classes discussed in Table 2. Do note that the validation set for the original dataset and augmented dataset is different, which is why we are only looking at the relative performance of models on the augmented dataset and not comparing it with models trained on the original dataset.

Model (Sub-task 1)	Metrics		
	Recall	Precision	F1-score
RoBERTa baseline	0.653	0.3935	0.4911
T5 base (original dataset)	0.691	0.283	0.402
T5 base (augmented dataset)	0.577	0.359	0.443

Table 7: Final results for sub-task 1

Model (Sub-task 2)	F1 scores							
	unb	sha	pre	aut	met	com	the	avg
RoBERTa baseline	0.3535	0	0.1667	0	0	0.2087	0	0.1041
RoBERTa base (original dataset)	0.286	0	0	0	0	0	0	0.041
RoBERTa base (augmented dataset)	0.437	0.383	0.163	0.192	0.179	0.357	0	0.2445

Table 8: Final results for sub-task 2

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

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

Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle. 2020. Cost-sensitive BERT for generalisable sentence classification with imbalanced data. *CoRR*, abs/2003.11563.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Veit Sandfort, Ke Yan, Perry J. Pickhardt, and Ronald M. Summers. 2019. Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks.

Shobhita Sundaram and Neha Hulkund. 2021. Gan-based data augmentation for chest x-ray classification.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2016. Seqgan: Sequence generative adversarial nets with policy gradient. *CoRR*, abs/1609.05473.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling.

SSN_NLP_MLRG at SemEval-2022 Task 4: Ensemble Learning strategies to detect Patronizing and Condescending Language

Kalaivani Adaikkan and **Durairaj Thenmozhi**
Department of Computer Science and Engineering
Sri Sivasubramaniya Nadar College of Engineering, Chennai, India
kalaiwind@gmail.com, theni_d@ssn.edu.in

Abstract

In this paper, we describe our efforts at SemEval 2022 Shared Task 4 on Patronizing and Condescending Language (PCL) Detection. This is the first shared task to detect PCL which is to identify and categorize PCL language towards vulnerable communities. The shared task consists of two subtasks: Patronizing and Condescending language detection (Subtask A) which is the binary task classification and identifying the PCL categories that express the condescension (Subtask B) which is the multi-label text classification. For PCL language detection, We proposed the ensemble strategies of a system combination of BERT, Roberta, Distilbert, Roberta large, Albert achieved the official results for Subtask A with a macro f1 score of 0.5172 on the test set which is improved by baseline score. For PCL Category identification, We proposed a multi-label classification model to ensemble the various Bert-based models and the official results for Subtask B with a macro f1 score of 0.2117 on the test set which is improved by baseline score.

1 Introduction

Social media is a wide platform and it grows rapidly. People can communicate with each other and express their opinions easily without any hesitation on social media. Patronizing and Condescending Language (PCL) [1] is language use shows a superior attitude rise towards vulnerable communities in the social media. This effect is unconscious and the intention is trying to help communities like an individual, group of people in need by raising awareness, moving the user audience to action, and standing for the rights. However, these dominant attitudes can lead to discrimination and the user audience is unaware of this diminishing treatment due to its subtlety. Moreover, online social media publications reached more audiences in day-to-day life and we noticed that diminishing treatment of vulnerable groups leads to greater inequalities. so, PCL can potentially be very harmful, as it feeds stereotypes, routinizes discrimination, and drives to greater exclusion.

Detecting PCL language and its categorization of PCL language on social media have gained a lot of interest recently. The detection of PCL is still an emergent area of study in NLP. To the best of my knowledge, this is the first shared task [2] to detect the PCL and their categories from the vulnerable communities. The challenge is to detect that PCL is difficult both for humans and NLP systems, due to its subtle nature, and its subjectivity reasoning required to understand this kind of language. SemEval 2022 task 4 presents the problems of detecting PCL and its categorizes of PCL which express the condescending language in English tweets to the NLP community. The PCL shared task consists of two subtasks: Subtask A is to identify the content is PCL or Not PCL. Subtask B is to classify whether the content into Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier.

This paper describes the systems submitted for SemEval 2022 shared task on PCL Detection by the team SSN_NLP_MLRG. We have participated in the shared task for all the two subtasks. First, we experimented with Bert-based models and we used the ensembling strategies to enhance the performance of the model. Finally, we performed voting to decide the final output. The majority voting on 5 classification models yielded better results than individual systems. The paper is organized as follows: In section 2, we describe the Background work, in section 3, we describe our models, we present the experimental setup in section 4, and compare results in section 5, we provide the conclusion of our work.

[1] https://www.merriam-webster.com/dictionary/patronizing

[2] https://competitions.codalab.org/competitions/34344

2 Background

In this section, we describe the task provided to the participants and the two subtasks.

2.1 Task Description

The participants were required to produce labels indicating if a paragraph is PCL or Not PCL in the shared of subtask A, and we categorize the PCL Language in the shared task subtask B (P'erez-Almendros et al., 2022).

Subtask A is the binary classification task. Each text content took one of these labels for subtask A as follows: PCL Language: The content shows a superior attitude and language towards a vulnerable community in media. Not PCL: The content is not intended for the PCL Langauge

Subtask B is the multi-label text classification task. Each text content took into these categories of labels for subtask B as follows: Unbalanced power relations: The author keeps distance from the community or based on the situation they express the will, capacity, or responsibility to help people in need. The author also presents to give something positive to the audience in a more vulnerable situation, especially the author concedes is a right but they do not have any authority to decide to give. Shallow solution: A superficial charitable and simple action by the privileged community which is presented either as life-saving or life-changing or as a solution for a deep-rooted problem. Presupposition. The author assumes a situation as certain without having all the valid information and trustworthy source for it (e.g. a survey of research work). Examples of presupposition such as usage of stereotypes or cliches.

Authority voice: The author stands themselves as a superior power of the group, or advises the members of a community about the specific situation they are living. Metaphor. They can conceal PCL, making a comparison between unrelated concepts. An example of a metaphor is euphemisms. Compassion. The author shows the vulnerable individual or group of people about raising a feeling of pity and compassion from the audience towards them. It is commonly characterized by the use of flowery vulnerable words. The poorer, the merrier. How they spread a positive attribute towards the vulnerable community. People learn to live in vulnerable situations and to admire their values. The typical example of 'poor people is happier because they don't have material goods. Table 1 presents the sample annotated data.

2.2 Related work

The authors (P'erez-Almendros et al., 2020) described a new annotated PCL dataset which is aimed to identify and categorize language that is patronizing or condescending language towards vulnerable communities and used the Bert model to detect and classify the harmful PCL language. Recently, Several works are carried out to detection of offensive language (Kalaivani and Thenmozhi, 2020a), hate speech (Kalaivani and Thenmozhi, 2020b), fake news detection, trustworthiness (Atanasova et al., 2018) and fact-checking (Elsayed et al., 2021) prediction is driven towards a particular community. The work (Fiske, 1993) presents a theory of the power of stereotyping and controlling the power of other outcomes. The author (Giles et al., 1993) analyzed the effects of responses and attitudes based on age group towards patronizing and harmful speech. Discourse analysis promises the need to satisfy the teacher's, student's textual values that build on techniques and provide a smoother relationship (Huckin, 2002). Margić (2017) examined the communication courtesy or condescending between the native and non-native English speakers. We observed that most of the work is related to the unfair treatment of the particular underprivileged community.

3 Methodology

We used the pre-trained models BERT (Bidirectional Encoder Representations from Transformers). We fine-tune a BERT language model (Devlin et al., 2019) for PCL classification. we also fine-tuned a RoBERTa-base (A robustly optimized Bert pretraining approach) model (Liu et al., 2019) to classify PCL Language and the PCL categories which can be expressed condescension and viewed as an optimized version of BERT. We used two variants of the RoBERTa method that are RoBERTa-large-cased and RoBERTa-base-cased pre-trained models. We also fine-tune the DistilBERT (Distilled version of BERT model) model (Sanh et al., 2019) is the transformer model, which is a lighter and faster variant of BERT. We used the AlBERT (A lite BERT) model (Lan et al., 2019) to fine-tune the system to predict the PCL language. To further explore the performance, we apply the Ensemble strategies to combine the transformers models output to predict the PCL and Category of PCL

Text	Subtask A	Subtask B
1. The scheme saw an estimated 150,000 children from poor families being sent to parts of the British Empire between 1920 and 1974 , by religious orders and charities who said they would lead better lives	**1**	**[1, 0, 0, 1, 0, 0, 0]**
2. Durban 's homeless communities reconciliation lunch	**0**	**[0, 1, 0, 0, 0, 0, 0]**

Table 1: Sample annotated paragraph. For subtask A, '0' presents PCL and '1' presents Not PCL. For subtask B, Seven category of PCL are Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier

language which is based on the majority voting concept. In all cases, we trained the model for 10 epochs. Finally, we got a macro-average f1 score of 0.5172 for subtask A and f1 average score of 0.2117 for subtask B respectively.

4 Experimental setup

4.1 Data description

The dataset for SemEval 2022 Shared task 4 consists of 10,469 paragraphs are split into training, development, and testing sets for subtask A and 993 unique paragraphs, totaling 2,760 instances of PCL, for Subtask B . Don't patronize me dataset offers content from media forums. the training data size is 8,375 contents and the development data size is 2,095 contents and the size of the test data is 3,832 contents. Table 1 presents the split of experiment data. The shared task of subtask A is a binary classification task in which the aim is to build systems able to detect the given paragraph content is PCL or Not PCL. The shared task of the PCL Category classification is a Multi-label classification task in which the aim is to build systems able to classify the PCL category into Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier.

4.2 Data Preprocessing

We applied down sampling negative instances data augmentation techniques to balance the dataset because the negative instances are 7,581 contents and positive instances are only 794 contents. Preprocessing the text is an important role as the data from social media can be quite noisy and contain a lot of noisy words, excessive use of punctuation, URLs, symbols, misspelling words. We perform data preprocessing by using NLTK libraries. First, we remove the duplication because it affects the system performance. we remove the stop words. After that, we remove the punctuations, URLs, numerals, emojis and then convert all the upper case English text into lower case text.

4.3 Experimental setting

For both subtasks A and B, We implement the Ensemble model using Simple transformers. We used the colab notebook for implementation purposes with the high-end RAM, GPU for training. For the hyperparameters for the BERT-based five models, we set epochs as 10. For the multilabel classification task, we used simple transformers and a multi-label classification model to predict the PCL Language category. we analyzed the individual classification of all five BERT-based models for both the subtasks. We also examined the final output which is the combination of five models based on a majority voting system for classification.

5 Experimental analysis

This section presents the analysis of the results and submitted official results

5.1 Result Analysis

We experimented with the various transformer model are BERT, DistilBERT, AlBERT, Roberta base, Roberta large, and the ensemble of all five models. We analyzed the comparison scores of various approaches based on the evaluation metrics of precision, recall, and macro average f1 score for the shared task A. Task 1 is a binary classification task that will be evaluated using f1 over the positive class. Task 2 is framed as a multi-label classification problem. For each paragraph, your model will assign a label for each of the seven PCL categories. Then, results for this task will be evaluated using per-class f1, and the final ranking for this subtask will be based on macro-average f1. Table 2 presents the results of subtask A. Table 3 shows the results of subtask B.

Model	Precision	Recall	Macro f1
Run 1:			
BERT model	0.3832	0.6940	0.4938
Run 2:			
Ensemble model	0.4228	0.6656	0.5171

Table 2: Test Results of subtask A

Model	UNB POW	SHAL	PRES	AUTH	MET	COMP	MERR	Avg f1
Run 1:								
BERT	0.3459	0.3376	0.2068	0.1933	0.1212	0.2772	0.0	0.2117
Run 2:								
AlBERT	0.3438	0.3157	0.2056	0.1666	0.1666	0.2677	0.0	0.2094

Table 3: Test Results of subtask B. UNB POW - Unbalanced power relations, SHAL - Shallow solution, PRES - Presupposition, AUTH - Authority voice, MET - Metaphor, COMP - Compassion, MERR - Poorer and Merrier

We submitted two runs for both of the subtasks. For run 1, we submitted the prediction made by the BERT model for subtasks A and B. For run 2, we submitted the prediction made from the ensemble model for subtask A and Albert model for subtask B. We observed that the performance of the Ensemble model achieved good results compared to the BERT model for subtask A and the performance of the BERT model achieved good results compared to the Albert model. Finally, we got the macro f1 score of ensemble model is 0.5172 for the subtask A and the macro f1 average of Bert model is 0.2117 for the subtask B respectively.

6 Conclusion and Future Work

This paper presents the submitted runs for the patronizing and condensing language identification in SemEval 2022 task 4. The results show that the Not-PCL language and PCL language in the dataset receives the same macro f1 scores. We experimented with different approaches such as a BERT model, AlBERT, Roberta base, Roberta large, and Distilbert and Ensemble models. Based on the evaluation, BERT performs well for subtask B to classify the PCL content into Seven categories that express condensing language. Ensemble model performs well for subtask A to detect the content is PCL language or Not PCL language. Our team submission had a macro f1 score of 0.5172 for subtask A and a macro f1 score of 0.2117 for subtask B which are improved by the baseline f1 scores. For future work, we will handle the imbalanced dataset by using external resources and apply the data augmentation techniques to enhance the performance of our model.

References

Pepa Atanasova, Alberto Barrón-Cedeño, Tamer Elsayed, Reem Suwaileh, Wajdi Zaghouani, Spas Kyuchukov, Giovanni Da San Martino, and Preslav Nakov. 2018. Overview of the CLEF-2018 checkthat! lab on automatic identification and verification of political claims. task 1: Check-worthiness. *CoRR*, abs/1808.05542.

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

Tamer Elsayed, Preslav Nakov, Alberto Barrón-Cedeño, Maram Hasanain, Reem Suwaileh, Giovanni Da San Martino, and Pepa Atanasova. 2021. Overview of the CLEF-2019 checkthat!: Automatic identification and verification of claims. *CoRR*, abs/2109.15118.

Susan Fiske. 1993. Controlling other people: The impact of power on stereotyping. *The American psychologist*, 48:621–8.

Howard Giles, Susan Fox, and Elisa Smith. 1993. Patronizing the elderly: Intergenerational evaluations. *Research on Language and Social Interaction*, 26(2):129–149.

Thomas N. Huckin. 2002. Critical discourse analysis and the discourse of condescension.

A Kalaivani and D Thenmozhi. 2020a. SSN_NLP_MLRG at SemEval-2020 task 12: Offensive language identification in English, Danish, Greek using BERT and machine learning approach. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2161–2170, Barcelona (online). International Committee for Computational Linguistics.

A. Kalaivani and D. Thenmozhi. 2020b. SSN_NLP_MLRG@HASOC-FIRE2020: Multilingual Hate Speech and Offensive Content Detection in Indo-European Languages using ALBERT. In *Working Notes of FIRE 2020 - Forum for Information Retrieval Evaluation, Hyderabad, India, December 16-20, 2020*, volume 2826 of *CEUR Workshop Proceedings*, pages 188–194. CEUR-WS.org.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

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

Branka Drljaa Margić. 2017. Communication courtesy or condescension? linguistic accommodation of native to non-native speakers of english. *Journal of English as a lingua franca*, 6:29–55.

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Sapphire at SemEval-2022 Task 4: A Patronizing and Condescending Language Detection Model Based on Capsule Networks

Sihui Li
Yunnan University, Yunnan, P.R. China
sihui_li@mail.ynu.edu.cn

Xiaobing Zhou *
Yunnan University, Yunnan, P.R. China
zhouxb@ynu.edu.cn

Abstract

This paper introduces the related work and the results of Team Sapphire's system for SemEval-2022 Task 4: Patronizing and Condescending Language Detection. We only participated in subtask 1. The task goal is to judge whether a news text contains PCL. This task can be considered as a task of binary classification of news texts. In this binary classification task, the BERT-base model is adopted as the pre-trained model used to represent textual information in vector form and encode it. Capsule networks is adopted to extract features from the encoded vectors. The official evaluation metric for subtask 1 is the F1 score over the positive class. Finally, our system's submitted prediction results on test set achieved the score of 0.5187.

1 Introduction

Patronizing and Condescending Language (PCL) can be considered when someone's language has a superior attitude towards others, demeans others, or describes the situation of others in a compassionate way. Such expressions are often unconscious, and are used by people to try to induce action or raise awareness. Because of its subtlety and often well-meaning when used, users often overlook the demeaning elements of this expression. Such elements may contribute to the stereotyped influence of society on a group, making discrimination normalized and even leading to stronger exclusion (Pérez-Almendros et al., 2022).

Detecting PCL in media text is a challenging task. Recognizing PCL based on Natural Language Processing (NLP) can alert speakers to examine the rationality of their speeches, so that speeches can be more inclusive and constructive, which in turn leads to more responsible communication.

When processing corpus, the pre-trained model can convert text information into vector representation, making it more suitable for NLP tasks. Early pre-trained models were designed to learn representational word embeddings, such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). Although such methods can capture the semantics of words through word embeddings, they cannot capture the concepts in the context. With the introduction of new technologies, there are now pre-trained models that can learn to represent contextual word embeddings, such as the ELMo (Peters et al., 2018) model based on LSTM (Shi et al., 2015) and the BERT (Devlin et al., 2018) model based on Transformer Encoder (Vaswani et al., 2017).

In recent years, using deep neural networks in NLP, such as Convolutional Neural Networks (CNNs) in text classification (Kim, 2014), has become mainstream. Capsule networks (Sabour et al., 2017), as a structure proposed on the basis of CNNs to improve spatial sensitivity in computer vision, is also used in text classification tasks (Yang et al., 2019; Ding et al., 2019). Kim et al. (2020) further suggest a simple routing method that effectively reduces the computational complexity of dynamic routing.

This task aims to predict whether each news text for each ID contains PCL. Text is represented as a vector and encoded using a pre-trained BERT model. It mainly uses the capsule networks to extract the encoded vector, and uses the output of the fully connected layer to represent the label probability. The rest of the paper is organized as follows: Section 2 introduces the system architecture. Section 3 describes the dataset, implementation details settings and experimental results. The summary and outlook for future work will be presented in Section 4.

2 System Architecture

In this section, we will introduce the system architecture we use in the task, which will consist of two parts. One is embedding and coding, and the other is feature extraction and prediction. We call the model that is ultimately used to test the class prediction results of the dataset as BERT-Caps. The architecture of BERT-Caps model is shown in Figure 1.

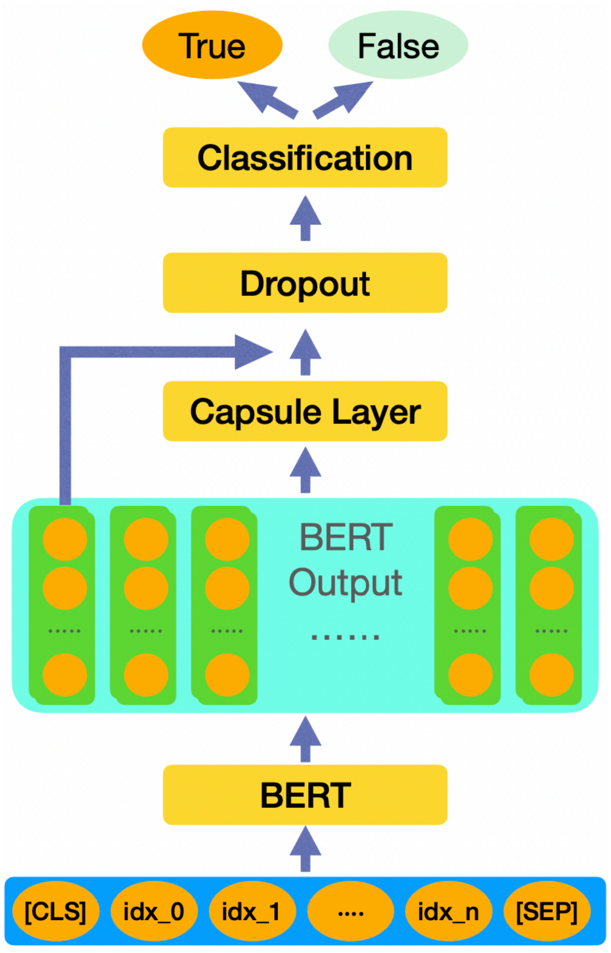


Figure 1: The architecture of the BERT-Caps model.

2.1 Embedding and Encoding

When using the early pre-trainied model for text classification, the text is usually represented as a word embedding and then the vector matrix is sent to a bidirectional recurrent network for encoding to improve the system's ability to perceive contextual information. In this paper, we mainly use the BERT-base model to represent text into vector form and encode it.

In order to fit the pre-trained model, we need to preprocess the text in the dataset accordingly. As standard news text, we do only a little text processing on the news text: unify the text to lowercase. Add markers to the beginning and end of the text. For example, when using BERT as the pre-trained model, [CLS] and [SEP] will be used to mark the beginning and end of the text, respectively. Then according to the dictionary information, the words are converted into a list of their position numbers in the dictionary. Collate to get a list that marks the beginning and the end of each sentence.

This part of the work is mainly achieved through the tokenizer attached to the module used when importing the pre-trained model. For the imported model, we set the trainable value of each layer of the model to True.

2.2 Feature Extraction and Predict

We use the capsule networks to perform feature extraction on the hidden state of the last layer of the pre-trained model. In the capsule layer, the input is firstly processed by the Conv1d function. The convolution output is treated as a set of capsules, and a new set of capsules of the specified shape is derived through the dynamic routing algorithm. The result is the output of the capsule layer.

The flattened capsule layer output and the text vector corresponding to the first bit in the pre-trained model are linked together. In order to improve the generalization ability of the model, dropout is used. During training, the concatenated outputs are first processed by dropout and then fed into the fully connected layer to predict the probability that the news text has PCL. The loss function of this model adopts categorical crossentropy.

3 Experiment and Result

3.1 Dataset and Official Evaluation Metrics

The dataset used in the experiment is provided by SemEval-2022 Task4, Patronizing and Condescending Language Detection.(Pérez-Almendros et al., 2020)

In this dataset, the degree of PCL is divided into five levels from 0 to 4. In subtask 1, the level of 0-1 is regarded as a negative example, and 2-4 is regarded as a positive example. Participants were asked to predict the presence or absence of PCL component in the text. The differentiated test set contains 9476 negative labels and 993 positive labels, almost reaching 10:1. Due to the imbalance of samples in the dataset, the F1 score over the positive class was adopted as the official evaluation metric.The formula for F1 score is as following:

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{1}$$

Precision means the ratio of correctly predicted positive observations to the total predicted positive observations. Recall means the ratio of correctly predicted positive observations to all observations in the real class.

3.2 Implementation Details

In terms of data segmentation, we import the train_test_split function from the Scikit-learn(Pedregosa et al., 2011) module to divide the dataset into training set and validation set, set *test_size* to 0.2, *random_state* to 35.

All experiments in this paper are based on using the TensorFlow2 backend.

When using BERT-base-uncased[1] as the pre-trained model, we use the Keras-BERT (Shorten and Khoshgoftaar, 2021) module to implement the Tokenizer and import the model.

We also tried other BERT-based models, such as RoBERTa-base and DeBERTa-base. When implementing Tokenizer and importing models, we use the Transformers (Wolf et al., 2020) module.

The number of capsules, the number of hidden neurons, and the number of iterations of the dynamic routing algorithm are set to 10, 64, and 3, respectively.

The fully connected layer that outputs the final result in each model uses softmax as the activation function. The hyperparameters used are mentioned in Table1.

Parameters	**subtask 1**
Epochs	8
Batch_size	8
Max_length	128
Drop_rate	0.25
Optimizer	Adam
Initial *lr*	1e-5

Table 1: Hyperparameters

In actual training, in order to alleviate the overfitting situation, ReduceLROnPlateau is introduced. Also set ModelCheckpoint to save each model with the smallest loss on the existing basis.

3.3 Experiment and Result

The system uses the dataset provided by the task organizer for training. The BERT-Caps model that finally gets the submitted prediction results is saved at the end of the 8th epoch training.

The results are shown in Table 2. The values of RoBERTa_baseline comes from the result published on the Competition Page[2]. As can be seen from the table, as a result, our model has a greater improvement in precision than the baseline. We also tried to train some BERT-Caps models that reduced the number of capsules in the capsule layer and increased the number of hidden neurons, but there was no significant improvement in metric.

Table 2 shows that without the capsule networks, the performance of the model will be greatly reduced compared to the original model. Without dropout, the prediction performance decreases less than without the capsule networks.

We also tried to keep almost the same system architecture, only replacing the pre-trained model and tokenizer. Unexpectedly, in the experimental environment of this paper, both DeBERTa-Caps and RoBERTa-Caps are not as good as BERT-Caps.

The best test set predictions submitted by our team were produced by the BERT-Caps model. Considering with the F1 scores obtained by the top four teams in the English data are all over 0.6400, indeed, there is a gap. Team Sapphire's final ranking is 35th.

4 Conclusion

This paper describes the experiments conducted by Team Sapphire in subtask 1 of SemEval-2022 Task 4: Patronizing and Condescending Language Detection. We introduced the system architecture, experimental dataset situation and results in Section 2 and 3, respectively. From the experimental results, the BERT-Caps model can achieve better results on the test set. In future work, we will improve our method to achieve better results. For example, using other text representations, and adjusting the weight of the loss function.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yunxia Ding, Xiaobing Zhou, and Xuejie Zhang. 2019. Ynu_dyx at semeval-2019 task 5: A stacked bigru model based on capsule network in detection of hate. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 535–539.

[1]https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip

[2]https://sites.google.com/view/pcl-detection-semeval2022/ranking

Model	Precision	Recall	F1 score
BERT-Caps	0.5935	0.4606	0.5187
BERT-Caps w/o dropout	0.6	0.4542	0.5170
BERT-Caps w/o capsule networks	0.5782	0.4195	0.4862
RoBERTa-Caps	0.625	0.3943	0.4835
DeBERTa-Caps	0.5870	0.4574	0.5141
RoBERTa_baseline	0.3935	0.6530	0.4911

Table 2: Results: subtask 1

Jaeyoung Kim, Sion Jang, Eunjeong Park, and Sungchul Choi. 2020. Text classification using capsules. *Neurocomputing*, 376:214–221.

Y. Kim. 2014. Convolutional neural networks for sentence classification. *Eprint Arxiv*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Matthew Peters, M. Neumann, M. Iyyer, M. Gardner, and L. Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. *Advances in neural information processing systems*, 30.

Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28.

Connor Shorten and Taghi M Khoshgoftaar. 2021. Kerasbert: Modeling the keras language. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 219–226. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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

Min Yang, Wei Zhao, Lei Chen, Qiang Qu, Zhou Zhao, and Ying Shen. 2019. Investigating the transferring capability of capsule networks for text classification. *Neural Networks*, 118:247–261.

McRock at SemEval-2022 Task 4: Patronizing and Condescending Language Detection using Multi-Channel CNN, Hybrid LSTM, DistilBERT and XLNet

Marco Siino, Marco La Cascia, and **Ilenia Tinnirello**
Università degli Studi di Palermo, Palermo, Italy
{marco.siino, marco.lacascia, ilenia.tinnirello}@unipa.it

Abstract

In this paper we propose four deep learning models for the task of detecting and classifying Patronizing and Condescending Language (PCL) using a corpus of over 13,000 annotated paragraphs in English. The task, hosted at SemEval-2022, consists of two different subtasks. The Subtask 1 is a binary classification problem. Namely, given a paragraph, a system must predict whether or not it contains any form of PCL. The Subtask 2 is a multi-label classification task. Given a paragraph, a system must identify which PCL categories express the condescension. A paragraph might contain one or more categories of PCL. To face with the first subtask we propose a multi-channel Convolutional Neural Network (CNN) and an Hybrid LSTM. Using the multi-channel CNN we explore the impact of parallel word emebeddings and convolutional layers involving different kernel sizes. With Hybrid LSTM we focus on extracting features in advance, thanks to a convolutional layer followed by two bidirectional LSTM layers. For the second subtask a Transformer BERT-based model (i.e. DistilBERT) and an XLNet-based model are proposed. The multi-channel CNN model is able to reach an F1 score of 0.2928, the Hybrid LSTM model is able to reach an F1 score of 0.2815, the DistilBERT-based one an average F1 of 0.2165 and the XLNet an average F1 of 0.2296. In this paper, in addition to system descriptions, we also provide further analysis of the results, highlighting strengths and limitations. We make all the code publicly available and reusable on GitHub[1].

1 Introduction

With the exponential growth of contents shared on social networks, a lot of new challenging tasks have emerged. Many are currently studied and addressed by scholars, and a pletora of novel machine learning approaches have been proposed (Arpaci et al., 2021), (Hosseinalipour and Ghanbarzadeh, 2022), (Siino et al., 2020). Some of the most common tasks, often co-located with international conferences, are those about fake news (Rangel et al., 2020), hate speech (Bosco et al., 2018), misogyny (Fersini et al., 2018) and cyberbulling (Kumar et al., 2018) detection.

For these purposes there is a constantly growing need for tools that can automatically extract and classify information from online feeds, to face with consolidated as well as with emerging social issues. Interest in Natural Language Processing (NLP) has increased in recent years with advances in machine and deep learning architectures. There have been significant efforts in developing methods to automatically detect and classify text content available online nowadays.

Together with the already mentioned tasks, an emerging one is about detecting Patronizing and Condescending Language (PCL) (Pérez-Almendros et al., 2020). The PCL Detection Task hosted at SemEval-2022 is covered in detail in (Pérez-Almendros et al., 2022) and briefly discussed here. The main task is made of two subtasks. The first one is a binary classification problem where, given a paragraph, a model has to predict wheter the paragraph contains or not PCL. The second one is a multi-label classification task where each paragraph has to be labelled with one to seven categories of PCL. Classes are not mutually exclusive and so a paragraph could express one or more categories of PCL.

To face with the first subtask we propose two deep models. The first one is a multi-channel Convolutional Neural Network (CNN). Such a network consists of parallel word embedding and convolutional layers to allow different sets of weights for trained embeddings - because of different kernel sizes employed by convolutional layers. In terms of *Precision*, *Recall* and *F1*, results of our model show certain room for improvements in future work. The

[1]https://github.com/marco-siino/McRock-SemEval-2022-Task4

second model is a hybrid bidirectional LSTM. Such a network is composed by a convolutional layer and two bidirectional LSTM layers.

For the second subtask we propose two Transormer-based models (Vaswani et al., 2017). The first one is a lighter and faster version of BERT (i.e. DistilBERT) (Sanh et al., 2019). Our model is opportunistically trained on an undersampled version of the training dataset. The model is able to outperform RoBERTa (Liu et al., 2019). The second is an XLNet-based one (Yang et al., 2019). The model is based on a generalized autoregressive pretraining method. It enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order. Under comparable experiment setting, XLNet outperforms BERT (Devlin et al., 2019) on several tasks, often by a large margin, including question answering, natural language inference, sentiment analysis, and document ranking. Our model implementation is opportunistically trained on an undersampled version of the training dataset. The model is able to outperform RoBERTa (Liu et al., 2019) in terms of average F1.

The rest of the paper is made as follows. In Section 2 we provide some background on the Task 4 hosted at SemEval-2022. In Section 3 we provide a description of the models presented. In Section 4 we provide details about the experimental setup to replicate our work. In Section 5 the results on the official task and some discussion are provided. In section 6 we present our conclusion and proposals for future works.

2 Background

In this section we provide some background about the Task 4 hosted at SemEval-2022. The aim of this task is to identify PCL, and to categorize the linguistic techniques used to express it, specifically when referring to communities identified as being vulnerable to unfair treatment by the media. Participants at the Task 4 received a dataset with sentences in context (paragraphs), extracted from news articles. Although news articles were collected from different countries, they were all provided in English. The task consists of the two subtasks listed below.

1. Subtask 1: Binary classification. Given a paragraph, a system must predict whether or not it contains any form of PCL. Two opposite labelled samples from the dataset provided are shown below.

   Non-PCL Sample Text: *"Council customers only signs would be displayed . Two of the spaces would be reserved for disabled persons and there would be five P30 spaces and eight P60 ones ."*

   Non-PCL Sample Label: [0]

   PCL Sample Text: *"It can not be right to allow homes to sit empty while many struggle to find somewhere to live, others having to sleep rough on pavements during Christmas, hoping against hope, for some charity to provide shelter. The number left homeless and destitute is alarming not necessarily at Christmas?"*

   PCL Sample Label: [1]

2. Subtask 2: Multi-label classification. Given a paragraph, a system must identify which PCL categories express the condescension. The PCL taxonomy has been defined based on previous works on PCL. The proposed categories are:

   - Unbalanced power relations
   - Shallow solution
   - Presupposition
   - Authority voice
   - Metaphor
   - Compassion
   - The poorer, the merrier

Two samples from the dataset provided are shown below. For each sample the label is an array containing seven elements. For each element, symbol *1* means that the corresponding PCL category is expressed in the paragraph.

Sample Text 1: *"Yes ... because there is NO HOPE where he lives . India is a third-world country . Do n't be fooled by call centers in big cities . Most of the country is rural and most of the population is illiterate and hopeless ."*

Sample Label 1: [1, 0, 1, 0, 0, 1, 0]

Sample Text 2: *"For refugees begging for new life , Christmas sentiment is a luxury most of them could n't afford to expect under shadow of long-running conflicts ."*

Sample Label 2: [0, 0, 1, 0, 0, 1, 0]

Task organizers released a training and a dev set before the competition officially started. For both sets the gold labels were provided. During first phase - Practice phase - participants were able to develope and test their models uploading predictions on CodaLab. After releasing the unlabelled test set the second phase - Evaluation phase - started. Results for both phases are available online [2].

3 System Overview

In this section we discuss the models presented for each subtask and the design choices made by our team motivating them. For both models the code is publicly available and reusable. Further details are provided in Section 4.

3.1 Subtask 1: Binary Classification

Given the binary nature of the task and his subject, for our first submission we developed a more versatile CNN based on the one presented in (Siino et al., 2021). Such a network is composed of parallel word embedding and convolutional layers to allow different weights for embeddings and convolutional filters. A general overview of the model architecture is shown in Figure 1. The rationale of the model presented is to have more parallel convolutional-based channel, each with different word embeddings and kernel filter weights. More properly, we set kernel size of 1, 2, 16 and 32 for each of the 32 *Conv1D* layer filters. In this way we drive our model to focus more on single token, pair of tokens, group of 16 and of 32 tokens respectively. On the basis of our experiments these are the best-performing kernel sizes for the proposed task on our preliminary 10 cross-fold validation. In addition to this behaviour we expect different coordinates for each word/token in each word embedding channel, with the aim of getting a more fine-grained positioning of words/tokens in the embedding space.

Based on our preliminary experiments, we found that on five different seeds initialization, the best word embedding size for our model is 50. This size is consistent with the common values reported in literature (Melamud et al., 2016). For each dense layer we did not use any activation function. We trained our model with a binary cross-entropy loss and using the Adam optimization algorithm (Kingma and Ba, 2014).

[2] https://sites.google.com/view/pcl-detection-semeval2022/ranking

For our second submission, we developed a light Hybrid LSTM. The model consists of a convolutional layer followed by two bidirectional LSTM layers. Such a strategy is motivated by our decision to extract relevant features from the word embedding layer before the first bidirectional one. A general overview of the model architecture is shown in Figure 2. Based on our preliminary experiments on five different seeds initialization, we found that the best word embedding size for the model was 50. For each dense layer we did not use any activation function. We trained our model with a binary cross-entropy loss and using the Adam optimization algorithm (Kingma and Ba, 2014).

3.2 Subtask 2: Multi-Label Classification

For our first submission at the Subtask 2 we choosed a transformer-based model lighter than BERT (i.e. DistilBERT). Due to the high number of experiments to perform, we needed a faster model to train. DistilBERT is a smaller general-purpose language representation model. In DistilBERT the original size of BERT model is reduced by 40%, while retaining 97% of its language understanding capabilities and being 60% faster. In terms of knowledge distillation, while BERT is the teacher, DistilBERT is the student. Student is represented by a compact model and is trained to reproduce the behaviour of the larger model (i.e. the teacher). Such a compact model is trained with a linear combination of three losses: the *distillation loss* (i.e. L_{ce}), the *masked language modeling loss* (i.e. L_{mlm}), and the *cosine embedding loss* (i.e. L_{cos}). Because of the distilled nature of the model, training and fine-tuning on a specific dataset for a specific task is of prominent importance. For a detailed discussion of DistilBERT refer to (Sanh et al., 2019). While we firstly compared the results on the dev set provided, we finally trained our model on the full training set - union of train and dev set - providing predictions on the test set. In addition we found beneficial maintaining the information about casing of characters. So we did not lowercase the text provided, implementing a cased version of DistilBERT and setting as output for each label seven digits corresponding to the seven categories of PCL. Finally we preprocessed each sample to include country and keyword of each paragraph in the input text.

For the second submission we implemented an XLNet-based model. Different unsupervised pre-

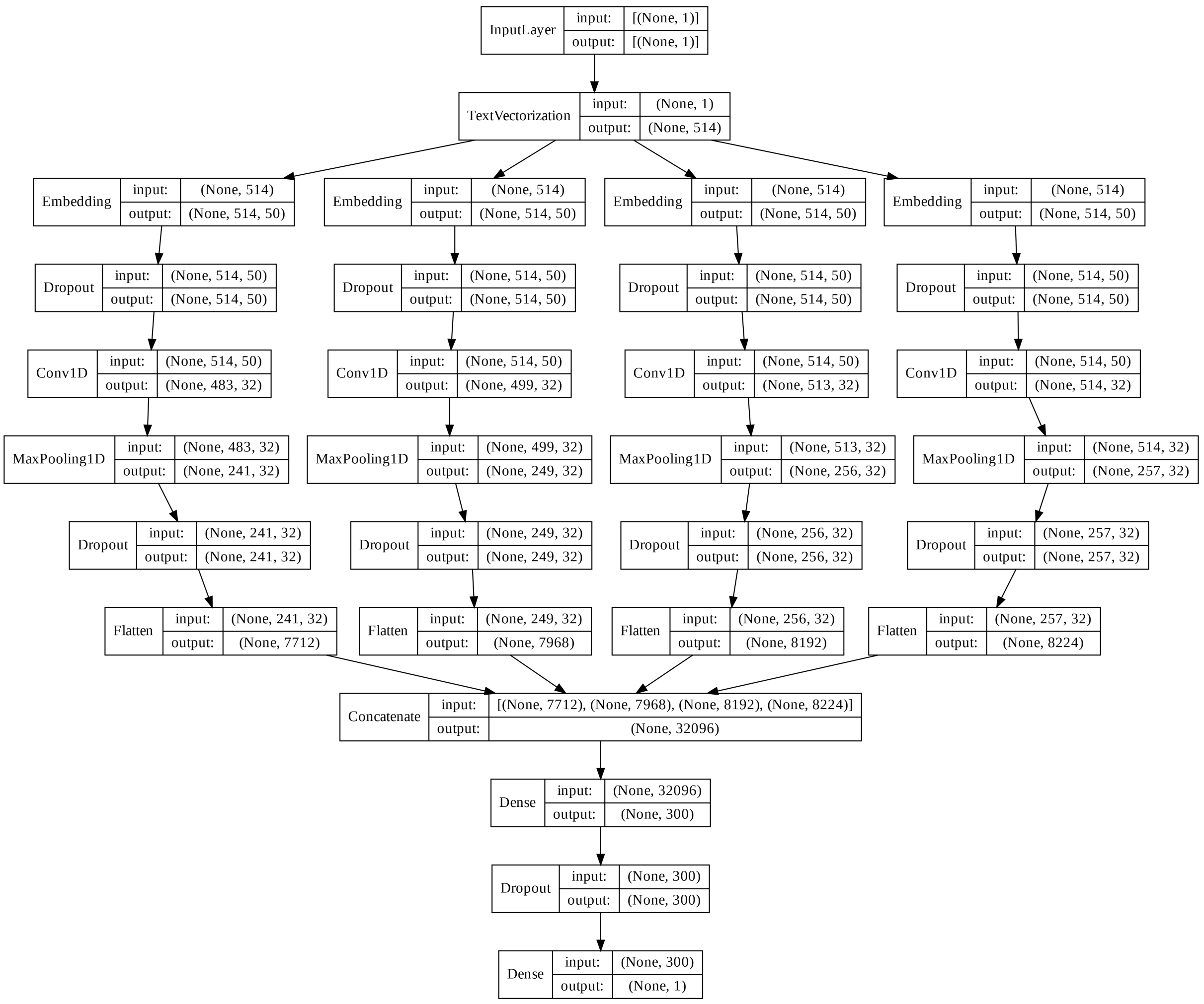


Figure 1: Overview of the multi-channel CNN presented for the first subtask at SemEval-2022. Each channel has a different kernel size at *Conv1D*, driving model attention on different sized windows of words. The kernel size of filters used at each *Conv1D* are 1, 2, 16 and 32. Each convolutional layer has 32 filters separately trained during training phase. Such a strategy allows extraction of different-sized features for a fine-grained learning.

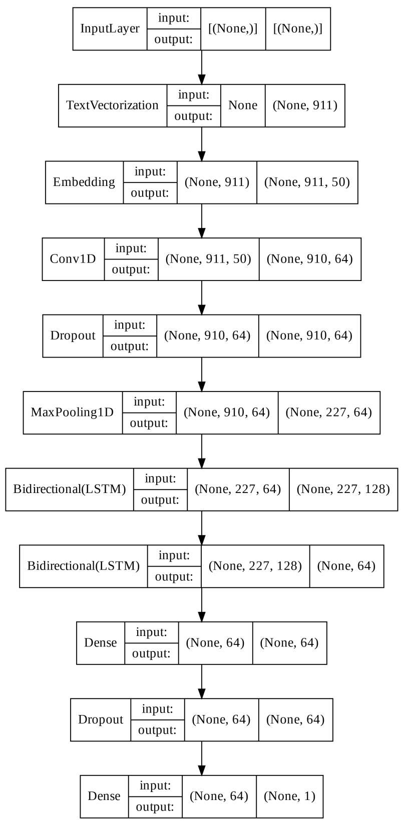

InputLayer | input: | output: | [(None,)] | [(None,)]
TextVectorization | input: | output: | None | (None, 911)
Embedding | input: | output: | (None, 911) | (None, 911, 50)
Conv1D | input: | output: | (None, 911, 50) | (None, 910, 64)
Dropout | input: | output: | (None, 910, 64) | (None, 910, 64)
MaxPooling1D | input: | output: | (None, 910, 64) | (None, 227, 64)
Bidirectional(LSTM) | input: | output: | (None, 227, 64) | (None, 227, 128)
Bidirectional(LSTM) | input: | output: | (None, 227, 128) | (None, 64)
Dense | input: | output: | (None, 64) | (None, 64)
Dropout | input: | output: | (None, 64) | (None, 64)
Dense | input: | output: | (None, 64) | (None, 1)

Figure 2: Overview of the Hybrid LSTM presented for first subtask hosted at SemEval-2022. The presence of the *Conv1D* layer is motivated by our intention to extract relevant features from the previous embedding layer. The kernel size of the 64 filters used by the convolutional layer is 2. Such a strategy should allows extraction of relevant bi-grams from the input text.

training objectives have been explored in literature. XLNet implements a generalized autoregressive pretraining method that uses a permutation language modeling objective to combine the advantages of autoregressive and autoencoding methods. The neural architecture of XLNet is developed to work seamlessly with the autoregressive objective, including the integration of Transformer-XL and the careful design of the two-stream attention mechanism. XLNet achieves substantial improvement over previous pretraining objectives on various tasks. Among them, autoregressive language modeling and autoencoding have been the two most successful pretraining objectives. Furthermore, XLNet integrates ideas from Transformer-XL (Dai et al., 2019) into pretraining. An XLNet model integrates two techniques from Transformer-XL, namely the relative positional encoding scheme and the segment recurrence mechanism. The relative positional encodings is applied based on the original sequence. Furthermore, the recurrence mechanism is included into the proposed permutation setting and enable the model to reuse hidden states from previous segments.

Training and fine-tuning of an XLNet for a specific task is of prominent importance. While we firstly compared the results on the dev set provided, we finally trained our model on the full training set - e.g., union of train and dev set - providing predictions on the test set.

4 Experimental Setup

We implemented our first two models using Keras[3] and TensorFlow[4]. The dataset provided for the binary classification task is unbalanced in terms of negative and positive PCL instances. To face with this issue we undersampled the negative instances. On the basis of our preliminary experiments, we found beneficial undersampling negative instances to be just six times more the positive ones. Furthermore we found beneficial to include in each sample (both for training and prediction) the keyword and the country field of each paragraph from the dataset. Then we used a batch size of 100. We empirically found that a good early stopping point for the training phase is obtained with 10 epochs and a learning rate of 0.001. We ran the experiments on Google Colab using the default GPU (NVIDIA Tesla K80). The training time was around 15 seconds for each

[3]https://keras.io/
[4]https://www.tensorflow.org/

of the ten epochs. The official metrics used for the task were Precision, Recall and F1 on positive instances (sample containing PCL). But during our development phase we focused on the model loss (i.e., *binary crossentropy loss*). This choose was dictated by the fact that the gold labels of the test set were not provided.

The models for Subtask 2 were implemented using Simple Transformers[5]. We used DistilBERT and XLNet as the pre-trained language models. We preprocessed the dataset to include, within the text of each sample, the country and the keyword of the paragraph. To train our final models we built a single dataset consisting of the train and the dev set. Then we undersampled negative instances (i.e. Non-PCL samples) to alleviate bias in the unbalanced dataset provided. We ran the experiments on Google Colab, using an NVIDIA Tesla K80 GPU. The official metrics used for the task were F1 for each category and average F1 among them. In this case too, during our development phase we focused only on the loss of the models to perform some fine-tuning.

5 Results

For Subtask 1 the metrics used are Precision, Recall and F1 defined as shown in 1,2,3 respectively.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3}$$

Each *True Positive* (TP) is computed on the positive instances (i.e. paragraphs containing PCL). So a TP is a sample containing PCL and correctly classified, a *False Positive* (FP) is a sample without PCL but wrongly classified as a PCL sample, a *False Negative* (FN) is a sample containing PCL but wrongly classified as not containing PCL. Therefore, Precision is the number of the correctly predicted PCL samples over the total number of predicted PCL samples. Recall is the number of the correctly predicted PCL samples over the total number of actual PCL samples. Finally the F1 Score is the harmonic mean of Precision and Recall.

The final ranking for the first subtask is drawn up accordingly to the F1 score on the test set provided.

[5]https://github.com/ThilinaRajapakse/simpletransformers

	F1	P	R
RoBERTa-baseline	48.29	34.99	77.89
Multi-Channel CNN	32.29	23.46	51.76
Hybrid LSTM	26.32	31.47	22.61
Random-baseline	17.35	10.40	52.26

Table 1: Performance comparison on dev set. The results of the two baseline methods provided by the organizers (i.e. RoBERTa and Random baseline) compared to our models based on a multi-channel CNN and Hybrid LSTM.

In Table 1 are shown the results on the dev set provided by the organizers. Results are ordered according to F1 score. Our model based on multi-channel CNN is able to outperform the Random-baseline provided in terms of F1 and Precision, obtaining similar results in terms of Recall. RoBERTa-baseline performs better along the three metrics provided.

It is interesting to note that RoBERTa is a model pre-trained on over 160GB of text. Compared to our proposed model it requires much more in terms of resources and time needed. Despite such efforts, RoBERTa outperforms our model by only 16% and around 11% in terms of F1 and Precision. The most significant difference is with Recall. This means that the proportion of actual positives identified correctly by our model is lower compared to RoBERTa. This could be mainly due to the inability of our model at contrasting the bias learned because of the unbalanced dataset provided, where Non-PCL paragraphs are, in fact, the vast majority. Our team did an additional submission involving two deep models based on an Hybrid LSTM (i.e. made of convolutional and bidirectional LSTM layers) and on an XLNet (Yang et al., 2019). Our proposed Hybrid LSTM is able to outperform the Random-baseline provided in terms of F1 and precision. RoBERTa-baseline performs better along the three metrics provided. Compared to the Hybrid LSTM model, the multi-channel CNN outperforms the Hybrid LSTM. However the Hybrid LSTM performs better with regard to precision. Such a result leads to the conclusion that Hybrid LSTM correctly predicts an higher number of actual PCL paragraphs with respect to the total predicted PCL paragraps. Therefore, further investigation might be conducted on combinations of main components of the two proposed models in the effort to improve the F1.

	F1	P	R
hudou (1)	65.10	64.60	65.62
RoBERTa (44)	49.11	39.35	65.30
Multi-CNN (69)	29.28	23.40	39.12
Hybrid LSTM (*NA*)	28.15	29.62	26.81
mahangchao (79)	4.48	10.59	2.84
makahleh (80)	0.0	0.0	0.0

Table 2: Performance comparison on test set. In table are shown the RoBERTa-baseline, the first classified (i.e. *hudou*), the two last classified and our models results. In parentheses are shown the positions in the final ranking according to F1 score. *NA* stands for *Not Assigned* because only the best result of the two model submitted is considered for final ranking.

In Table 2 are shown the results on the test set provided by the organizers without the gold labels. Results are ordered based on the F1 score. Compared to the winner (e.g. *hudou*), RoBERTa exhibits the most significant gap in Precision. Which means that proportion of positive instances correctly classified by the winner team is significantly more compared to RoBERTa. However, in this case too, RoBERTa outperforms our model with similar gap along the three metrics with respect to the results presented for the dev set. Our two submitted models exhibit similar performances on the test set. In this case too, the most significant gap is in Recall.

For Subtask 2 the metric used is F1 along the seven categories provided and the final ranking was drawn up considering the average F1 along the seven categories on the test set provided. For this subtask there is an important bias due to the unbalanced nature of the dataset with regard to each category. In Table 3*(a)* the results on the dev set are shown. Results are ordered based on the average F1 score. For each category our XLNet is able to outperform the Random-baseline. The average F1 is 15% more than such a baseline. It is worth noting that results with a random predictor are not uniformly distributed along each category. This distribution provides further evidences about the unbalanced nature of the dataset with regard to this multi-label classification subtask. Furthermore the random predictor outperforms F1 score of RoBERTa in four of the seven categories provided. However RoBERTa performs a lot better in detecting *Unb*, *Pre* and *Com* language (namely, *Unbalanced power relations*, *Presupposition* and *Compassion*). These performances could be motivated by the greater number of samples in the dataset expressing the first category. Compared to RoBERTa our DistilBERT-model does better for five categories out of seven. And for this single category (i.e. *Presupposition*) the gap is under 4%. Compared to our other submission, the XLNet heavily outperforms DistilBERT in terms of F1 for each category and in the final average F1. In Table 3*(b)* we report the results of the first model, our proposed models, RoBERTa and the last classified one, according to the final ranking drawn up considering the average F1. In this case too our models outperform RoBERTa, in terms of F1, for six out of seven categories. On the test set, RoBERTa performs better in detecting *Unb*. However, compared to the results on dev set, our two proposed models perform with a lower average F1 gap. And there is just a category (i.e. *Metaphor*) where DistilBERT significantly outperforms the XLNet. It is worth noting that the best performing model is able to reach an average F1 of 46.89, outperforming of over 20% and 36% our proposed models and RoBERTa respectively. This lead to a conclusion about the very large room for improvement in this multi-label task. Some of the difficulties in reaching an average F1 of at least 50% could be due to the unbalanced dataset as much as the intrinsic complexity of the task.

6 Conclusion

We propose four deep learning models to detect and classify PCL on the English dataset provided by task organizers at SemEval-2022. For the first subtask we developed a Multi-Channel CNN, training parallel word emebeddings and convolutional layers with different kernel sizes and an Hybrid LSTM. While results of these architectures exhibit a large room for improvements, the models are lighter and faster compared to the RoBERTa-baseline model proposed by the task organizers. For the second subtask we implemented a DistilBERT-based and XLNet-based models. Compared to RoBERTa, DistilBERT is smaller, faster and lighter. Instead, XLNet performs better on average F1 both on dev and test set. To face with the task proposed we opportunistically trained the models including the information about country and keyword related to each

	Unb	Sha	Pre	Aut	Met	Com	The	**AVG**
XLNet	47.99	20.41	24.61	20.06	16.67	39.24	8.89	25.41
DistilBERT	47.60	15.90	23.84	15.53	10.91	31.23	0.0	20.72
RoBERTa-baseline	35.35	0.0	29.63	0.0	0.0	28.78	0.0	13.40
Random-baseline	11.30	3.23	5.09	3.22	6.04	8.21	1.31	5.48

(a)

	Unb	Sha	Pre	Aut	Met	Com	The	**AVG**
guonihe (1)	65.60	52.94	36.90	40.66	35.90	49.18	47.06	46.89
XLNet (29)	32.32	32.93	19.18	20.55	22.22	26.35	7.14	22.96
DistilBERT (*NA*)	32.62	30.49	18.80	18.31	26.00	25.37	0.0	21.65
RoBERTa-baseline (37)	35.35	0.0	16.67	0.0	0.0	20.87	0.0	10.41
nikss (49)	0.0	1.01	0.0	0.0	0.0	0.0	1.09	0.03

(b)

Table 3: Performance comparison on dev set *(a)* and test set *(b)* for Subtask 2. The table shows F1 calculated for each category and the average F1 in the last column. For Subtask 2 our proposed models based on DistilBERT and XLNet outperform RoBERTa on both dev and test set. In parentheses are shown positions in final ranking. NA stands for *Not Assigned* in this case too.

sample. In addition we undersampled the negative instances in the dataset to avoid the model to focus more on non-PCL samples. The trained models are able to outperform RoBERTa. However, looking at the final ranking of the task, the room for improvements is significant. In future works would be useful implementing models taking advantage of a balanced dataset. Both for the binary classification task and for the multi-label one. Another interesting aspect to further investigate would be about the behaviours of the proposed models on multilingual datasets. Although pre-trained models are actually the state of the art for many NLP tasks, the hardness of the PCL detection task - proved by the final scores obtained by the winners at SemEval-2022 - could worsen the results on each metric. Finally, it could be beneficial experimenting with hybrid and ad-hoc models combining different pre-trained and non pre-trained models to improve the results specifically on this task.

Acknowledgments

We would like to thank anonymous reviewers for their comments and suggestions that have helped to improve the presentation of the paper.

CRediT Authorship Contribution Statement

Marco Siino: Conceptualization, Formal analysis, Investigation, Methodology, Software, Supervision, Validation, Visualization, Writing - Original draft, Writing - review & editing. **Marco La Cascia:** Supervision, Writing - review & editing. **Ilenia Tinnirello:** Supervision, Writing - review & editing.

References

Ibrahim Arpaci, Mostafa Al-Emran, Mohammed A Al-Sharafi, and Khaled Shaalan. 2021. A novel approach for predicting the adoption of smartwatches using machine learning algorithms. In *Recent advances in intelligent systems and smart applications*, pages 185–195. Springer.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, volume 2263, pages 1–9. CEUR.

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

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

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018. Overview of the task on automatic misogyny identification at ibereval 2018. *Ibereval@ sepln*, 2150:214–228.

Ali Hosseinalipour and Reza Ghanbarzadeh. 2022. A novel approach for spam detection using horse herd optimization algorithm. *Neural Computing and Applications*, pages 1–15.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ritesh Kumar, Guggilla Bhanodai, Rajendra Pamula, and Maheshwar Reddy Chennuru. 2018. Trac-1 shared task on aggression identification: Iit (ism)@ coling'18. In *Proceedings of the first workshop on trolling, aggression and cyberbullying (TRAC-2018)*, pages 58–65.

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

Oren Melamud, David McClosky, Siddharth Patwardhan, and Mohit Bansal. 2016. The role of context types and dimensionality in learning word embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1030–1040.

Carla Pérez-Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Francisco Rangel, Anastasia Giachanou, Bilal Hisham Hasan Ghanem, and Paolo Rosso. 2020. Overview of the 8th author profiling task at pan 2020: Profiling fake news spreaders on twitter. In *CEUR Workshop Proceedings*, volume 2696, pages 1–18. Sun SITE Central Europe.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Marco Siino, Elisa Di Nuovo, Tinnirello Ilenia, and Marco La Cascia. 2021. Detection of hate speech spreaders using convolutional neural networks. In *PAN 2021 Profiling Hate Speech Spreaders on Twitter@ CLEF*, volume 2936, pages 2126–2136. CEUR.

Marco Siino, Marco La Cascia, and Ilenia Tinnirello. 2020. Whosnext: Recommending twitter users to follow using a spreading activation network based approach. In *2020 International Conference on Data Mining Workshops (ICDMW)*, pages 62–70. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Team Stanford ACMLab at SemEval-2022 Task 4: Textual Analysis of PCL Using Contextual Word Embeddings

Upamanyu Dass-Vattam, Spencer Wallace, Rohan Sikand, Zach Witzel, Jillian Tang
Stanford University
{udvattam, spenwall, rsikand, zachwitz, jiltang}@stanford.edu

Abstract

We propose the use of a contextual embedding based-neural model on strictly textual inputs to detect the presence of patronizing or condescending language (PCL). We finetuned a pre-trained BERT model to detect whether or not a paragraph contained PCL (Subtask 1), and furthermore finetuned another pre-trained BERT model to identify the linguistic techniques used to convey the PCL (Subtask 2). Results show that this approach is viable for binary classification of PCL, but breaks when attempting to identify the PCL techniques. Our system placed 32/79 for subtask 1, and 40/49 for subtask 2.

1 Introduction

The goal of the task is to be able to identify whether or not a piece of text contains condescending or patronizing language, and if it contains patronizing language identity which linguistic techniques are used to express that sentiment (P'erez-Almendros et al., 2022). Studies have shown that PCL fuels discriminatory behavior, creates and feeds stereotypes, subtly blames needy individuals for their problems, and oversimplifies problems. In general, PCL makes it harder for needy communities to get the help they need and reach total inclusivity (P'erez-Almendros et al., 2020). This is obviously negative, and being able to combat it with AI may help automate the process and get past inherent biases that humans identifying PCL may have.

Our system's goal is to use contextualized word embeddings to feed the model, and thus have the model analyze the semantics of the text. We specifically focused on a purely textual analysis and did not provide the model with any metadata to see if the model could learn the PCL patterns just from the text, because although real world usage would likely include those features, the ability to learn from the text itself would be useful and more universally applicable.

In this task, we learnt that it is easier for a model to simply detect the presence of PCL than the techniques used in a piece of PCL. This is likely because of both unequal distributions of data and the fact that the context for the types of PCL likely look very similar, making the model default to a category of PCL with a higher frequency when not incredibly clear. We ranked 32/70 on subtask 1, which is detecting the presence of PCL, and ranked 40/49 on subtask 2, which involves identifying the PCL techniques used. In particular, the model struggled with the 'authority voice' and 'metaphor' categories.

2 Task Overview

The dataset provided was the Don't Patronize Me! Dataset (P'erez-Almendros et al., 2020), which is a collection of paragraphs mentioning vulnerable communities and published in media in 20 English speaking countries. Each paragraph has the country code where the paragraph was published and the keyword that was used to search for it. For subtask 1, the paragraphs are manually annotated with a label from 0-4 on how much PCL it contains; these are converted to binary labels on whether or not a paragraph contains PCL, where 2-4 indicate positive examples and 0-1 indicate negative examples. For subtask 2, all paragraphs in the dataset contain PCL and are annotated with spans containing categories of linguistic techniques that are used to express the condescension. These categories are:

- Unbalanced power relations
- Shallow solution
- Presupposition
- Authority voice
- Metaphor
- Compassion
- The poorer, the merrier

The training portion of the dataset for subtask 1 contained 10,636 paragraphs, and the corresponding dataset for subtask 2 had 993 paragraphs with 2,792 instances of PCL techniques.

3 System Overview

3.1 Data Representation

We felt that attempting to learn from the text itself and removing the contextual metadata could lead to more robust textual analysis. Therefore, we relied solely on the paragraph as our input feature. We tried two approaches: 1) using pre-trained GloVe embeddings with a dimension of 300, which track co-occurences of words in a global corpus (Pennington et al., 2014), and 2) tokenizing the paragraphs using a BERT tokenizer and inputting these into a pre-trained BERT model (Devlin et al., 2019). For subtask 2, where the focus was on shorter labeled spans rather than entire paragraphs, we still used the full text to provide more context for the detection of PCL.

3.2 Subtask 1

GloVe embeddings are good at capturing word analogies due to its global vectorization and its ability to capture sub-linear relationships in the vector space (Pennington et al., 2014). We felt that the analogy ability was specifically important to this task, because it could help capture tonal similarities between instances of PCL. Therefore, we began with a bag-of-words model where we summed the GloVe embeddings of each word in the text, then performed logistic regression to output a binary label indicating whether or not the text contained PCL. However, due to the high class imbalance present in the data, this model predicted no PCL for nearly all inputs.

Our final model was a fine-tuned BERT model. Initially, we re-labeled the training data with the final binary labels of PCL and not PCL, but this led to issues due to the high class imbalance. We decided to make the model a multiclass classifier which output the original 0-4 labels, and then convert these to binary labels. This allowed us to better adjust model weights to reflect the imbalances in class distributions, because the imbalances were not standard to all PCL, and varied dramatically based on the subtype.

3.3 Subtask 2

The most notable decision in this subtask was using the entire paragraph instead of focusing on the spans of PCL. We initially actually tried training on just the labeled spans, but these did not provide enough context for the BERT model to fine-tune to the data. Therefore, we used the entire text as the input to provide more context. Due to the small amount of data for the second subtask, we also chose to apply transfer learning and start our training from the fine-tuned model from the first subtask.

One major problem with the task is that the contexts between the types of PCL are all similar, as there can be many instances of categories within the same paragraph of text in smaller spans. This leads to the model defaulting to predicting the more frequent classes. We tried to address this by adding in the spans without context as training data; however this actually decreased performance.

4 Experimental Setup

We used BertTokenizerFast to tokenize the text and fine-tuned on the pre-trained BertForSequenceClassification model, both from the HuggingFace Transformers library (Wolf et al., 2020). We conducted hyperparameter optimization using the HyperOpt package (Bergstra et al., 2013), using population-based training (Jaderberg et al., 2017). It automatically generated sets of hyperparameters for us, and then based on the results of training with those hyper parameters updated the future hyperparameter sets. Our final model was trained using Adam optimization with a learning rate of 2.31468e-05 (Kingma and Ba, 2014); we trained the model for 6 epochs with a batch size of 8. We used a train/test split of 70/30 and evaluated based on accuracy and F1.

5 Results

Our model had precision 0.4017, recall 0.7666, and F1 0.5271 on the evaluation data for subtask 1, and has an average F1 of 0.0963 for subtask 2. We placed 32/79 for subtask 1, and 40/49 for subtask 2. Based on the results, the model had a hard time with the types of PCL that showed up less frequently in the data, and tended to perform best on the categories that were more frequent. In retrospect, using the same model architecture and setup on the two subtasks was not the optimal way

Metric	Score
Precision	0.4017
Recall	0.7666
F1	0.5271

Table 1: Results for Subtask 1.

PCL Category	Score
Unbalanced Power Relations	0.1596
Shallow Solutions	0.2694
Presupposition	0.0423
Authority Voice	0.0
Metaphor	0.0
Compassion	0.0864
The Poorer, The Merrier	0.1212
F1 Average	0.0963

Table 2: Results for Subtask 2.

to approach the task, despite some compelling reasons to approach it that way. We did not perform any quantitative analysis or ablations, but given the chance we would augment the less frequent PCL categories and see if that would fix the prediction issues for subtask 2, even if it wouldn't necessarily improve accuracy.

6 Conclusion

We developed a system which attempted to first classify whether or not a piece of text contained patronizing or condescending language, and then identify the technique used to convey the PCL. In particular, we focused on examining whether or not pure textual analysis using contextual word embeddings alone would be enough to perform the aforementioned tasks. Based on our results, this approach is only viable for the binary classification of whether or not a text contains PCL.

In the future, we would explore creating an ensemble of models, only one of which uses textual analysis, and the rest would focus on things like meta data and word frequencies which do not rely on context. Comparing the result of that ensemble to a traditional approach to this problem which uses many of those methods simultaneously would show whether or not there is a strong overlap between sources, locations, etc. and PCL, or whether only the text itself is the best indicator of a sign of PCL.

7 Acknowledgements

Thanks to the Stanford ACMLab for facilitating this entry and their instruction, particularly Ethan Chi and Patrick Liu for their reviews.

References

James Bergstra, Daniel Yamins, and David Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning*, pages 115–123. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Journal of the Association for Computing Machinery*, arXiv:1810.04805.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M. Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, Chrisantha Fernando, and Koray Kavukcuoglu. 2017. Population based training of neural networks. arXiv:1711.09846.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

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

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

Team LRL_NC at SemEval-2022 Task 4: Binary and Multi-label Classification of PCL using Fine-tuned Transformer-based Models

Kushagri Tandon, Niladri Chatterjee
Indian Institute of Technology Delhi
Hauz Khas, Delhi-110016, India
{kushagri.tandon,niladri}@maths.iitd.ac.in

Abstract

Patronizing and condescending language (PCL) can find its way into many mediums of public discourse. Presence of PCL in text can produce negative effects in the society. The challenge presented by the task emerges from the subtleties of PCL and various data dependent constraints. Hence, developing techniques to detect PCL in text, before it is propagated is vital. The aim of this paper is twofold, a) to present systems that can be used to classify a text as containing PCL or not, and b) to present systems that assign the different categories of PCL present in text. The proposed systems are primarily rooted in transformer-based pretrained language models. Among the models submitted for Subtask 1, the best F1-Score of 0.5436 was achieved by a deep learning based ensemble model. This system secured the rank 29 in the official task ranking. For Subtask 2, the best macro-average F1-Score of 0.339 was achieved by an ensemble model combining transformer-based neural architecture with gradient boosting label-balanced classifiers. This system secured the rank 21 in the official task ranking. Among subsequently carried out experiments a variation in architecture of a system for Subtask 2 achieved a macro-average F1-Score of 0.3527.

1 Introduction

The aim of the current task, viz. Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022), is to identify presence of condescending and patronizing tones in text, particularly by the media when referring to vulnerable communities. Preponderance of PCL in news and different social media texts, often targeted towards marginalised and under represented communities is a major social concern these days. It can feed stereotypes, tilt the scales of superiority towards a particular community, and fuel discriminatory behaviour. Thus the task of identifying PCL particularly that directed to vulnerable communities is a crucial task.

The task is based on the English language Don't Patronize Me! dataset (Pérez-Almendros et al., 2020). It consists of two subtasks, binary classification (Subtask 1) and multi-label classification (Subtask 2). In Subtask 1, given a paragraph the aim is to predict whether the paragraph consists of any form of patronizing and condescending language (PCL). In Subtask 2, the aim is to assign each paragraph a subset of labels which express different categories of PCL. The major challenges offered in such tasks emerge from various linguistic aspects, such as use of cryptic sentences, sarcasms used, polysemous nature of the English words among others. For the present task, further challenge emerges from the imbalance of dataset as discussed in Section 2.

Experiments were conducted with various systems and the best performing ones for both the subtasks are discussed in detail. The two systems that perform best among these, for both the subtasks are ensemble techniques, which employ two or more classifier models at different stages of the system.

The paper is organized as follows. The task background is discussed in Section 2. Sections 3 and 4 discuss the details of the systems and the experimental setup, respectively. The results from the systems are given in Section 5. The paper is concluded in Section 6.

The code for the proposed systems have been made available at https://github.com/KushagriT/SemEval2022-TeamLRL_NC

2 Background

As mentioned in Section 1, this task is based on the Don't Patronize Me! dataset. The paragraphs for this dataset have been extracted form the News on Web (NoW) corpus (https://www.english-corpora.org/now/), and have been manually anno-

Label:	1	2	3	4	5	6	7
Neg/Pos	14	52	46	45	52	21	261

Table 1: Label Ratios

tated. The statistics of the subset of data used for this task are as follows. The Train subset has 8375 samples, Dev subset has 2094 samples, and the Test set has 3832 samples. The label set considered for this multi-label classification consists of seven labels, namely 'Unbalanced power relations'(1), 'Shallow solution'(2), 'Presupposition'(3), 'Authority voice'(4), 'Metaphor'(5), 'Compassion'(6), and 'The poorer, the merrier'(7). These labels are henceforth addressed by their sequence ids as mentioned above.

The number of negative examples per positive example, for the binary classification task, in the Train + Dev dataset (henceforth referred to as Training dataset), is approximately 10. For Subtask 2, the number of negative examples per positive example (approximated to nearest integer) for each label in the Training dataset is given in Table 1 (denoted as Neg/Pos). An approximation of these ratios are used as scaling factors to manage the unbalanced classes when creating gradient boosting classifiers for multi-label classification.

Work has been carried out in the field of NLP to identify different types of linguistic variations or harmful languages from text such as, sarcasm detection (Chatterjee et al., 2020), hate speech detection (Djuric et al. (2015), Gitari et al. (2015)), and fake news detection (Shu et al. (2017), Conroy et al. (2015)). Since the introduction of BERT (Devlin et al., 2019), transformer-based language models have been used for a variety of NLP tasks, such as the use of BERT for a regression task of predicting eye-tracking features for a given word of the sentence (Choudhary et al., 2021), or use of BERT for the task of document classification (**?**, Adhikari et al. (2019)).

All the methods proposed in this paper use further pre-trained transformer-based language models to extract document embeddings. The motivation comes from some recent work that has been carried out in detecting condescending language from text. Wang and Potts (2019) introduce a new labeled dataset, namely TalkDown, of condescending acts in context, and establish baselines for this dataset using BERT.

3 System Overview

Experiments were conducted with two types of models for each of the two subtasks. Each system uses transformer-based language models for generating text embeddings. These language models were further pre-trained on Masked Language Modeling (MLM) task on the given data. The paragraphs from the training dataset were prepared for MLM with the model specific tokenizer by masking tokens in the input with probability 0.15, and using truncation and padding to maximum length of 256.

The present experiments use further pre-trained RoBERTa (Liu et al., 2019) and XLNet (Yang et al., 2019) language models[1]. These further pre-trained RoBERTa (roberta-base) and XLNet (xlnet-base-cased) models are henceforth denoted as MyRoberta and MyXlnet, respectively. This further pre-training is done using the given data on the existing pre-trained RoBERTa and XLNet models in order to fine-tune the models for the two specific subtasks, so they become adapted to the given corpus irrespective of the task at hand, whether it is binary classification or multi-label classification.

The systems discussed in Sections 3.1 and 3.2, use a custom attention head architecture[2]. The input to this attention head is of dimension $batch\ size \times sequence\ length \times embedding\ dimension$, and the output is of dimension $batch\ size \times embedding\ dimension$. The attention mechanism is used so that the system learns the emphasis of different tokens towards the classification. This layer is henceforth referred to as Attn_head.

Embeddings are generated using these models to extract document representations, which will be used as features in each system. For training the proposed models, discussed in sections 3.1 and 3.2, and for the training of the MyRoberta and MyXlnet models, smart batching is used, with a maximum token length limit of 256. In smart batching the dataset is sorted by length of the sequences before creating batches, and padding is done to each sequence in each batch to the length of the longest sequence in that batch. In the following subsections the model algorithms are discussed in detail. Smart batching has been used to optimize the training speed for the transformer-based models, since now most of the resulting batches will have shorter

[1]Training details are discussed in the appendix
[2]Architecture details discussed in the appendix

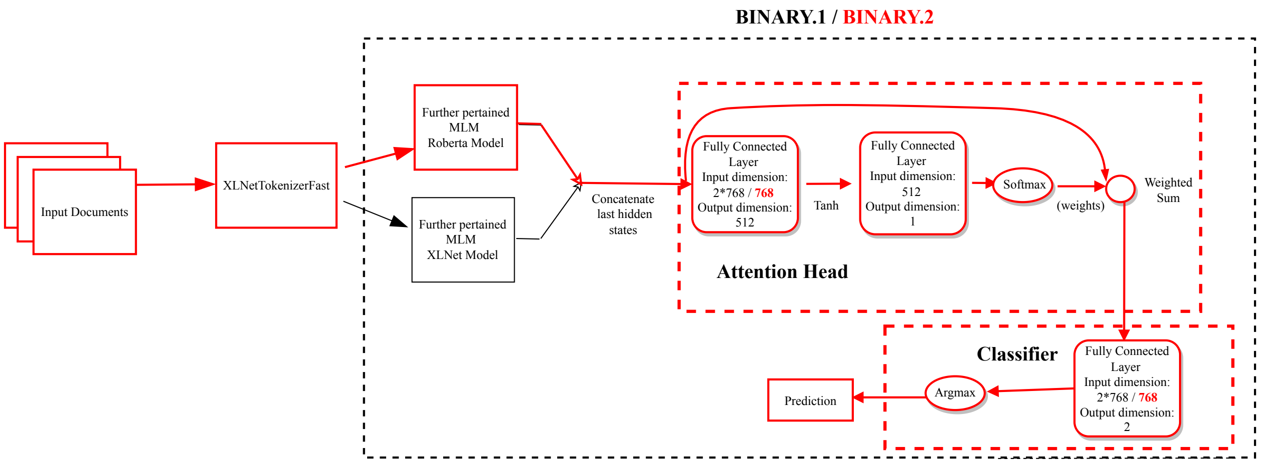

BINARY.1 / BINARY.2
Input Documents
XLNetTokenizerFast
Further pertained MLM Roberta Model
Further pertained MLM XLNet Model
Concatenate last hidden states
Fully Connected Layer Input dimension: 2*768 / 768 Output dimension: 512
Tanh
Fully Connected Layer Input dimension: 512 Output dimension: 1
Softmax
(weights)
Weighted Sum
Attention Head
Classifier
Fully Connected Layer Input dimension: 2*768 / 768 Output dimension: 2
Argmax
Prediction

Figure 1: BINARY.1 and BINARY.2 (flow indicated in red)

sequence lengths which would prevent memory overload.

The following notations have been used in the further sections. The two subtasks of Binary and Multi-label classification have been referred to as, Subtask 1 and Subtask 2, respectively. The two systems in for each of these tasks have been denoted as BINARY.1, and BINARY.2 for Subtask 1, and MULTI.1 and MULTI.2 for Subtask 2. MULTI.1 and MULTI.2, each have two sub-models which are denoted as MULTI.1.A and MULTI.1.B for MULTI.1, and MULTI.2.A and MULTI.2.B for MULTI.2.

3.1 Subtask 1: Binary Classification

In this section the two proposed systems for the task of binary classification are discussed.

3.1.1 BINARY.1: MyRoberta + MyXlnet + Attn_head

The first component of BINARY.1 consists of MyRoberta and MyXlnet. Concatenation of the last hidden states from the two models is passed as an input to Attn_head. The output from this layer is sent to a fully connected layer which serves as the classifier unit. The architecture for BINARY.1 is given in Figure 1. The entire flow indicates the algorithm for BINARY.1.

This system is trained for the task of binary classification with Cross Entropy loss using AdamW optimizer which is Adam (Kingma and Ba, 2014) optimizer with weight decay (Loshchilov and Hutter, 2017). A linear learning rate scheduler with 50 warm-up steps is used. The hyperparameters for this system are given in Table 2.

Hyperparameter	Value
Train Epochs	5
Batch Size	16
Initial Learning Rate	2e-5

Table 2: Hyperparameters for BINARY.1

By using ensemble technique the quality of representation of these documents is improved, and using attention this model learns the emphasis of each token in the text sequence in contributing towards each label in the predicted label set. This can be observed from the ablation experiments discussed in Section 5.

3.1.2 BINARY.2: MyRoberta + Attn_head

The basic architecture used in BINARY.2 is similar to that of BINARY.1 described in Section 3.1.1. This system, however, trains the following model.

The first component of this system is MyRoberta. The last hidden states from this model is passed as input to Attn_head. The output from this layer is sent to a fully connected layer which serves as the classifier unit. The architecture for BINARY.2 is given in Figure 1, with its flow indicated in red. In case of BINARY.2, the input dimension for the first fully connected layer in the Attention Head unit and the classifier unit is 768 instead of 2*768, as for BINARY.1.

In this system, while training the weights in the pooler layer and the last 5 layers of MyRoberta are re-initialized. The weights for the fully connected layers in MyRoberta are re-initialized with mean 0 and standard deviation same as that of the initializer range of MyRoberta[3]. The weights and biases of the layer normalization in MyRoberta are re-initialized to constant values of 1 and 0, respectively.

This system is trained for the task of binary classification with Cross Entropy loss using AdamW optimizer. A linear learning rate scheduler with 50 warm-up steps is used. For BINARY.1 a smaller batch size is used to prevent memory overload.

Hyperparameter	Value
Train Epochs	5
Batch Size	32
Initial Learning Rate (Group 1)	1e-5
Initial Learning Rate (Group 2)	2e-5
Initial Learning Rate (Group 3)	4e-5
Initial Learning Rate (Group 4)	5e-5

Table 3: Hyperparameters for BINARY.2

The initial layers of the MyRoberta model encode the more general information that is present in the text. Additionally, since MyRoberta has already been trained on the task-specific data, the embedding layer and the first four layers of MyRoberta are given a lower initial learning rate of 1e-5. This parameter group is denoted as Group 1. As the layers move closer the output or the classifier layer, the model encodes task-specific information. Hence for the next four layers (Group 2), the learning rate is chosen as 2e-5, and for the last four layers (Group 3) the learning rate is chosen to be 4e-5. The classifier and the pooler layers are assigned a higher learning rate of 5e-5. This parameter group is denoted as Group 4. Each of these layers have weight decay of 0.01, except for bias and layer normalization weights. Table 3 shows the values of the different hyperparameters for this system.

3.2 Subtask 2: Multi-Label Classification

In this section the two proposed systems for the Subtask 2, namely MULTI.1 and MULTI.2 are discussed. MULTI.1 consists of two sub-models, MULTI.1.A and MULTI.1.B. MULTI.1.A is MyRoberta + FCL_1 where FCL_1 denotes a fully connected layer, and MULTI.1.B is a collection of label-balanced XGBoost Classifiers (XGB). MULTI.2 also consists of two sub-models. In MULTI.2.A, a similar architecture is used as in MULTI.1.A, but with different dimensions and with an additional layer named FCL_2 which a fully connected classifier unit. MULTI.2.B has two parts: i) Fuzzy C-Means clustering to extract features, ii) a multi-label classifier model based on fuzzy membership. These two parts will be denoted as FCM and Fuzzy_CLF, respectively. The flow of output between the two models are explained in Section 3.2.2.

3.2.1 MULTI.1: MyRoberta + FCL_1 + XGB

This system has two parts, namely a model to extract features from the text (MULTI.1.A), and a model to assign a set of labels to this text using a classifier (MULTI.1.B). The architecture for this model is given in Figure 2.

To extract features MyRoberta is fine-tuned for the task of multi-label text classification. The first component of MULTI.1.A is MyRoberta. The CLS embedding from MyRoberta is passed to a fully connected classifier layer with output dimension 7, corresponding to the seven labels. In this model, while training the weights in the pooler layer and the last 5 layers for the MyRoberta are re-initialized as described in Section 3.1.2.

MULTI.1.A is trained using Binary Cross Entropy loss between the true multi-labels with the output, applied with sigmoid activation. It is trained using AdamW optimizer with. A linear learning rate scheduler is used, with 100 warm-up steps and number of total steps corresponding to 25 epochs. This is done to avoid a lower learning rate at 5 epochs. The hyperparameters for this system are given in Table 4.

Hyperparameter	Value
Train Epochs	5
Batch Size	16
Weight Decay	0.01
Initial Learning Rate	5e-5

Table 4: Hyperparameters for MULTI.1.A

The output from the above model is used as input to the final multi-label classifier (MULTI.1.B). Two sets of experiments were conducted, one with the CLS embeddings from MyRoberta, and another with the output of the fully connected classifier layer from MULTI.1.A, as the desired input to MULTI.1.B. These results are given in Section 5. MULTI.1.B consists of individual binary XGBoost (Chen and Guestrin, 2016) classifiers for each of the seven labels. These classifiers account for data imbalance for each given label by

[3]`MyRoberta.config.initializer_range`

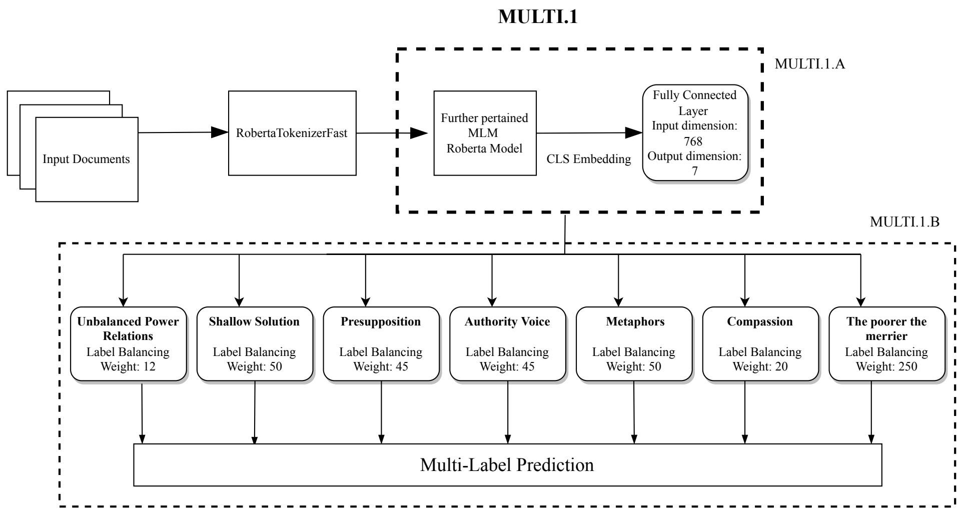

MULTI.1
MULTI.1.A
Input Documents
RobertaTokenizerFast
Further pertained MLM Roberta Model
CLS Embedding
Fully Connected Layer Input dimension: 768 Output dimension: 7
MULTI.1.B
Unbalanced Power Relations Label Balancing Weight: 12
Shallow Solution Label Balancing Weight: 50
Presupposition Label Balancing Weight: 45
Authority Voice Label Balancing Weight: 45
Metaphors Label Balancing Weight: 50
Compassion Label Balancing Weight: 20
The poorer the merrier Label Balancing Weight: 250
Multi-Label Prediction

Figure 2: MULTI.1

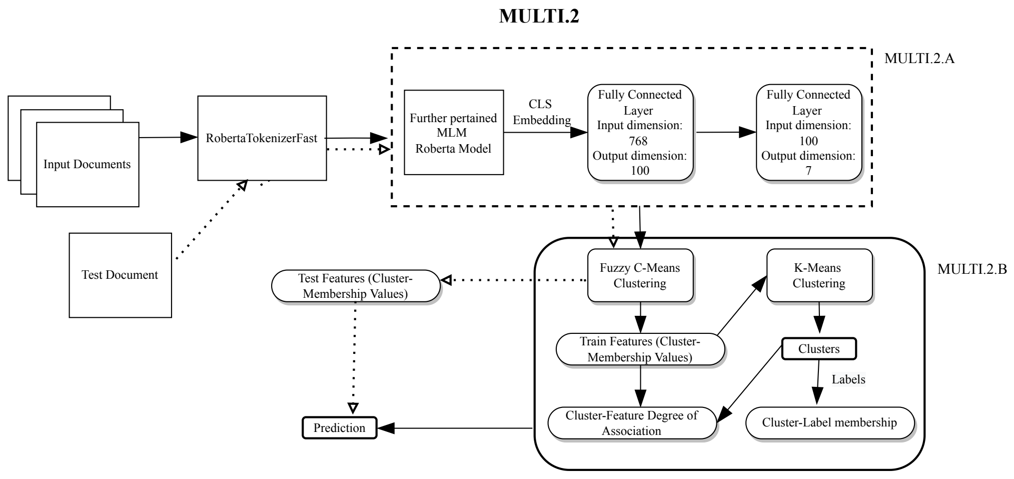

MULTI.2
MULTI.2.A
Input Documents
RobertaTokenizerFast
Further pertained MLM Roberta Model
CLS Embedding
Fully Connected Layer Input dimension: 768 Output dimension: 100
Fully Connected Layer Input dimension: 100 Output dimension: 7
Test Document
Test Features (Cluster-Membership Values)
MULTI.2.B
Fuzzy C-Means Clustering
K-Means Clustering
Train Features (Cluster-Membership Values)
Clusters
Labels
Cluster-Feature Degree of Association
Cluster-Label membership
Prediction

Figure 3: MULTI.2

using the parameter `scale_pos_weight` to indicate the number of negative examples per positive example in the training dataset[4]. The value of `scale_pos_weight` is chosen as the approximation of this value in different partitions of the dataset (Train, Dev and Train+Dev).

3.2.2 MULTI.2: MyRoberta + FCL_1 + FCL_2 + FCM + Fuzzy_CLF

This system uses a fuzzy membership-based ensemble classifier (Tandon and Chatterjee, 2022), consisting of two sub-models, namely MULTI.2.A and MULTI.2.B. The sub-model MULTI.2.A uses fine-tuned MyRoberta for feature extraction, which is fed to the sub-model MULTI.2.B for classification. The architecture for this system is given in Figure 3.

[4]The other parameters for these XGBoost Classifiers are given in the Appendix

The first component of MULTI.2.A is MyRoberta. The CLS embedding from MyRoberta is passed to a fully connected layer with output dimension 100, followed by a fully connected classifier layer with output dimension 7. While training the weights in the pooler layer and the last five layers for the MyRoberta are re-initialized as described in Section 3.1.2. This model is trained using Binary Cross Entropy loss applied with sigmoid activation, using AdamW optimizer. A linear learning rate scheduler is used with 100 warm-up steps and number of total steps corresponding to 25 epochs. This model uses 10 train epochs. All the other hyperparameter values are same as given in Table 4.

Output from the first fully connected layer in MULTI.2.A is used as an input to MULTI.2.B. MULTI.2.B uses this input for Fuzzy C-Means (Bezdek et al., 1984) clustering algorithm, in which the clusters are considered as fuzzy sets over the set of all samples. Each cluster is represented by a fuzzy membership function. The parameters for this model are c (number of clusters) and m (the weighting exponent). This algorithm outputs Fuzzy C-partition $\mathbf{X} = [X_{i,j}]_{l\times c}$ which is used as the set of features for the main classification model. Here, l is the number of training documents. The use of this as the set of features aims at measuring the underlying uncertainty using membership-based measures.

Using these features the set of input documents $\mathcal{D}$ are clustered in k hard clusters. Each cluster thus formed is considered as a fuzzy set on the prescribed label set. In case of k clusters and p labels, the cluster is represented by a $p-$dimensional vector of fuzzy membership values which are utilized to assign the label set for an unseen example. Next, a measure of association of clusters to each of the extracted features (generated from Fuzzy C-Means clustering algorithm) is derived, which aids in retrieving a value to represent a new instance as a $k-$dimensional vector $(R_1, \cdots, R_k)$, where each R_i is the projection of the instance in the i^{th} cluster. Top s ($s \leq k$) clusters with highest R values are chosen. For a threshold value α the predicted label set for this instance is computed as the set of all labels whose membership to the union of these clusters is greater than or equal to α. The mathematical details of the algorithm are given in the Appendix.

4 Experimental Setup

The experiments were carried on Google Colaboratory in Python 3.7.12 with Nvidia Tesla P100 GPU. PyTorch (Paszke et al., 2019) and Huggingface Transformers (Wolf et al., 2020) are the key frameworks used to carry out the experiments.

The text from the training data was used to further pre-train RoBERTa and XLNet models, without any preprocessing. The text was tokenized using the fast implementations of RobertaTokenizer and XLNetTokenizer from the transformers library i.e., RobertaTokenizerFast and XLNetTokenizerFast, respectively.

The input text was tokenized using XLNetTokenizerFast for BINARY.1 and using RobertaTokenizerFast for BINARY.2.

For MULTI.1 and MULTI.2, the text in the training data was preprocessed using the following preprocessing steps.

- The punctuations `" , ; –` were removed from the text.
- Extra spaces between contractions were removed, and digits were removed from the text.
- The contractions were fixed using contractions library.[5]
- HTML symbols were removed.
- Any additional punctuations in particular, `!"#$%&()*+,-./:;<=>?@[\]^_`|~` were removed.

For MULTI.1 and MULTI.2, the preprocessed text is tokenized using RobertaTokenizerFast from the transformers library. The XGBoost classifier has been implemented using the XGBoost[6] library and the MiniBatchKmeans has been implemented using scikit-learn[7] Python library (Pedregosa et al., 2011). Fuzzy C-means is implemented using the 'fuzzy-c-means'(Dias, 2021) Python framework. For MULTI.2, the parameters are given in Table 5. These parameters are chosen by performing hyper-parameter tuning using Train and Dev partitions of the dataset.

Parameter	Value
m	1.6
c	22
k	38
α	0.55
s	4

Table 5: MULTI.2.B Parameters

5 Results

Several experiments were conducted using the systems described in Section 3 and some variations of these systems. The systems were trained on the Training dataset (Train + Dev) and the metrics obtained on Test set are discussed in this section.

5.1 Subtask 1

In BINARY.1, to understand the impact of the layers MyRoberta, MyXlnet, and Attn_head, ablation experiments were carried out. For each of these experiments, each one of these layers are systematically removed and the results are reported. Since

[5]https://github.com/kootenpv/contractions
[6]https://xgboost.readthedocs.io/en/stable/
[7]https://scikit-learn.org/stable/index.html

Model	Precision	Recall	F1-Score
BINARY.1: MyRoberta + MyXlnet + Attn_head	0.607	**0.492**	**0.544**
BINARY.1: MyRoberta + Attn_head	0.606	0.486	0.539
BINARY.1: MyXlnet + Attn_head	0.621	0.470	0.535
BINARY.1: MyRoberta + CLS	0.606	0.470	0.529
BINARY.1: MyXlnet + CLS	0.568	0.489	0.525
BINARY.1: MyRoberta + MyXlnet + CLS	0.588	0.476	0.526
BINARY.2: MyRoberta + Attn_head	**0.631**	0.470	0.539
BINARY.2: MyRoberta + CLS	0.598	0.483	0.534
roberta-baseline	0.394	0.653	0.491

Table 6: Experiments: Subtask 1

Model	1	2	3	4	5	6	7	Average
MULTI.1: MyRoberta + FCL_1 + XGB	0.521	0.427	0.252	0.304	0.288	0.433	0.148	0.339
MULTI.1: MyRoberta + XGB	0.534	0.395	**0.253**	0.276	**0.395**	**0.454**	0.162	**0.353**
MULTI.2: MyRoberta + FCL_1 + FCL_2	0.545	0.410	0.231	**0.308**	0.279	0.432	0.214	0.345
MULTI.2: MyRoberta + FCL_1 + FCM + Fuzzy_CLF	0.480	0.390	0.176	0.268	0.247	0.350	0	0.273
MULTI.2: MyRoberta + FCL_1 + FCM + Base_CLF	0.534	0.421	0.232	0.276	0.235	0.420	0	0.303
MULTI.2: MyRoberta + FCM + Fuzzy_CLF	0.504	**0.457**	0.163	0.283	0.256	0.390	0	0.293
MULTI.2: MyRoberta + FCM + Base_CLF	**0.548**	0.405	0.219	0.256	0.286	0.362	0	0.297
MULTI.2: MyRoberta + FCL_1 + Base_CLF	0.541	0.368	0.227	0.288	0.273	0.413	0.148	0.322
MULTI.2: MyRoberta + Base_CLF	0.539	0.400	0.200	0.286	0.286	0.416	**0.267**	0.342
roberta-baseline	0.354	0	0.167	0	0	0.209	0	0.104

Table 7: Experiments: Subtask 2

BINARY.1 considers an ensemble of MyRoberta and MyXlnet, exactly one of these is eliminated in each set of experiments. Additionally, the attention mechanism on the last hidden state is substituted with using CLS embeddings (denoted CLS) from the previous layer. The results from these experiments are given in Table 6. This table also includes the result for BINARY.2 and the official roberta-baseline. Here, the proposed model in BINARY.2 is denoted by MyRoberta + Attn_head. For each of these models, the final layer is a fully connected layer which serves as a classifier unit.

The best precision is observed for the model MyXlnet + Attn_head among the BINARY.1 ablations, and for BINARY.2, overall. However proposed model i.e., MyRoberta + MyXlnet + Attn_head with the ensemble embedding layers and an attention head performs the best according to the overall F1-Score.

5.2 Subtask 2

In MULTI.1, two variations of the proposed model can be achieved, depending on the input to MULTI.1.B. The CLS embedding from MyRoberta in MULTI.1.A can be considered as an input to MULTI.1.B, or the output of the fully connected layer in MULTI.1.A can be considered as input to MULTI.1.B.

MULTI.2.A consists of three components, namely MyRoberta followed by two fully connected layers denoted as FCL_1 and FCL_2, respectively. MULTI.2.B has two parts, namely the model consisting of Fuzzy C-Means clustering to extract features followed by a multi-label classifier model based on fuzzy membership. These two parts will be denoted as FCM and Fuzzy_CLF, respectively, as mentioned in Section 3.2. Ablation experiments were performed by removing components of these model to understand their impact in the overall system. For these experiments the main classifier layer is substituted with a One-vs-the-rest classifier which fits one classifier per class. The base estimator for this classifier is taken as a support vector classifier (This classifier was designed using the scikit-learn library) . This classifier is denoted as Base_CLF.

The results from these experiments are given in Table 7. This table consists of F1-Scores for each of the seven labels and the macro-average F1-Score.

It is observed that the F1-Score for the label 'Unbalanced power relations' is the highest among all other labels, at 0.548. This value has been observed for the the MULTI.2 ablation MyRoberta + FCM + Base_CLF. The highest F1-Score for the label 'Shallow Solution' is 0.457 using the model MyRoberta + FCM + Fuzzy_CLF. The highest F1-Scores for the labels 'Presupposition', 'Metaphor' and 'Compassion' are observed using MULTI.1 with the CLS embedding from MyRoberta considered as an input to MULTI.1.B. The highest F1-Score for the labels 'Authority Voice' is observed for MyRoberta + FCL_1 + FCL_2. The highest F1-Score for the label 'the poorer, the merrier' is observed for the MULTI.2 ablation MyRoberta + Base_CLF. Overall best performance is observed for MULTI.1 with the CLS embedding from MyRoberta considered as an input to MULTI.1.B. It is noted that all the presented experiments perform better than the roberta-baseline.

6 Conclusion

In this paper, systems for the task of binary and multi-label classification for detection of PCL in text, have been proposed. For the binary classification task, the first proposed system uses an ensemble transformer-based language model architecture. The other system detailed in the present work, for binary classification is a fine-tuned transformer-based language model with some custom layers. The first system proposed for the task of multi-label classification combines outputs from a transformer-based model fine-tuned for multi-label classification, and uses the embeddings from this model as features for individual label-balanced XGBoost models. Another system for multi-label classification has also been discussed, which uses a transformer-based model fine-tuned for multi-label classification, and uses the outputs from this model as inputs to a fuzzy-membership based ensemble classifier.

7 Acknowledgements

The experiments were conducted on Google Colaboratory.

A Appendix: Hyperparameters for Further pre-training

The RoBERTa and XLNet models were further pre-trained on the text from the Train+Dev set, with the parameter setting given in Table 8. This pre-training was carried out using RobertaForMaskedLM (RoBERTa Model with a language modeling head on top) and XLNetLMHeadModel (XLNet Model with a language modeling head on top) for RoBERTa and XLNet models, respectively. The value of batch size was chosen based on computation specifications, and the values of learning rate and weight decay were chosen based on empirical results presented in Sun et al.. To tune the hyperparameter representing the number training steps, the Train subset is used for training and the Dev set is used for validation.

Experiment Setting	Value
Train steps	30000
Batch Size	64
Initial Learning Rate	2e-5
Weight Decay	0.1
Optimizer	AdamW
Learning Rate Scheduler	Cosine
Warm-up steps	10

Table 8: Experimental Setting for further pre-training

B Appendix: Attention Architecture

This is defined with a fully connected layer with input dimension as the size of the document embeddings of dimension $batch\ size \times sequence\ length \times embedding\ dimension$ from the previous layer, and output dimension taken as 512. This is followed by a Tanh activation and another fully connected layer with input dimension as 512, and output dimension as 1. A softmax activation is applied to this layer to get the weights corresponding to each token in the sequence. Using these weights a weighted sum of the document embeddings of the size which were an input to this layer, is calculated producing an output of size $batch\ size \times embedding\ dimension$.

C Appendix: Hyperparameters for XGBoost Classifiers

The seven XGBoost classifiers trained in System B use the hyperparameter values as given in Table 9. For the hyperparameters not mentioned in the table, their default values given in the Python implementation of XGBoost Classifier are used. In the table, the parameter n_estimators denotes the number of gradient boosted trees, eta indicates the learning rate, min_split_loss indicates the minimum loss reduction required to make a further partition on a leaf node of the tree, max_delta_step indicates the maximum delta step allowed to each leaf output, subsample indicates the subsample ratio of the training instances, reg_alpha denotes the L1 regularization term on weights, and tree_method denotes the tree construction algorithm used in XGBoost.

Parameter	Value
n_estimators	100
eta	0.5
min_split_loss	0.1
max_delta_step	2
subsample	0.6
reg_alpha	0.5
tree_method	gpu_hist
objective	binary:logistic

Table 9: XGBoost Classifier Parameters

D Appendix: MULTI.2.B Algorithm

The algorithm for MULTI.2.B is given in two parts, the model generation algorithm and the prediction algorithm, as given below.

D.1 Algorithm for Model Generation

1 MiniBatchKMeans (Sculley, 2010) clustering algorithm is applied to $\mathbf{X} = [X_{i,j}]_{l\times c}$ to obtain k clusters namely $(D_1, \cdots, D_k)$, resulting in $\Omega = [\omega_{i,j}]_{k\times l}$ where

$$\omega_{i,j} = \begin{cases} 1 & \text{, if the } j^{\text{th}}\text{document is in cluster } D_i \\ 0 & \text{, otherwise} \end{cases}$$

2 A degree of association $\zeta(i,j)$ of each cluster D_i with each feature f_j is calculated as average of the feature values of f_j over all the documents in cluster D_i i.e., if the indices of the samples in cluster D_i are $\{e_1^{(i)}, \cdots, e_{|D_i|}^{(i)}\}$, then,

$$\zeta(i,j) = \frac{\sum_{r=1}^{|D_i|} X_{e_r^{(i)},j}}{|D_i|}$$

3 Cluster D_i is modeled as a fuzzy set of labels, with membership of label c_r to D_i as $\mu_{D_i}(c_r)$, for $r = 1, 2, \cdots, p$, which is modeled as the proportion of occurrences of label c_r in the label set of each document in the cluster D_i. Thus,

$$\mu_{D_i}(c_r) = \frac{\sum_{s=1}^{l} \omega_{i,s} G_{s,r}}{\sum_{s=1}^{l} \omega_{i,s}}$$

where $\mathbf{G} = [G_{i,j}]_{l\times p}$ is defined as

$$\begin{bmatrix} \mathbf{Y}_1 & \mathbf{Y}_2 & \cdots & \mathbf{Y}_l \end{bmatrix}^T$$

$\mathbf{Y}_i = (Y_{i,1}, \cdots, Y_{i,p})$ is the label vector for the i^{th} sample where $Y_j = 1$ if the document has label j and $Y_j = 0$ otherwise.

D.2 Algorithm for Prediction

Given an input text the following steps are performed to compute its label set.

1 This text is passed as input to MULTI.2.A and the output from the first fully connected layer is sent to the trained Fuzzy C-Means clustering algorithm. The Fuzzy C-partition from this is denoted by $\mathbf{v} = (v_1, \cdots, v_c)$.

2 Using $\zeta(i,j)$, a vector $\mathbf{R} = (R_1, \cdots, R_k)$ is calculated as,

$$R_i = \frac{\sum_{j=1}^{c} v_j \zeta(i,j)}{\sum_{j=1}^{c} \zeta(i,j)}$$

Here R_i represents the weighted average of $(v_1, \cdots, v_c)$ with weights as the degree of association vector $(\zeta(i,1), \cdots, \zeta(i,c))$ corresponding to the i^{th} cluster.

3 Top s, $s \leq k$ clusters with highest R values, say $\{D_{\gamma_1}, D_{\gamma_2}, \cdots, D_{\gamma_s}\}$ are chosen and their fuzzy union (or fuzzy t-conorm) is considered. In this case the fuzzy t-conorm algebraic sum[8] is used.

[8] $u(a,b) = a + b - ab$

4 For a given threshold α, predicted set of labels is computed as $\{c_i | \mu_{D_{\gamma_1} \cup D_{\gamma_2} \cup \cdots, D_{\gamma_s}}(c_i) \geq \alpha\}$.

References

Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. 2019. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*.

James C. Bezdek, Robert Ehrlich, and William Full. 1984. FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203.

Niladri Chatterjee, Tanya Aggarwal, and Rishabh Maheshwari. 2020. Sarcasm detection using deep learning-based techniques. In *Deep Learning-Based Approaches for Sentiment Analysis*, pages 237–258. Springer.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Shivani Choudhary, Kushagri Tandon, Raksha Agarwal, and Niladri Chatterjee. 2021. MTL782_IITD at CMCL 2021 shared task: Prediction of eye-tracking features using BERT embeddings and linguistic features. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 114–119, Online. Association for Computational Linguistics.

Nadia K Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. *Proceedings of the association for information science and technology*, 52(1):1–4.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Madson Dias. 2021. omadson/fuzzy-c-means. Original-date: 2019-05-13T16:29:01Z.

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, page 29–30, New York, NY, USA. Association for Computing Machinery.

Njagi Dennis Gitari, Zhang Zuping, Hanyurwimfura Damien, and Jun Long. 2015. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(4):215–230.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

D. Sculley. 2010. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web - WWW '10*, page 1177, Raleigh, North Carolina, USA. ACM Press.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *SIGKDD Explor. Newsl.*, 19(1):22–36.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to Fine-Tune BERT for Text Classification?

Kushagri Tandon and Niladri Chatterjee. 2022. Multilabel text classification with an ensemble feature space. *Journal of Intelligent & Fuzzy Systems*, (Preprint):1–12.

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

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

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

GUTS at SemEval-2022 Task 4: Adversarial Training and Balancing Methods for Patronizing and Condescending Language Detection

Junyu Lu, Hao Zhang, Tongyue Zhang, Hongbo Wang, Haohao Zhu, Bo Xu, Hongfei Lin*
School of Computer Science and Technology, Dalian University of Technology, China
(dutljy,zh373911345,zty9818,hldwhb2017,zhuhh)@mail.dlut.edu.cn
(xubo,hflin)@dlut.edu.cn

Abstract

Patronizing and Condescending Language (PCL) towards vulnerable communities in general media has been shown to have potentially harmful effects. Due to its subtlety and the good intentions behind its use, the audience is not aware of the language's toxicity. In this paper, we present our method for the SemEval-2022 Task4 titled "Patronizing and Condescending Language Detection". In Subtask A, a binary classification task, we introduce adversarial training based on Fast Gradient Method (FGM) and employ pre-trained model in a unified architecture. For Subtask B, framed as a multi-label classification problem, we utilize various improved multi-label cross-entropy loss functions and analyze the performance of our method. In the final evaluation, our system achieved official rankings of 17/79 and 16/49 on Subtask A and Subtask B, respectively. In addition, we explore the relationship between PCL and emotional polarity and intensity it contains. Our code is available on Github [1].

1 Introduction

Patronizing and Condescending Language (PCL) expresses a superior attitude towards vulnerable communities (e.g. women, refugees, poor families), and describes them or their situation in a charitable way that evokes feelings of compassion (Pérez-Almendros et al., 2022). Although it is generally used involuntarily and with good intentions, the use of PCL can potentially be very harmful, as it feeds stereotypes, routinizes discrimination and drives to greater exclusion. Due to the subtlety of PCL, PCL detection is difficult for both humans and NLP systems and has aroused broad attention.

To address the challenge of patronizing and condescending language detection in general media, Pérez-Almendros et al. (2022) introduce the Task 4 at SemEval-2022, and build a dataset with annotated paragraphs extracted from news articles in English. Given a paragraph, systems must predict whether it contains condescending language or not (Subtask A), and whether it contains any of the 7 subtypes identified in the PCL taxonomy (Subtask B).

For Subtask A, a binary classification task, we introduce adversarial training based on Fast Gradient Method (FGM) (Miyato et al., 2016), enhancing the robustness of the model. And in Subtask B, a multi-label classification problem, there is a long-tailed distribution of each label. To address the class imbalance problem, we utilize various improved multi-label cross-entropy loss functions: Focal loss (Lin et al., 2017), Class-balanced focal loss (Cui et al., 2019) and Distribution-balanced loss (Wu et al., 2020). We analyze the performance of our methods and demonstrate the contribution of each component of the architecture.

In addition to completing basic evaluation tasks, we also explore the relationship between PCL and emotional polarity and intensity it contains in official dataset. The experimental results demonstrate that the above two have relevance.

The structure of the paper is as follows: We first provide a brief overview of related research, and then introduce our proposed framework. Besides, experiments and evaluations as well as the analysis of results are given. Finally, we discuss the future directions of our work.

2 Related Work

Patronizing and condescending language has been studied extensively in sociolinguistics and the traits of PCL have been suggested by related research. PCL builds stereotypes (Fiske, 1993), which strengthen exclusion, discrimination, rumour spreading (Nolan and Mikami, 2013) and unbalanced power relations (Sap et al., 2019), relying on subtle language (Mendelsohn et al., 2020). It tends

[1] https://github.com/Nutpok/GUTS-at-SemEval-2022-Task-4.git

to avoid stating the reasons for deep-rooted societal problems by concealing those responsible and proposes temporary solutions (Chouliaraki, 2010), which oversimplify the core problems (Head, 2008). The abuse of PCL exacerbates the difficulty of improving the lives of disadvantaged groups (Nolan and Mikami, 2013) and dehumanizes minorities in news media (Mendelsohn et al., 2020). Due to its hazard, PCL is classified as a milder form of toxic speech (Dale et al., 2021).

The increasingly social issue caused by PCL has attracted considerable attention of researchers in the natural language processing (NLP) field. Wang and Potts (2019) introduced the task of condescension detection in direct communication and built a dataset with annotated social media messages. Pérez-Almendros et al. (2020) proposed Don't Patronize Me!, an annotated dataset with PCL, and demonstrated the effectiveness of the model for PCL detection (Kenton and Toutanova, 2019).

3 Methodology

3.1 Preliminaries

We utilize a transformer-based pre-trained language model (PLM), such as BERT and RoBERTa, to represent the input sentences. Each sentence $x = [CLS, t_1, t_2, ..., t_T, SEP]$ is embedded as $s \in R^{n \times d_{emb}}$, where n is the sequence length and d_{emb} is the dimension of the embedding. We add a softmax classifier on the sentence-level embedding, such as the final hidden state h_{CLS} of the $[CLS]$ in BERT:

$$p_i = softmax(Wh_{[CLS]}) \tag{1}$$

where $W \in \mathbb{R}^{C \times d_{emb}}$, and C denotes the number of classes.

3.2 Adversarial Training

Adversarial training (Goodfellow et al., 2015) is a effective regularization method for classifiers to improve robustness to small, approximately worst case perturbations. In SubtaskA, we introduce Fast Gradient Method (FGM) (Miyato et al., 2016), a novel approach in adversarial training, to improve the generalization ability of the model in PCL detection. Figure 1 shows the overall framework of our model.

According to FGM, we apply tiny perturbations to sentence embeddings rather than original input itself. The adversarial perturbation r_{adv} on s is defined as:

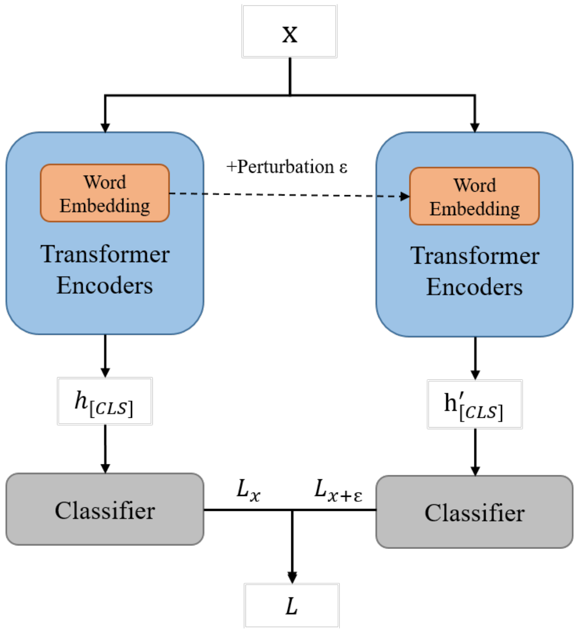


Figure 1: Model architecture for our proposed method in Subtask A.

$$r_{adv} = \epsilon \cdot g/\|g\|_2 \; where \; g = \nabla_s L(s, y) \tag{2}$$

where ϵ is a hyperparameter limiting the size of the adversarial perturbations.

To integrate the information trained from original and adversarial samples, we use an overall loss function as follows:

$$L = L(s, y) + L_{adv}(s + r_{adv}, y) \tag{3}$$

3.3 Balancing Methods

Subtask B becomes a challenging multi-label text classification task because of its long-tailed distribution of labels, each training sample $\{(x^1, y^1), \dots, (x^N, y^N)\}$ has a multi-label group $y^k = [y_1^k, \dots, y_C^k] \in \{0, 1\}^C$, and a classification result $z^k = [z_1^k, \dots, y_C^k]$. In this work, we use different balancing methods (Huang et al., 2021) re-weighting the binary cross entropy to address the class imbalance problem. And the sigmoid function is used for computing $p_i^k = \sigma(z_i^k)$. The codes of these several balanced loss functions are open source[2].

Focal Loss (FL) proposed by Lin et al. (2017) places a higher weight of loss on "hard-to-classify" instances, which are predicted with low probability. The FL can be formulated as follows:

$$L_{FL} = \begin{cases} -\alpha(1 - p_i^k)^\gamma \log(p_i^k) & if \; y_i^k = 1 \\ -\alpha(p_i^k)^\gamma \log(1 - p_i^k) & \text{otherwise} \end{cases} \tag{4}$$

[2]https://github.com/Roche/BalancedLossNLP

where $\gamma \geq 0$ is a non-negative tunable focusing parameter to differentiate between easy and difficult samples and $\alpha \in [0, 1]$ is a weighting factor to balance the training weights of positive and negative samples, p_i^k is the kth choice of p_i.

Class-balanced Focal Loss (CB) (Cui et al., 2019) re-balances the loss according to the effective number of samples for each class. Data sampling can be viewed as a random coverage problem, therefore we assign weights to the different classes based on the number of effective samples. The class-balanced term is defined as:

$$r_{CB} = \frac{1-\beta}{1-\beta^{n_i}} \tag{5}$$

where $\beta \in (0, 1)$ controls the effect of effective number of samples on marginal benefit. And we can use this term to re-weight focal loss:

$$L_{CB} = \begin{cases} -r_{CB}(1-p_i^k)^\gamma \log(p_i^k) & if\ y_i^k = 1 \\ -r_{CB}(p_i^k)^\gamma \log(1-p_i^k) & \text{otherwise} \end{cases} \tag{6}$$

Distribution-balanced loss (DB) Wu et al. (2020) present DBloss to overcome the additional imbalance caused by label co-occurrence upon resampling. In the case of single label, the resampling probability of each instance can be defined as: $P_i^C = \frac{1}{C}\frac{1}{n_i}$; while under multi-label conditions, the instance is repeatedly sampled by each positive class it contains, thus the resampled probability can be defined as $P^I = \frac{1}{C}\sum_{y_i^k=1}\frac{1}{n_i}$. And we can obtain a balancing term: $r_{DB} = P_i^C/P^I$. With a smooth function $\hat{r}_{DB} = \alpha + \sigma(\beta \times (r_{DB} - \mu))$, mapping the weight r_{DB} to a reasonable range, the re-balanced loss function is defined as:

$$L_{R-FL} = \begin{cases} -\hat{r}_{DB}(1-p_i^k)^\gamma \log(p_i^k) & if\ y_i^k = 1 \\ -\hat{r}_{DB}(p_i^k)^\gamma \log(1-p_i^k) & \text{otherwise} \end{cases} \tag{7}$$

To mitigate the over-suppression of negative labels, Wu et al. (2020) introduce a Negative Tolerant Regularization (**NTR**) in the loss function. NTR initializes a non-zero bias v_i as a threshold, and linearly scales the negative logits before the original loss is computed negative, together with a regularization parameter λ to constrain the gradient between 0 and 1. The distribution-balanced loss with NTR can be defined as:

$$L_{DB} = \begin{cases} -\hat{r}_{DB}(1-q_i^k)^\gamma \log(q_i^k) & if\ y_i^k = 1 \\ -\hat{r}_{DB}\frac{1}{\lambda}(q_i^k)^\gamma \log(1-q_i^k) & \text{otherwise} \end{cases} \tag{8}$$

where $q_i^k = \sigma(z_i^k - v_i)$ for positive instances and $q_i^k = \sigma(\lambda(z_i^k - v_i))$ for negative ones. Due to its strong applicability, NTR can also be utilized in Focal loss and DBloss to avoid over-suppression (Huang et al., 2021).

4 Experiments

4.1 Dataset and Evaluation

The dataset from the Task4 of SemEval2022 contains paragraphs about potentially vulnerable social groups[3]. The paragraphs have been extracted from the News on Web (NoW)[4] corpus (Davies, 2013). The total number of training set is 10469 and the final test set contains 2971 samples. The statistics of datasets are shown in Table 1 and the distribution of PCL categories is reported in Table 2.

Label	Samples	Proportion
PCL	993	9.49%
no PCL	9476	90.51%

Table 1: The distribution of labels in SubTaskA.

PCL Categories	Samples	Proportion
Unb. power rel.	716	6.84%
Shallow solu.	196	1.87%
Presupposition	224	2.14%
Authority voice.	230	2.20%
Metaphor	197	1.88%
Compassion	469	4.48%
The p., the mer.	40	0.04%

Table 2: The distribution of labels in SubTaskB.

To estimate the performance of the system, the organizers used different metrics for subtask A and B. In Subtask A, a binary classification task, F1 over the positive class is applied as evaluation measure, while for Subtask B, framed as a multilabel classification problem, results are evaluated based on the macro-average F1 of seven PCL categories.

4.2 Experimental Settings

We utilize Roberta-base (Liu et al., 2019) as the pretrained language model for representing the input paragraphs. The AdamW optimizer is used for model training. In evaluation period, we perform five-folds cross-validation on training set and evaluate the performance of our model using average metrics over five-folds. We keep the model parameters for optimal performance. In test phase, we

[3]https://github.com/Perez-AlmendrosC/dontpatronizeme
[4]https://www.english-corpora.org/now/

utilize each fold of the optimal model to predict on the offical test set and vote on the results to obtain the final predictions.

Specially, we implement our model with `transformers`[5] package. During the training phase, we evaluate the performance of the model every 200 steps and retain the parameters of the model that performed best on the validation set. The hyperparameters settings adopted are shown in Table 3. All models are trained on NVIDIA Geforce GTX 3090 GPU.

Hyperparameters	SubtaskA	SubtaskB
seed	1234	1234
epochs	5	15
batch size	32	8
learning rate	2e-4	2e-4
alpha	0.6	0.95
gamma	2	4
dropout	0.25	-

Table 3: The hyperparameters of the experiment.

4.3 Results and Discussions

The influence of adversarial training. Table 4 shows the influence of adversarial training in Subtask A. Based on the experimental results, we observe that the introduction of FGM can improve the detection capability of the model in both evaluation phase and test phase. It shows that adversarial training can improve the robustness of the model.

Evaluation phase	
Model	**F1(postive)**
RoBERTa	0.5699
RoBERTa+FGM	0.5785
Test phase	
Model	**F1(postive)**
RoBERTa	0.5545
RoBERTa+FGM	0.5790

Table 4: The performance of our model in Subtask A.

The influence of balancing methods. Table 5 shows the results of our framework trained with various loss functions in Subtask B. It is observed that the performance after introducing the balancing methods is significantly more superior than BCE, while the effect is further improved after employing NTR.

In the period of test, we choose two models with the best performance during the evaluation phase to predict the samples, which are trained with FL+NTR and CB+NTR, respectively. Moreover, the model trained with FL+NTR performs better in the final test set. It is because CB is more sensitive to the assumed sample space size β. If there is a significant difference between the training set and the label distribution of the test set, the ability of the model to address label imbalance will be reduced. In the follow-up work, we will conduct more experiments to observe the impact of parameters on hyperparameters.

Evaluation Phase	
Loss Function	**F1(macro)**
BCE	0.2923
FL	0.3662
DB	0.3767
CB	0.3776
FL+NTR	0.3917
CB+NTR	0.3922
Test Phase	
Loss Function	**F1(macro)**
FL+NTR	0.3700
CB+NTR	0.3537

Table 5: The performance in Subtask B.

5 Emotional Polarity and Intensity of PCL

In this section, we conduct a further analysis to explore the relevance between PCL and emotional polarity and intensity it contains.

We employ NLTK[6], a natural language processing toolkit, to determine the emotional features of a paragraph. For a given text, parser of NLTK returns a sentiment score in a interval of [-1,1], which determines if sample is positive or negative and shows emotional intensity. We divide the sentiment score into 5 levels, and the mapping relationships reflecting motional polarity and intensity are shown in Table 6 and Table 7.

Sentiment Score	Emotional Level
$[-1, -0.6]$	-2
$[-0.6, -0.2]$	-1
$[-0.2, 0.2]$	0
$[0.2, 0.6]$	1
$[0.6, 1]$	2

Table 6: Mapping between sentiment scores and emotional level of the polarity.

[5]https://huggingface.co/

[6]https://github.com/nltk/nltk

Sentiment Score	Emotional Level
$[-0.2, 0.2]$	0
$[0.2, 0.4] \cup [-0.4, -0.2]$	1
$[0.4, 0.6] \cup [-0.6, -0.4]$	2
$[0.6, 0.8] \cup [-0.8, -0.6]$	3
$[0.8, 1] \cup [-1, -0.8]$	4

Table 7: Mapping between sentiment scores and emotional level of the intensity.

We divide the training set into 5 subsets based on the sentiment level and calculate the number of samples. Then we count the proportion of paragraph containing PCL in each subset. The experimental result is reported in Figure 2 and 3.

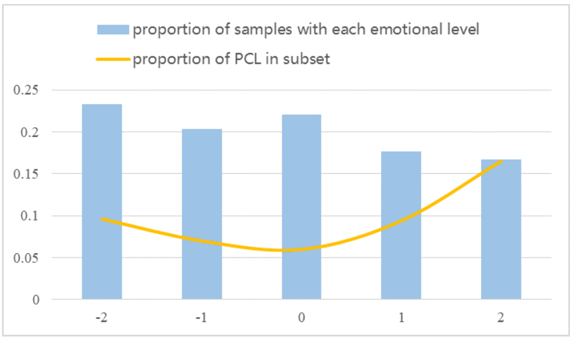


Figure 2: The emotional polarity level of PCL. Blue strip: proportion of samples with each level reflecting emotional polarity in the entire dataset, yellow line: proportion of PCL in subset with each emotional level.

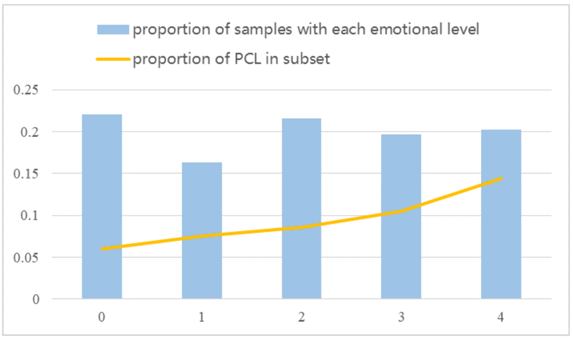


Figure 3: The emotional intensity level of PCL. Blue strip: proportion of samples with each level reflecting emotional intensity in the entire dataset, yellow line: proportion of PCL in subset with each emotional level.

From the results, we can observe that: a) The paragraph containing PCL is more likely to express positive emotions since the use of PCL is often with good intentions. b) Paragraphs with higher emotional intensity are more likely to contain PCL. This is because there are numerous excerpts of live speeches, speakers tend to express their opinions in a stronger tone, which is often condescending.

6 Conclusion and Future Work

In this work, we present our approach to the SemEval-2022 Task 4 to tackle the problem of patronizing and condescending language detection. We employ adversarial training and balancing methods for PCL classification with long-tailed class distribution and demonstrate the effectiveness of our methods.

Besides basic deep learning techniques, introducing multi-task learning in PCL detection, such as predicting the sentiment polarity of a paragraph, is also a problem worth discussing. We have found that PCL is associated with the emotional polarity and intensity of paragraphs. In the future, we will further explore the relationship between sentiment analysis and PCL detection and propose corresponding multitasking frameworks.

Acknowledgement

We thank our anonymous reviewers for their helpful comments. This work was supported by grant from the Natural Science Foundation of China (No.62076046, 62006034), Natural Science Foundation of Liaoning Province (No.2021-BS-067) and the Fundamental Research Funds for the Central Universities (No.DUT21RC(3)015).

References

Lilie Chouliaraki. 2010. Post-humanitarianism: Humanitarian communication beyond a politics of pity. *International journal of cultural studies*, 13(2):107–126.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277.

David Dale, Anton Voronov, Daryna Dementieva, Varvara Logacheva, Olga Kozlova, Nikita Semenov, and Alexander Panchenko. 2021. Text detoxification using large pre-trained neural models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7979–7996, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mark Davies. 2013. Corpus of news on the web (now): 3+ billion words from 20 countries, updated every day. *Retrieved January*, 25:2019.

Susan T Fiske. 1993. Controlling other people: The impact of power on stereotyping. *American psychologist*, 48(6):621.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples.

Brian W Head. 2008. Wicked problems in public policy. *Public policy*, 3(2):101–118.

Yi Huang, Buse Giledereli, Abdullatif Köksal, Arzucan Özgür, and Elif Ozkirimli. 2021. Balancing methods for multi-label text classification with long-tailed class distribution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8153–8161.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

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

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A framework for the computational linguistic analysis of dehumanization. *Frontiers in artificial intelligence*, 3:55.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

David Nolan and Akina Mikami. 2013. 'the things that we have to do': Ethics and instrumentality in humanitarian communication. *Global Media and Communication*, 9(1):53–70.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2019. Social bias frames: Reasoning about social and power implications of language. *arXiv preprint arXiv:1911.03891*.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. *arXiv preprint arXiv:1909.11272*.

Tong Wu, Qingqiu Huang, Ziwei Liu, Yu Wang, and Dahua Lin. 2020. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *European Conference on Computer Vision*, pages 162–178. Springer.

HITMI&T at SemEval-2022 Task 4: Investigating Task-Adaptive Pretraining And Attention Mechanism On PCL Detection

Zihang Liu,Yancheng He,Feiqing Zhuang,Bing Xu[*]
Machine Intelligence and Translation Laboratory,
Harbin Institute of Technology, Harbin, China
21s103280@stu.hit.edu.cn, 21s103127@stu.hit.edu.cn,
21s003015@stu.hit.edu.cn, hitxb@hit.edu.cn

Abstract

This paper describes the system for the Semeval-2022 Task4 "Patronizing and Condescending Language Detection".An entity engages in Patronizing and Condescending Language(PCL) when its language use shows a superior attitude towards others or depicts them in a compassionate way. The task contains two parts. The first one is to identify whether the sentence is PCL, and the second one is to categorize PCL. Through experimental verification, the RoBERTa-based model will be used in our system. Respectively, for subtask 1, that is, to judge whether a sentence is PCL, the method of retraining the model with specific task data is adopted, and the method of splicing [CLS] and the keyword representation of the last three layers as the representation of the sentence; for subtask 2, that is, to judge the PCL type of the sentence, in addition to using the same method as task1, the method of selecting a special loss for Multi-label text classification is applied. We give a clear ablation experiment and give the effect of each method on the final result. Our project ranked 11th out of 79 teams participating in subtask 1 and 6th out of 49 teams participating in subtask 2.

1 Introduction

The effect of Patronizing and Condescending Language (PCL) towards vulnerable communities in the media is not always conscious and the intention of the author is often to help the person or group they refer to (e.g. by raising awareness or funds or moving the audience to action). However, these superior attitudes and discourse of pity can routinize discrimination and make it less visible. While there has been substantial work on modeling language that purposefully undermines others, the modeling of PCL is still an emergent area of study in NLP since PCL is the speaker's unconscious superior speaking attitude, the special word that causes PCL is subtle compared to the keywords in other natural language processing problems.

The authors decided to evaluate the questions separately. In Semeval-2022 task 4: Patronizing and Condescending Language Detection(Pérez-Almendros et al., 2022), the purpose of subtask 1 is to identify whether a sentence is PCL. In contrast, the goal of subtask 2 is to indicate the presence of PCL at the text span level, which detects the exact categories in the seven categories of PCL.In subtask 1, the method of using data set retraining to make the pre-trained language model learn the specific distribution of the data set, adding keywords to the input, and integrating five RoBERTa-based models, subtask 2 is to select k from 7 For classification tasks, task-specific loss calculation methods are designed. These methods will be explained in detail in the following sections.

2 Background

Research on PCL has been in various fields such as language studies (Margić, 2017), sociolinguistics (Giles et al., 1993), politics (Huckin, 2002) or medicine (Komrad, 1983). In recent years, natural language processing systems for recognizing PCL languages have also begun to emerge, for example, (Wang and Potts, 2019) introduced the task of modeling humility in direct communication from an NLP perspective, and developed a dataset of annotated social media messages. In the same year, (Sap et al., 2019) discuss the social and power implications behind the use of certain languages, an important concept in the imbalanced power relations that often arise in condescending treatment. But there has not been a standard in terms of accuracy and definition of PCL. Therefore, this article will first explain the definition of PCL and define some categories of the linguistic techniques used to express PCL.

2.1 What is PCL

Somebody is patronizing or condescending when their language denotes a superior attitude towards others, talks down to them, or describes them or their situation in a charitable way, raising a feeling of pity and compassion. For example,*People across Australia ordered pizzas to be delivered on Saturday night, with the ample leftovers donated to local homeless shelters.* is a sentence that contains PCL for the sentence conveys a superior attitude towards the homeless.

Patronizing and Condescending Language (PCL) is often involuntary and unconscious, and the authors using such language are usually trying to help communities in need by e.g., raising awareness, moving the audience to action, or standing for the rights of the under-represented. On the other hand, due to its subtlety, subjectivity, and the (generally) good intentions behind its use, the audience is often unaware of this diminishing treatment. But PCL can potentially be very harmful, as it feeds stereotypes, routinizes discrimination, and drives to greater exclusion.

PCL detection is difficult both for humans and NLP systems, due to its subtle nature, its subjectivity, and the fair amount of world knowledge and commonsense reasoning required to understand this kind of language. With this task, we expect to push the boundaries of this new challenge in the NLP community.

2.2 Categories of PCL

Our PCL taxonomy has been defined based on previous works on PCL. We consider the following categories:

Unbalanced power relations The author distances themselves from the community or the situation they are talking about and expresses the will, capacity or responsibility to help those in need. It is also present when the author entitles themselves to give something positive to others in a more vulnerable situation, especially when what the author concedes is a right which they do not have any authority to decide to give.

Shallow solution A simple and superficial charitable action by the privileged community is presented either as life-saving/life-changing for the unprivileged one or as a solution for a deep-rooted problem.

Presupposition When the author assumes a situation as certain without having all the information or generalizes their or somebody else's experience as a categorical truth without presenting a valid, trustworthy source for it (e.g. a research work or survey). The use of stereotypes or clichés is also considered to be an example of presupposition.

Authority voice When the author stands themselves as a spokesperson of the group, or ex-plains or advises the members of a community about the community itself or a specific situation they are living.

Metaphor They can conceal PCL, as they cast an idea in another light, making a comparison between unrelated concepts, often with the objective of depicting a certain situation in a softer way. For the annotation of this dataset, euphemisms are considered as an example of metaphors.

Compassion The author presents the vulnerable individual or community as needy, raising a feeling of pity and compassion from the audience towards them. It is commonly characterized by the use of flowery wording that does not provide information, but the author enjoys the detailed and poetic description of the vulnerability.

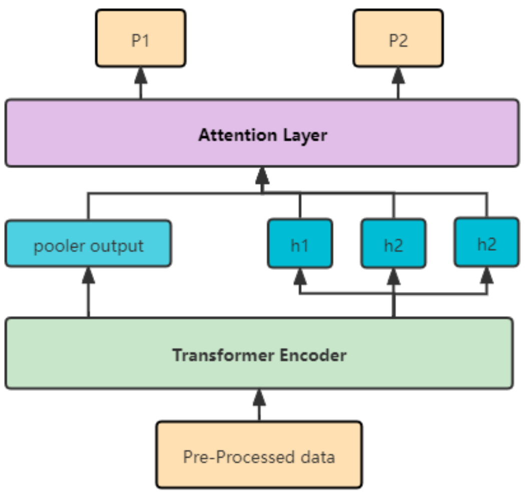
The poorer, the merrier The text is focused on the community, especially on how the vulnerability makes them better (e.g. stronger, happier, or more resilient) or how they share a positive attribute just for being part of a vulnerable community. People living in vulnerable situations have values to admire and learn from. The message expresses the idea of vulnerability as something beautiful o or poetic. We can think of the typical example of 'poor people are happier because they don't have material goods.

3 System description

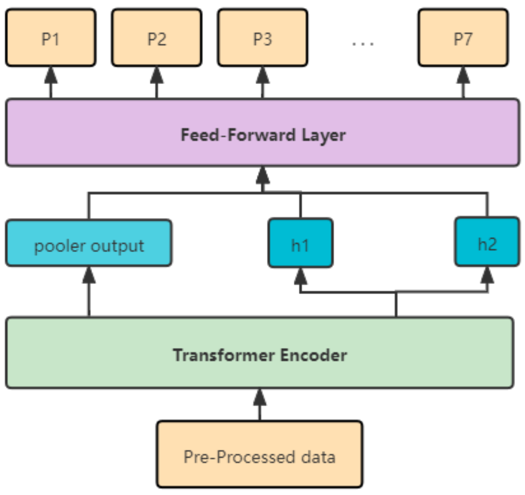
In subtask 1 and subtask 2, we ensemble several models to obtain the results, which are all in RoBERTa-Based architecture (Liu et al., 2019). RoBERTa learns an inner representation of the English language that can be used to extract features useful for downstream tasks. In subtask 1, we pretrain the model on task-specific data. In subtask 2, we utilize multi-label categorical cross-entropy loss to improve performance.

3.1 Data pre-processing

Data for both subtask 1 and subtask 2 contain important information such as the keyword of the sentence and country code. In subtask 1, We truncate the original text centered on the keyword and



(a) RoBERTa-Based architecture of subtask 1.



(b) RoBERTa-Based architecture of subtask 2.

Figure 1: RoBERTa-Based architecture. In our system, the transformer encoder is RoBERTa, the input is pre-processed data mentioned above, pooler output denotes the last layer hidden-state of the first token of the sequence (classification token), hN means hidden states of the keyword extracted during pre-processing, which are the output of the Nth layer from the bottom of the encoder. Attention Layer uses the attention mechanism and calculates attention scores of inputs as weights. FeedForward Layer consists of two linear layers and performs the nonlinear transformation. PN denotes the probability that the Nth label belongs to the sentence.

extract the keyword and its position in the sentence. Also, the article location of the sentence, the keyword of the sentence and the country of the sentence are added to the input as additional features to make the model learn more useful information. Noting that the given country names are in abbreviated form, we restore them to their full form. With this approach, the input formats are shown in Tabel 1. What's more, considering the label imbalance problem of subtask 1, we find the sentences containing PCL from the data in subtask 2 and merge them to form several new sentences as data augmentation.

In subtask 2, We collect different labels of the same sentence to form a single piece of data and use the same way as subtask 1 to pre-process the data. Finally, we lowercase the pre-processed text of both subtasks before they are tokenized.

3.2 Task-Adaptive pretraining

It is proved that Task-Adaptive pretraining can help improve the performance of downstream tasks (Gururangan et al., 2020). In order to make our model better learn the distribution of the data for this task, we pretrain RoBERTa-large model on unlabeled data from subtask 1 and subtask 2. For the same consideration, we process the pretraining data in the way mentioned in Section 3.1.

We apply masked language modeling to pretrain RoBERTa model and use dynamic masking according to the RoBERTa paper. Compared with the original model, the model pretrained in this way can improve the performance to a greater extent.

3.3 RoBERTa-Based architecture

We tried different pretrained models on two subtasks. In our experiments, models initialized with RoBERTa outperform other models. So we choose RoBERTa and pretrain it on task-specific data as our basic model.

Model of subtask 1 In subtask 1, our system uses ensembles of 5 models based on pretrained RoBERTa-Based architecture. As shown in Figure 1(a), the RoBERTa-Based architecture consists of two components: Transformer Encoder, Attention Layer.

First, we pre-process the data to carry more information and tokenize the input into a form accepted by the model. The transformer encoder then is used to extract context representation of the whole sentence. During pretraining, transformer-based language models always use inputs with special tokens(such as [CLS]), so we take out the

extra information	Original text	Pre-processed text
par_id: 1964 keyword: refugee country: my	hospitals fill as rohingya refugees shiver through winter.	from 1964, keyword: refugee, country: Malaysia, hospitals fill as rohingya refugees shiver through winter.
par_id: 4136 keyword: homeless country: za	durban 's homeless communities reconciliation lunch.	from 4136, keyword: homeless, country: South Africa, durban 's homeless communities reconciliation lunch.

Table 1: Examples of Pre-processed text, where "extra information" means additional information in the training data, "Original text" means the original sentence to be judged as PCL or not, "Pre-processed text" means the sentence after pre-processing.

last layer hidden-state of the first token of the sequence(named pooler output), which is the representation of "[CLS]", to obtain a vector representation of the whole sentence. Also, we extract the hidden-state of the keyword of the last three layers, as it is proved that high-level network of transformer encoder learns rich semantic information features(Jawahar et al., 2019).

After we get the pooler output and hidden states of the keyword of each sentence, the two representations are concatenated and fed into an attention layer. We utilize the self-attention mechanism to calculate attention scores as weights in order to make the model attend to essential information. Finally, perform a linear transformation to get reduced representations. The whole process for the model to get the classification results is as follows:

$$Attn(e) = Softmax(A(eW_1+b_1)W_2+b_2) \quad (1)$$

$$Out = (Attn(e) \cdot e)W_3 + b_3 \quad (2)$$

Where e denotes the concatenation of pooler output and the last three hidden states of the keyword of each sentence. A is the Gaussian Error Linear Unit (GELU) (Hendrycks and Gimpel, 2016) activation function, Out denotes the probabilities of sentence-level labels.

Model of subtask 2 In subtask 2, our system uses ensembles of 2 models based on pretrained RoBERTa-Based architecture. As shown in Figure 1(b), the RoBERTa-Based architecture consists of two components: Transformer Encoder, Feed-Forward Layer.

The pre-processed data is obtained by the same method as subtask 1. We also take out the pooler output of the encoder, Furthermore, through experiments, we find that the last two hidden layer outputs of the keyword in each sentence are more effective for this subtask. Then the two representations are concatenated and fed into the Feed-Forward Layer, which is a combination of multiple linear and nonlinear transformations. Finally, we get the probability of each PCL category implied in the sentence.

According to a previous work (Sun et al., 2020), we use the loss function called multi-label categorical cross-entropy. Considering that the task is a multi-label classification problem, a common implementation is to use sigmoid activation, and then turn it into n binary classification problems, using the sum of the cross-entropy of the binary classification as the loss. Supposing k target categories are selected from n candidate categories, when n $\gg$ k, this approach will face a serious class imbalance problem. Therefore, we try to extend softmax and cross-entropy to multi-label classification, which expects each target class score is not less than the score for each non-target class. Instead of turning multi-label classification into multiple binary classification problems, it becomes a pairwise comparison of target class scores and non-target class scores to avoid class imbalance phenomenon. In the implementation, the weight of each label is automatically balanced with the good properties of log-sum-exp. The calculation process of the loss is as follows:

$$log(1 + \sum_{i \in \Omega neg} e^{s_i}) + log(1 + \sum_{i \in \Omega pos} e^{-s_j}) \quad (3)$$

Where Ωneg is the set of negative labels and Ωpos is the set of positive labels. s_N is the score of the Nth label in the corresponding set. In our experiments, we find that using this loss function can help improve the model performance.

task		Train	Valid	Total
subtask 1	PCL	7581	794	2094
	not PCL	1895	199	8375
subtask 2	Unb. power rel.	574	142	716
	Shallow solution	160	36	196
	Presupposition	162	62	224
	Authority voice	192	38	230
	Metaphor	145	52	197
	Compassion	363	106	469
	The p., the mer.	29	11	40

Table 2: Statistics of the dataset

4 Experiment

4.1 Dataset

We trained our models on SemEval-2022 Task 4 training data which is an annotated dataset with Patronizing and Condescending Language(PCL) towards vulnerable communities(Pérez-Almendros et al., 2020). The organizers not only annotated all text spans as containing PCL, but also provided PCL category labels, including a total of seven more fine-grained level categories. At last, there are 10469 marked data in total. The organizers of the competition have divided the data into training set and validation set using the split ratio 8:2. And each text contains an average of 232 tokens. Subtask 1 is a binary classification task, so the labels are just PCL and not PCL, but most of them contain PCL accounts for the majority resulting in imbalance between classes. Subtask 2 is a multi-label binary classification task that aims to predict which PCL categories these texts belong to. And The proportion of each category is more balanced. The statistics of these datasets are given in Table 2.

4.2 Metric

For Subtask 1, it is a binary classification task that will be evaluated using F1 value of the positive class. Subtask 2 is a multi-label classification task, which is evaluated by macro-F1. The calculation formula is as follows. The experimental results are all obtained by averaging three runs with different random initialization.

$$F1 = \frac{2 \times P \times R}{P + R} \tag{4}$$

$$Macro - F1 = \frac{1}{n} \sum_{s=1}^{n} F_z \tag{5}$$

4.3 Experiment Settings

After many experiments, the results show that the effect of using RoBERTa-large is the best and most stable. So at last, all models used in the end are all based on the RoBERTa-large. At the same time, the maximum length of the text is 512. We use Adam as the optimizer with a learning rate of 2e-5. We also use gradual warmup(Goyal et al., 2017) and cosine annealing schedule for learning rate. The coefficient of L2-regularization is 1e-5 and batch size is 32. All the experiments are done on 2 NVIDIA 3090 GPUs, and Limited by the size of GPU memory, we used gradient accumulation.

4.4 Results

For Subtask 1, we designed four models in total: (1)**RoBERTa-ft**: simply fine-tune RoBERTa-large model;(2) **RoBERTa-cls3**: extract the first hidden vector of the last three layers of RoBERTa model and cat them, then pass through a self-attention layer. At last, the hidden vector obtained by multiplying the softmax weight is classified through the linear layer to get predictions;(3) **RoBERTa-cls4**: similar with model1, The only difference is that this model extracts the first hidden vector of the last four layers. (4) **RoBERTa-key**: this model will take out the hidden vector corresponding to the keyword and splice it with the pooler-out vector, then pass through the linear layer to get predictions. For Subtask 2, we build two models at last, including: (1) **RoBERTa-ff**: cat the hidden vectors of the last two layers, and then spliced with pooler-out to pass through a feedforward layer. (2)**RoBERTa-att**: cat the hidden vectors of the last two layers, and then spliced with pooler-out to pass through an attention layer.

Table 3 shows the best F1 values of each model on the official Subtask 1 validation set. And Table 4 shows the F1 values of the above two models on each category and their average values in Subtask 2. For both Subtask 1 and Subtask 2, we set the maximum number of epochs to 10 and open early stop.

We can see from the data in the table that all of the considered methods clearly outperform the baseline. For Subtask 1, the RoBERTa-key achieves the best performance. We also try some other methods, such as extracting the last four hidden vectors of the model, calculating the average value or the maximum value, but their effect is not as good as the above methods. We think that

Method	**acc**	**F1**
baseline	-	0.5211
RoBERTa-ft	0.9254	0.6385
RoBERTa-cls3	0.9288	0.6410
RoBERTa-cls4	0.9303	0.6439
RoBERTa-key	0.9298	**0.6475**

Table 3: Results of detecting PCL, viewed as a binary classification problem (Subtask 1).

	Unb.	**Auth.**	**Sha.**	**Pre.**	**Com.**	**Meta.**	**The p.**	**average**
method	F1	F1	F1	F1	F1	F1	F1	F1
Baseline	0.3844	0.3614	0.3212	0.3745	0.3187	0.376	0.1045	0.3201
Robeta-att	0.5876	0.5423	0.4224	0.4341	0.4359	0.5026	0.1635	0.4412
RoBERTa-ff	0.5958	0.4942	0.3942	0.4492	0.3971	0.4874	0.2887	**0.4438**

Table 4: Results for the problem of categorizing PCL, viewed as a multi-label classification problem (Subtask 2).

Model	F1
RoBERTa	0.5921
+Prefix template	0.6045
+key-hidden	0.6127
+pre-train	0.6386
Last	**0.6475**

Table 5: Ablation results of our model

the keyword can be regarded as an object. For example, if the keyword is poor, the passage is to judge whether the author has an arrogant attitude towards the poor. Therefore, the hidden vector corresponding to the keyword contains more feature information and is very helpful for our judgment. At the same time, extracting the last few hidden vectors contains more information with different granularity. For Subtask 2, We also tried many other different structures, the RoBERTa-ff achieves the best performance we find after a lot of experiments.

4.5 Ablation

We used some stricks and methods in the competition, and we show the improvement effect of each method through the ablation experimental results on Subtask 1 in Table 5. It can be seen that the improvement brought by pre-training is the most significant which improves by more than two points. The second is to add a prefix template. In addition to these methods in the table, there are also slight improvements by some stricks such as resampling.

5 Conclusion

The paper describes our system at SemEval-2022 Task 4, which uses several different models based on RoBERTa. We used a series of methods such as pre-training, constructing prefix templates, and model fusion to achieve relatively good results. As we can see, using RoBERTa as the network backbone achieves better performance in this task. Also post-training and using the hidden vectors of the last few layers of RoBERTa can improve the effect. At the same time, we also tried FGM, focal loss and other methods, while none of them seemed to be beneficial for our task. Still, the F1 value is less than satisfactory, so we can see that identifying and categorizing Patronizing and Condescending Language are difficult challenges. In the future, we will consider using some external knowledge to help the judgment of the model.

References

Howard Giles, Susan Fox, and Elisa Smith. 1993. Patronizing the elderly: Intergenerational evaluations. *Research on Language and Social Interaction*, 26(2):129–149.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Mark S Komrad. 1983. A defence of medical paternalism: maximising patients' autonomy. *Journal of medical ethics*, 9(1):38–44.

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

Branka Drljača Margić. 2017. Communication courtesy or condescension? linguistic accommodation of native to non-native speakers of english. *Journal of English as a lingua franca*, 6(1):29–55.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2019. Social bias frames: Reasoning about social and power implications of language. *arXiv preprint arXiv:1911.03891*.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. 2020. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. *arXiv preprint arXiv:1909.11272*.

UMass PCL at SemEval-2022 Task 4: Pre-trained Language Model Ensembles for Detecting Patronizing and Condescending Language

David Koleczek and **Alex Scarlatos** and **Preshma Linet Pereira** and **Siddha Karakare**
University of Massachusetts Amherst
{dkoleczek, ajscarlatos, ppereira, skarkare}@umass.edu

Abstract

Patronizing and condescending language (PCL) is everywhere, but rarely is the focus on its use by media towards vulnerable communities. Accurately detecting PCL of this form is a difficult task due to limited labeled data and how subtle it can be. In this paper, we describe our system for detecting such language which was submitted to SemEval 2022 Task 4: Patronizing and Condescending Language Detection. Our approach uses an ensemble of pre-trained language models, data augmentation, and optimizing the threshold for detection. Experimental results on the evaluation dataset released by the competition hosts show that our work is reliably able to detect PCL, achieving an F1 score of 55.47% on the binary classification task and a macro F1 score of 36.25% on the fine-grained, multi-label detection task.

1 Introduction

Everyone has seen, experienced, and expressed patronizing and condescending language (PCL). Someone is patronizing or condescending when they communicate in a way that talks down to others, positions themselves in a superior position to the subjective group, or describes them in a charitable way in order to raise a feeling of compassion for the target person or group. Such language is seen frequently in social media or other mediums where hate is the norm. Many previous works have addressed offensive language and hate speech (Zampieri et al., 2019) (Basile et al., 2019) in such communities. However, until (Perez-Almendros et al., 2020) there has not been work in modeling PCL where 1) it is targeted towards vulnerable communities such as refugees or poor families 2) occurs in the *general media or news*. The authors note that this form of PCL is not usually meant to be harmful, and can often have good intentions, such as when the author is trying to raise awareness about an at-risk group. Yet, research in sociolinguistics has shown that regardless of the intent, PCL has negative effects on potential vulnerable communities such as perpetuating stereotypes, reinforcing societal misbehaviors, and oversimplifying deep-rooted problems. The authors of *Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities* originally proposed a new dataset to help in detecting such cases of PCL and are currently hosting a SemEval-2022 competition: Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022).

In this paper we describe our contribution to both competition subtasks: binary and multi-label classification. Our final ensemble leverages many techniques including pre-trained language models, data augmentation, intermediate fine-tuning on related datasets, and optimizing the detection threshold to empirically maximize F1. For the binary task, our system achieves an F1 score of 55.47%, or 6.36% higher than a comparable RoBERTa-based baseline model. We achieve a macro F1 score of 36.25% on the multi-label task, or 25.84% higher than a similar RoBERTa baseline. These results show encouraging progress for the task of detecting fine-grained and linguistically subtle types of PCL, although there is still room for improvement.

2 Background

The Patronizing and Condescending Language Detection task contains two subtasks. In the first subtask, binary classification, the system is given a paragraph of text and it must predict if it contains *any* form of PCL. In the second subtask, multi-label classification, a system is given a paragraph of text and it must identify if it contains each of seven types of PCL (one paragraph can contain multiple categories of PCL).

2.1 Dataset

The binary classification task dataset contains 10,636 total labeled text samples that were selected

Text	Keyword	Country	Label
This wally of an MP just wants his name in the press , what has he ever achieved fro Southend , what did this idle wally do to get the scanner tuned on at the hospital ... zero ! He raised in parliament the fact that his was his mummies birthday , but forgot to mention the homeless families on our streets .	homeless	gb	3
Our life has completely changed from when he as an able-bodied young man running around 5000 miles an hour organising everyone . Now , he 's more disabled than anyone that he ever helped .	disabled	nz	4
The World Health Organization did not give a reason for the increase in deaths , but a provincial health official in Sindh said that the disease hit areas where poor families did not vaccinate their children .	poor-families	us	2
However , she said immigrant patients urgently need treatment and counselling from health-care providers who speak Punjabi or Hindi , and that 's what Roshni – which means light – will offer them.	immigrant	ca	1
In February 2015 , five of the men , aged 23 to 25 at the time , went to the station to protest the "" arbitrary and violent "" arrest of one of their friends from the Cova da Moura neighbourhood , known for its large population of immigrants from Cape Verde , a former Portuguese colony off Africa 's northwest coast .	immigrant	in	0

Table 1: Samples from the competition dataset

	Train	Validation	Test
Positive	683	111	199
Negative	6436	1145	1894

Table 2: Label distribution in train, validation and test splits for the binary classification subtask

from news stories from the News on the Web corpus (Davies, 2013). All the selected paragraphs are from news stories that were published between 2010 and 2018 from 20 English speaking countries. Each paragraph is assigned a label ranging from 0 (not containing PCL) to 4 (being highly patronizing or condescending). Each paragraph is also associated with one of ten keywords used to retrieve the text from the corpus. Samples from this dataset can be found in Table 1 and the distribution of positive and negative labels in each of our data splits can be found in Table 2.

The dataset for the multi-label classification task contains paragraphs from the binary classification task which were labeled as containing PCL. Each paragraph is further labeled with at least one of the seven classes: unbalanced power relations, shallow solution, presupposition, authority voice, metaphor, compassion, and the poorer, the merrier. Each is associated with a country-code and keyword as before. Additionally, each paragraph has the start and end spans of where in the paragraph the particular category of PCL is contained. Further background information about this dataset can be found in the paper published by the competition hosts (Perez-Almendros et al., 2020).

2.2 Baselines

Recent advances in natural language processing have enabled complex and subtle tasks to be solved by fine-tuning large, pre-trained language models (Kalyan et al., 2021). These models can also be fine-tuned on small datasets (such as only 1,000 samples) and while still achieving significant predictive power (Mosbach et al., 2020), or even with just a few samples (Brown et al., 2020). As such, the competition authors provided baseline results using a pre-trained RoBERTa model, one of the go-to models for fine-tuning for a classification task (Liu et al., 2019). They provide results of multiple baselines which help put into context the improvements made by the participating systems. For example, one such baseline is simply random guessing, or assigning each paragraph a random label with equal probability. In our work we build upon the RoBERTa baseline by applying techniques often used in machine learning. Namely, we use early stopping using a validation dataset and set hyperparameters according to best practices discussed by (Mosbach et al., 2020). We also will experiment with a host of NLP techniques including data augmentation and other language models.

3 Subtask 1: System Overview

Considering the challenge of the task and the scarcity of positive examples of PCL, our approach was to experiment with methods focused on enriching our small dataset, such as data augmentation and external datasets. We started with fine-tuning a pre-trained RoBERTa (Liu et al., 2019) model to use as a foundation. RoBERTa has been shown to be very powerful for a range of NLP tasks such as text classification, while also being straightforward to implement and train using libraries such as HuggingFace (Wolf et al., 2019). We use its architecture unchanged with pretrained weights downloaded from HuggingFace. We add a standard binary classification head on top of RoBERTa's pooled outputs. For preprocessing input into the model, we use HuggingFace's auto-tokenizer. In the following subsections, we detail the additional methods and modifications to the RoBERTa baseline. The models used in our final ensemble use a mix of these methods and the exact ensemble methodology is described in section 3.6.

3.1 Pre-trained Language Models

In addition to RoBERTa, we use MPNet (Song et al., 2020). This model inherits the pre-training advantages of both Masked Language Modeling (Devlin et al., 2018) and permuted language modeling (Yang et al., 2019) under a unified view which splits tokens in a sequence into non-predicted and predicted parts. Experimental results show that MPNet outperforms previous models like BERT, XLNet and RoBERTa by 4.8, 3.4 and 1.5 points respectively on GLUE tasks. In our work, we fine-tune MPNet using the hyperparameters suggested by the authors, which can be found in Appendix A.

3.2 Intermediate Fine-Tuning

We leverage several external data sources which are related to our task of PCL detection. (Phang et al., 2018) originally proposed *Supplementary Training on Intermediate Labeled Data Tasks* (STILTs), which we refer to as intermediate fine-tuning. They observed that by 1) starting from a pre-trained language model such as BERT, 2) further training it on an intermediate, *labeled* task, and 3) finally fine-tuned on the target task; improved final performance on a variety of NLP tasks. (Pruksachatkun et al., 2020) and (Vu et al., 2020) provide further insight into tasks that are well suited for intermediate fine-tuning. We used these insights a to develop four independent binary classification tasks on Stereoset (Nadeem et al., 2020), BiasCorp (Onabola et al., 2021), Social Bias Inference Corpus (SBIC) (Sap et al., 2020), and TalkDown (Wang and Potts, 2019) datasets, and used them for intermediate fine-tuning. Two of these tasks, SBIC and Biascorp, are used in our final ensemble, while the other two were left out due to having no performance uplift on our task's test set.

BiasCorp is a dataset containing about 43,000 labeled examples of racial bias from Fox News, BreitbartNews and YouTube. Each text has 3 bias ratings from 0 to 5 and corresponding annotator confidence scores from 1 to 10. To turn this into a binary classification task, we use a similar approach from their paper by first computing a confidence weighted score, and then we threshold this score where scores above 1 are labeled as positive examples.

The **Social Bias Inference Corpus** is a labeled dataset of over 125,000 examples of social media posts from Reddit, Twitter, and hate sites like Stormfront. For our binary classification task, we use the provided offensiveness rating of a post, and use a threshold of 0.3 for positive examples.

3.3 Sentence-Level Features

Transformers are generally able to vastly outperform regression on engineered features. However, in some text labeling tasks, such as essay scoring, it has been shown that engineered features can be used in tandem with Transformer output to improve performance (Uto et al., 2020). This can be achieved simply by concatenating a vector of features $\mathbf{f_n}$ to BERT's CLS vector. An extended classification head will now take the extended vector as input, and similarly project to the final prediction. The classification head, including the added weights, is trained along with the fine-tuning of BERT.

Our set of features was inspired by (Uto et al., 2020), but we excluded the readability metrics because they are not as relevant for our task. Specifically, for text sample $\mathbf{x_n}$, we calculate *the number of words, number of sentences, number of exclamation marks, question marks, and commas, average word length, average sentence length, the number of nouns, verbs, adjectives, and adverbs, and the number of stop words*. Each one of these values is used as a separate dimension of $\mathbf{f_n}$. We leveraged the spaCy Python library (Honnibal and Montani, 2017) to calculate these values. We also included the feature calculation as a data pre-processing step, since calculating the features at run-time would be too costly.

3.4 Original Labels

Simply collapsing the 0-4 label in the dataset to a binary value results in a loss of relevant information. To combat this, we translate each label to a *probability* of a sample containing PCL. Specifically, we perform the following mapping: 0 - 0.0, 1 - 0.25, 2 - 0.5, 3 - 0.75, 4 - 1.0. We refer to these probabilities as the *original labels*. We adapt our loss function for this reformulated learning setup by computing the binary cross-entropy loss between the predicted probability and the original label.

3.5 Backtranslation

Data augmentation for NLP tasks is becoming increasingly popular with a vast variety of existing techniques. Data augmentation refers to any strategy aimed at increasing the amount of data available for training, by only leveraging the existing training set or domain knowledge about the task (Feng et al., 2021). (Sennrich et al., 2015) is a model-based technique for data augmentation where the original text is translated into a desired language and then back to the original language to rephrase it. This process can introduce a different style of sentence for the same meaning. We have used French as our intermediate language for backtranslation on our data. To implement backtranslation for our task, we used the MarianMT model (Tiedemann and Thottingal, 2020) provided by HuggingFace to convert between French and English.

3.6 Ensemble Model

Ensembling is a technique that leverages multiple trained models to improve task performance on the test set. For our final ensemble of three independently trained models, we used a relatively simple approach. Given a set of models M and a test dataset D, let $\hat{y}_{ni}$ be M_i's estimate of the probability that text sample x_n is a positive sample. This value is taken directly from M_i's softmax layer output. Each label is then assigned a final positive probability of $\hat{y}_n = \frac{1}{|M|}\sum_{i=1}^{|M|}\hat{y}_{ni}$. We use this augmented softmax score to make the final prediction for each label in the test set.

3.7 Choosing an Empirically Optimal Threshold for F1 Score

The F1 score is the metric used for evaluation in the competition and it is defined as the harmonic mean of precision and recall:

$$F_1 = \frac{2 * precision * recall}{precision + recall}$$

Note that the output of our transformer models for binary classification is the softmax of a final linear layer. In order to compute the F1 score, those continuous outputs must be binarized. Typically, that is done via a simple transformation:

$$f(x, threshold) = \begin{cases} 1, & x \geq threshold \\ 0, & x < threshold \end{cases} \quad (1)$$

, where x is the softmaxed model output and threshold is a value commonly set to be 0.5. Since the model is optimized using cross entropy loss, it is unclear if 0.5 is the best threshold to use in order to maximize F1 score. Since the F1 score is a *harmonic mean*, we would like to balance precision and recall in order to achieve the highest score. Roughly, precision can be increased (and recall decreased) by increasing the threshold, and vice versa by decreasing the threshold.

In order to find the best threshold, we use a simple, brute force algorithm to determine the best threshold on the validation data set. The algorithm takes vectors y_true and y_out, which are the true binary labels and the normalized model output, respectively. For each model output, we treat it as a *possible threshold* and compute the F1 score using the true labels and y_out binarized via $f(y\_out,$ model output) (assume f operates on vectors). The algorithm outputs the threshold that achieved the highest F1 score.

Algorithm 1: Brute-force F1 Threshold

```
Input: y_true, y_out
best_score ← 0
best_threshold ← 0.5
for threshold in y_out do
    bin_y_out ← f(y_out, threshold)
    score ← f1_score(y_true, bin_y_out)
    if score > best_score then
        best_score ← score
        best_threshold ← threshold
return best_threshold
```

4 Subtask 2: System Overview

Our system for task 2, multi-label classification, adapts many of the techniques we used in the binary classification task. Similar to our task 1 system, we leverage RoBERTa as a base model for further fine-tuning. However, the target is now a vector of binary labels. Another way to view the problem is as 7 binary classification tasks, one for each category of PCL. Our final ensemble includes models treating the problem in this way, and also training one multi-label classifier directly. We also independently train a binary classification model to generate a prediction for each class using the same model setup from the binary task. In either case, the final ensemble of models was computed identically to the binary classification task by treating each category independently. The sigmoid function was applied to each model output before creating the ensemble.

4.1 Data Processing

The dataset for the multi-label classification task contains only paragraphs from the binary classification task which were labeled as containing PCL. This creates a disconnect between the data that would be seen during evaluation (i.e. there are samples without any category of PCL). To handle this discrepancy in data distributions, we take every non-PCL paragraph from the binary classification task dataset and add it to the multiclass dataset with a label vector of all 0s.

5 Experimental Setup

We start by splitting our dataset into a train and test set according to the split provided by the competition authors, which is 80% train and 20% test data. From the training split, we take 15% to use as a validation split. Table 2 shows the distribution of positive and negative labels within the train, validation and test splits.

Using our validation split, for each experiment, we perform a hyperparameter sweep by varying the peak learning rate $\in$ {3e-5, 2e-5, 1e-5, 5e-6}. We use early stopping where the model is evaluated at the end of each epoch, and if for 10 consecutive epochs performance does not improve, we stop training and save the model from the epoch with the highest valdiation F1 score. For final evaluation on the test set we use this model that achieved the highest F1 score on the validation set. We choose a batch size that maximizes GPU usage, and use the default maximum sequence length of each model. A listing of the full hyperparameters used can be found in Appendix A. Random seeds for Python, `numpy`, and `PyTorch` were fixed at 221 for all experiments.

For our final results and submission to the competition, we use the validation split for early stopping and train including the test dataset. All models were trained with a learning rate of 2e-5 unless otherwise noted. We submitted two identical systems, only differentiated by the random seed used. All

Binary	Description	P	R	F1
Model 1	RoBERTa, intermediate fine-tuning, features	67.95	47.75	56.08
Model 2	MPNet, original labels	65.95	55.85	60.48
Model 3	RoBERTa, features, backtranslation	61.11	49.55	54.73
Ensemble		62.04	60.36	61.19

Table 3: Subtask 1 - Validation split scores for each of our models that composed the final ensemble. The scores for each model use the default threshold, while the ensemble's threshold is optimal for the validation split.

Multi-label	Description	P	R	F1
Model 1	RoBERTa, features, backtranslation	42.90	30.52	35.01
Model 2	RoBERTa, backtranslation, LR=1e-5	46.53	36.18	39.13
B4MC	Binary model 1 for each category	2.51	100.0	4.83
Ensemble		49.38	35.72	40.50

Table 4: Subtask 2 - Validation split scores for each of our models that composed the final ensemble.

results shown in Tables 3 and 4 are the random seed that performed better.

Our experiments were run a system with the following hardware; GPU: NVIDIA RTX 3090, CPU: AMD 5950X, RAM: 64GB 3200Mhz, SSD: 4TB SATA.

6 Results

Submitted systems were evaluated on the F1 score (macro F1 score for the multi-label task). We built several models that each combine different methods. Observe that in Tables 3 and 4 each model has a varying degree of balance between precision and recall. This diversity in outcomes and methodology between individual models helps our final ensemble achieve better overall results than any individual model. Our best ensemble for subtask 1 placed 22nd out of 78 and our best ensemble for subtask 2 placed 18th out of 49.

For subtask 1, our best model on the validation split was the one that used MPNet as a base and used the original labels during training. Model 1 used RoBERTa as a base and was first trained using intermediate fine-tuning, before being trained on

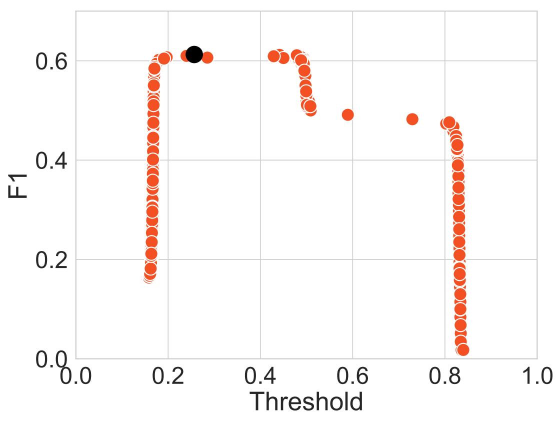


Figure 1: Plot of the resulting F1 score when using each validation softmax score (unbinarized) as the detection threshold.

the final dataset using sentence level features. Intermediate fine-tuning was done by first training using the SBIC task for 5 epochs, followed by training for 5 epochs on the Biascorp task. Each intermediate task was trained with a learning rate of 5e-6. Finally, model 3 used RoBERTa as a base and was trained with sentence level features and additional backtranslated samples.

For subtask 2, our best model used RoBERTa as a base and was trained with backtranslated samples and a learning rate of 1e-5. Model 1 was similar except it used sentence level features and our default hyperparameters found in Appendix A. Binary for multi-category (B4MC) was 7 individual models (one for each category), trained using the same methodology as our best binary model, model 1.

6.1 Thresholds

We use Algorithm 1 to find an optimal F1 threshold on the validation split in order to increase the F1 score for our final ensemble. Analysis of this optimization for our binary task ensemble is in Figure 1. Based on that plot, we set our prediction threshold to be 0.32, about in the middle of the range of thresholds that achieved the highest F1 scores. Notice that if we had picked a threshold of 0.5, we likely would have achieved a lower score due to the drop in scores around that mark.

We performed similar analysis for subtask 2. The thresholds we choose were 0.5, 0.31, 0.49, 0.27, 0.29, 0.21, and 0.4 for the categories unbalanced power relations, shallow solution, presupposition, authority voice, metaphor, compassion, and the poorer the merrier, respectively.

6.2 Error Analysis

To further analyze how our model compares to the baseline, we examine samples from the test set where 1) the baseline made an error, but our model made the correct prediction, and 2) where both the baseline and our model made the incorrect prediction. We examine false positives and false negatives in both cases.

In Table 5, we can see that the baseline's false negatives tend to be more poetic examples. They paint vivid, dramatic scenes, and use strong adjectives like "shocking" and "hopeless". This indicates that our model may have picked up on this poetic language signal as an indicator of PCL.

In the baseline's false positives, we see phrases that may be indicative of PCL, such as "in need of food", "treat men and women differently", and "being disabled". However, in context, these phrases are not condescending to the subjects of the statements. It's possible that the baseline was simply acting on the presence of these phrases, while our model was able to identify the lack of other supporting signals as evidence that the sentences were not PCL.

In Table 6, we can see where both the baseline and our model failed, where there are a variety of possible issues. The first two false negatives are only identifiable as PCL given their subtext (assuming the perspective of refugees and creating a division between disabled and non-disabled people), rather than obvious choices of words. While the third false positive uses flowery language (a possible PCL indicator), the identifying piece, "people from poor families have more perseverence...", uses a positive sentiment. The models seem to struggle with the "poorer the merrier" category, which this is an example of. It's possible that the model is overwhelmed with more negative sentiments from other categories so it tends to correlate that too strongly with PCL.

The false positives all use the kinds of words that seem to be common in the actual positive examples, such as "plight", "challenges", "disadvantaged", and "vulnerable". What the models seem to fail at is distinguishing between pointing out a group's disadvantage and using that disadvantage to patronize the group.

Overall, the models seem to be very active on words and phrases that might indicate PCL. While

Text	**Label**	**Baseline Pred.**	**Ensemble Pred.**	**Categories**
real poverty of britain : shocking images of uk in the sixties where poor really meant poor these hard-hitting photographs offer a glimpse into the harrowing day-to-day for poor families living in britain during the sixties .	pos.	neg.	pos.	Authority Voice, Compassion
rather sad . good set of pictures that tells a tale of survival , subsistence living , and hopeless nature of life in the tribal societies . exploiting an unexpected geo-political bonanza is a temporary relief that is not sustainable . education and economic development seem miles away .	pos.	neg.	pos.	Presupposition, Metaphors, Compassion
as a family , my father was a policeman and he used to come home with food (monthly) , and my mother used to pack small parcels and we used to give them to the poor families .	pos.	neg.	pos.	Unb. power rel., Shallow solu.
mrs charo said some people were injured while escaping the floods . she added that they are now in need of food , clothing and clean drinking water .	neg.	pos.	neg.	-
she added : i would also like to carefully point out that the issue was not her religious beliefs , but rather it is about choosing to treat men and women differently by shaking the hands of women but not men .	neg.	pos.	neg.	-
i realised that it was not impossible to achieve anything in the world despite being disabled .	neg.	pos.	neg.	-

Table 5: Samples where the ensemble improved over the baseline.

Text	**Label**	**Baseline Pred.**	**Ensemble Pred.**	**Categories**
many refugees do n't want to be resettled anywhere , let alone in the us .	pos.	neg.	neg.	Presupposition
the law stipulated 21 rights of the disabled persons . the disabled persons must get the national identity cards and be listed in the voters roll . even they will be able to contest in the polls .	pos.	neg.	neg.	Unb. power rel.
however , this success story is not uncommon . it often happens that people from poor families have more perseverance to fight tooth an nail in business than children of rich parents who are used to get everything they want with ease . people without a strong spirit will quickly break down and drop out from the competition .	pos.	neg.	neg.	Presupposition, Metaphors, The p., the mer.
your personal leadership has been critical to addressing the plight of the rohingya who fled to safety in your country . i thank you for all you have done to assist these men , women and children in need , he wrote in the message .	neg.	pos.	pos.	-
she also praised the efforts to stabilise the lives of refugees , by providing for their basic needs and helping them overcome their challenges , while stressing that supporting refugees is an ongoing part of the uae 's humanitarian directives , and the country has taken the responsibility to evaluate their needs and provide them with a variety of urgent and essential aid .	neg.	pos.	pos.	-

Table 6: Samples where both the baseline and ensemble were incorrect.

our best model appears able to deal with these features in somewhat intelligent ways, the main downside is that it cannot place the samples in a global context. Simply using certain adjectives to describe a certain group of people does not necessarily make a statement condescending. It depends on exactly which adjectives, which group, and what other context is in the statement. As humans, we have two main advantages in this task: 1) we understand the struggles and stereotypes of disadvantaged groups, and 2) we also can perform deductive reasoning to determine the underlying meaning of a phrase. Possibly remedies for our model might be 1) providing many more examples, so it can build the kind of context needed to make these distinctions, and 2) to enhance the model with ways of reasoning about the problem.

7 Conclusion

We developed a system to detect patronizing and condescending language by media towards vulnerable communities. We combined a multitude of NLP techniques in our final ensemble. Our system leveraged backtranslation, sentence level features, intermediate fine-tuning, and pre-trained language models. We also proposed multiple training techniques and optimizing the F1 threshold. These improvements resulted in our system placing in the top 25% for subtask 1 and the top 40% for subtask 2.

Acknowledgements

This work began as part of an advanced natural language processing course at the University of Massachusetts Amherst. We thank the teaching assistants Katherine Thai and Tu Vu and the instructor Mohit Iyyer for their wonderful teaching and feedback.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mark Davies. 2013. Corpus of news on the web (now): 3+ billion words from 20 countries, updated every day. Available online at `https://www.english-corpora.org/now/`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542*.

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

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456*.

Olawale Onabola, Zhuang Ma, Yang Xie, Benjamin Akera, Abdulrahman Ibraheem, Jia Xue, Dianbo Liu, and Yoshua Bengio. 2021. hbert+ biascorp–fighting racism on the web. *arXiv preprint arXiv:2104.02242*.

Carla Perez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. 2020. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *arXiv preprint arXiv:2005.00628*.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Masaki Uto, Yikuan Xie, and Maomi Ueno. 2020. Neural automated essay scoring incorporating hand-crafted features. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6077–6088, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across nlp tasks. *arXiv preprint arXiv:2005.00770*.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. *arXiv preprint arXiv:1909.11272*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

A Fine-tuning Hyperparameters

Hyper-parameter	Fine-tuning
Learning Rate	2e-5
Batch Size	16
Weight Decay	0.1
Epochs	Early stopping
Learning Rate Decay	Linear
Warmup Ratio	0.06

Table 7: Hyper-parameters for fine-tuning RoBERTa and MPNet models.

YNU-HPCC at SemEval-2022 Task 4: Finetuning Pretrained Language Models for Patronizing and Condescending Language Detection

Wenqiang Bai, Jin Wang and Xuejie Zhang
School of Information Science and Engineering
Yunnan University
Kunming, China
Contact: ynubwq@mail.ynu.edu.cn, {wangjin, xjzhang}@ynu.edu.cn

Abstract

This paper describes a system built for the SemEval-2022 competition. As participants in Task 4: Patronizing and Condescending Language Detection, we implemented the text sentiment classification system for two subtasks in English. Both subtasks involve determining emotions; subtask 1 requires us to determine whether the text belongs to the PCL category (single-label classification), and subtask 2 requires us to determine to which PCL category the text belongs (multi-label classification). Our system is based on the bidirectional encoder representations from transformers (BERT) model. For the single-label classification, our system applies a BertForSequenceClassification model to classify the input text. For the multi-label classification, we use the fine-tuned BERT model to extract the sentiment score of the text and a fully connected layer to classify the text into the PCL categories. Our system achieved relatively good results on the competition's official leaderboard.

1 Introduction

Text classification is an area of natural language processing (NLP) that aims to classify text using certain features. Previous studies on text classification tasks used traditional machine learning methods, which require researchers to manually design features. Feature extraction methods such as term frequency–inverse document frequency (TF-IDF) (Hakim et al., 2014) and N-Gram (Cavnar et al., 1994) are used to extract features from original documents, and then the features are input into classifiers such as naive Bayes(Berrar, 2019), support vector machines (SVMs) (Hearst et al., 1998), and decision trees (Vens et al., 2008). Since the advent of deep learning, text classification tasks are achievable without manual extraction of text features. Researchers must simply pretreat the text and incorporate it into a deep learning model for training. For text classification using deep learning methods, the classification accuracy is often higher than that of traditional machine learning methods. With their continuous improvement, deep learning models, such as recurrent neural networks (RNNs)(Zaremba et al., 2014), multi-channel CNN-LSTM (Zhang et al., 2017),gate recurrent units (GRUs) (Rana, 2016), long short-term memory (LSTM) (Shi et al., 2015), bidirectional long short-term memory (Bi-LSTM) (Zhang et al., 2015), and attention-based Bi-LSTM (Zhang et al., 2018) networks, can be used to solve text classification problems. In recent years, bidirectional encoder representations from transformers (BERT) (Devlin et al., 2018), a new deep learning model, has achieved, or even surpassed, human performance in multiple tasks within the NLP domain, including text classification.

Task 4 of the SemEval-2022 consists of the following two subtasks.

- Subtask 1: identifying whether the sentence contains any kind of PCL.
- Subtask 2: identifying which types of PCL the sentence contains.

In this paper, we introduce a deep learning system for SemEval-2022 Task 4: Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022). We applied the pretrained BERT model as the base model. This task contains two subtasks: single-label classification and multi-label classification. To accomplish both subtasks, we used fine-tuning methods on the base model with an additional classification layer. Our contributions are as follows:

- For the sentiment analysis task, we used the pretrained BERT model as the base model.
- To obtain the classification results, we added a fully connected layer at the end of the base model.

The remainder of this paper is organized as follows. Section 2 provides an overview of our system for the two subtasks. Section 3 presents the specific details of our system. Section 4 discusses the results of the experiments, and finally, we draw our conclusions in Section 5.

2 Overview

This section presents an overview of our system and experiments, consisting of the following steps:

1. The data processing step, in which we use text processing tools to clean the text content, such as removing HTML tags in the text.

2. The model training step, in which we build, train, and evaluate the model.

3. The result generating step, in which we evaluate the model and predict the results on the test dataset.

Task description. The two subtasks involved text sentiment analysis and classification. The difference between them is that subtask 1 only requires us to determine whether the text contains any kind of PCL. Subtask 2 is the multi-label classification task, and the data of subtask 2 are marked by a list of 0s and 1s, which indicate the type of linguistic techniques (Unbalanced_power_relations, Shallow_solution, Presupposition, Authority_voice, Metaphors, Compassion, The_poorer_the_merrier) used to express condescension.

2.1 Data processing

To use the original text as much as possible and reduce the impact of meaningless text on the model, we built text cleaning tools that can be used to remove redundant text from the original. In addition, to complete the text classification task, a special token is added to the front of the original sentence.
Preprocessing. The texts may have been retrieved from the Internet by an automated program and inevitably there will be some unnatural language in the text. Text processing tools, such as regular expressions and Beautiful Soup, are used to remove impurities, such as HTML tags and redundant punctuation, from the text. Because the original sentence cannot be used in the pretrained BERT model, a special token **[CLS]** is added to the front of the sentence, and the model receives the new sequence (with the added token) as input.

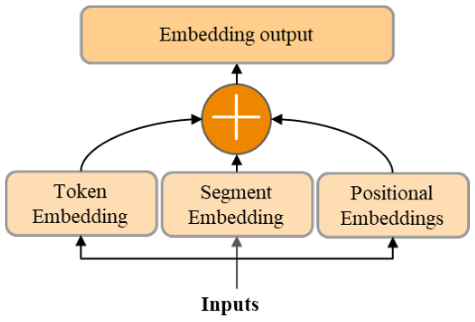


Figure 1: Embedding blocks

2.2 Deep learning models

In recent years, the use of deep learning for NLP text classification has become the most commonly adopted method in the industry. We used the pretrained BERT model to accomplish the tasks mentioned in the task description.
Bidirectional Encoder Representations from Transformers (BERT). As the name suggests, the BERT model is the encoder of the bidirectional transformer. BERT uses masked LM and next-sentence prediction to capture the representation at the word and sentence levels, respectively, and pretrains the model in a self-supervised manner.

Since the BERT model was proposed by Google in 2018, the entire field of NLP has entered a new stage. With BERT, we can easily fine-tune a pretrained model to achieve outstanding results that may even surpass human performance.

BERT consists of two main blocks: the embedding block and transformer encoder block, whose details are as follows.

1. **Embedding Block.** After preprocessing the original text, the output is fed to the embedding block, whose structure is shown in Figure 1.

   The embedding block has three embedding layers: the Token Embeddings, which convert each word into a fixed-dimensional vector similar to most deep learning models; Segment Embeddings, which distinguish between the two sentences; and Position Embeddings, which represent the position of each word in the sentence. These embedding layers transform the input text into a three-dimensional matrix $X \in R^{N \times n \times d}$, where N is the number of sentences in the text, n is the number of words in the sentence, and d is the dimension of the embedding vector.

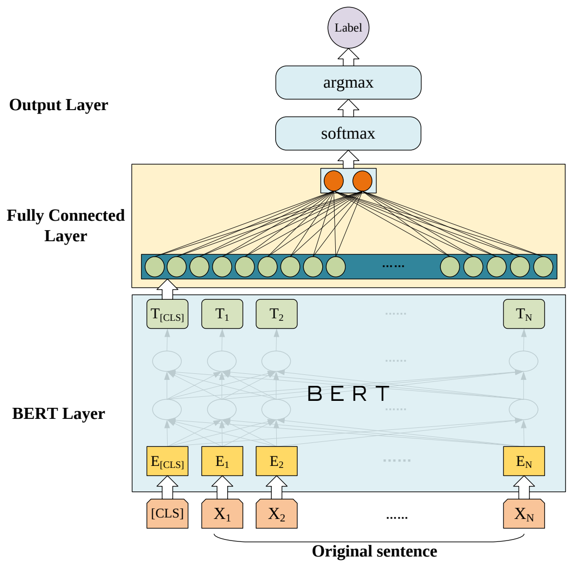


Figure 2: Single-label classification system

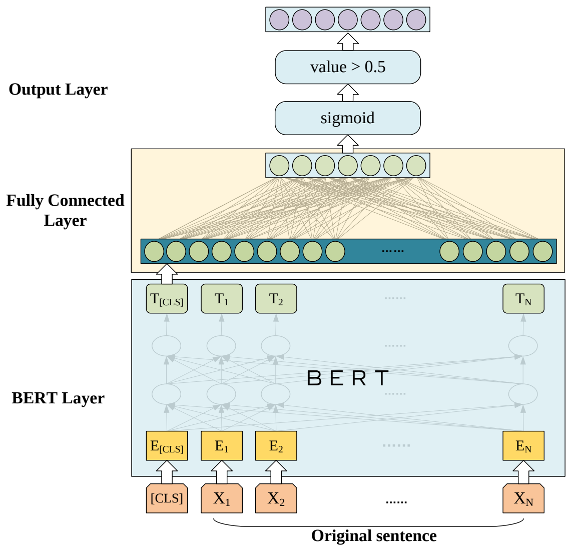


Figure 3: Multi-label classification system

2. **Encoder Block.** The encoder block comprises a series of transformer encoder blocks. Each transformer encoder block comprises two layers: the multi-head self-attention and feed-forward layers. The self-attention layer included in the encoder block of the transformer allows each word in the sentence to use the information of all other words in the sentence. The output of the current word does not need to depend on the output of the previous word, making the training well parallelized and greatly reducing the time to train the model. Because each word has a different impact on the sentence category, the attention mechanism can dynamically change the weight of each word.

3 Model Description

A pretrained BERT model is used to accomplish both subtasks with the two independent datasets. The details of the model built for these two subtasks are as follows.

3.1 Subtask 1: single-label classification

The architecture of the system built for subtask 1 has three different layers, as shown in Figure 2.

3.2 Subtask 2: multi-label classification

The system built for subtask 2 is similar to that for subtask 1, and the architecture of this system is only slightly different in the output layer. The structure is shown in Figure 3.

3.3 Details of the model architecture

BERT Layer. After preprocessing, the texts are input into the z BERT model, which contains the embedding and encoder blocks. Each word in the input sequence will output a fixed-dimensional (d) vector. In our BERT model (bert-based-uncased), d is 768.

Fully Connected Layer. The fully connected layer is used to convert a d-dimensional vector into a vector with the number of categories or labels as the dimension. In the text classification task, only the output of the first word, which is **[CLS]** at the BERT layer, is fed to the fully connected layer because it integrates the semantic information.

Output Layer. A matrix $X \in R^{N \times c}$ is output by the fully connected layer, in which N is the number of sentences and c is the number we manually set. In the single-label two-category classification task, it is set to 2, and the fully connected layer converts the 768-dimensional vector into a 2-dimensional vector. In the multi-label two-category classification task, it is set to the number of labels, 7, and the fully connected layer converts the 768-dimensional vector into a 7-dimensional vector.

To obtain the final result for the single-label classification task, the output of the fully connected layer is input into the *softmax* function to calculate the probability of the sentence belonging to the category, and the outcomes of the *softmax* function are fed to the *argmax* function to obtain the classification result.

$$class = \begin{cases} 0, value_0 \geq value_1, \\ 1, value_0 < value_1 \end{cases} \quad (1)$$

If the output value is 1, the sentence belongs to the label, that is, this sentence contains some kind of PCL; otherwise, the sentence does not contain any kind of PCL.

For the multi-label classification task, we input the result of the fully connected layer into a *sigmoid* function that maps each value in the output vector to a value between 0 and 1. Each value in the vector is then mapped to 0 or 1 according to the rounding rules.

$$label_i = \begin{cases} 0, label_i \leq 0.5, \\ 1, label_i > 0.5. \end{cases} \quad (2)$$

The output is a 7-dimensional vector that consists of 0 or 1. If the value is 1, the sentence used the technique corresponding to the vector element number to express the condescension.

3.4 Training and Hyperparameters

For these two classification tasks, we used the BCEwithLogits loss function and Adam (Kingma and Ba, 2017) optimizer to train both models. Both models use a stochastic gradient with mini-batches of size 16. The hyperparameters are as follows:

Hyperparameters The maximum input sequence length of the BERT model is 512, the dimension of word embeddings (d) is 768, the dropout ratio is 0.1 at each layer in the models, the learning rate is 1e-5, and the number of epochs is 15.

4 Experiment

Dataset. For the two subtasks, the corpus we used to train the model are from the competition(Pérez-Almendros et al., 2020), without other external data.

dontpatronizeme_pcl.tsv This dataset contains 10,469 paragraphs, and each paragraph is annotated with a label ranging from 0 to 4. In the single-label classification subtask, the original label annotated as either 0 or 1 is replaced with 0, and the other labels with 1.

dontpatronizeme_categories.tsv This dataset contains 993 unique paragraphs with a total of 2,760 instances of PCL. In the multi-label classification task, each paragraph is annotated with 7 labels ranging from 0 to 1.

Table 1: Subtask 1 result

Precision	Recall	F1_Score
0.5097	0.4132	0.4564

Table 2: Subtask 2 result

Label	Score
Unbalanced_Power_Relations	0.1600
Shallow_Solution	0.1245
Presupposition	0.0721
Authority_Voice	0.0968
Metaphor	0.0696
Compassion	0.1139
The_poorer_the_merrier	0.0385
Average	0.0965

Evaluation Methods. For subtask 1 (single-label classification), the competition metrics given by the competition organizer are precision, recall, and F1 score. For subtask 2 (multi-label classification), there are two competition metrics: prediction accuracy of each label and average prediction accuracy of all labels.

Results. The results of the two subtasks are shown in Tables 1 and 2.

For subtask 1, we ranked 42/81 in precision, 47/81 in recall, and 52/81 in F1 score.

For subtask 2, we ranked 35, 33, 35, 34, 33, 34, and 24 out of 81 for the seven labels: Unbalanced_power_relations, Shallow_solution, Presupposition, Authority_voice, Metaphors, Compassion, and The_poorer_the_merrier, respectively.

Experiments and Analysis. We used 80% of the training data as the training set and 20% of the training data as the validation set. We trained our model on the training set and used the validation set to evaluate the accuracy of the model. Our system achieved relatively good results on the competition's official leaderboard, which is inseparable from the excellence of the pretrained BERT model. The outstanding advantage of the pretrained model is that it can learn the language from a large amount of unlabeled data and then fine-tune on a small amount of labeled data. Thus, downstream tasks often lead to better learning of language and task-specific features.. Compared to traditional RNN and LSTM models, BERT can perform concurrently and simultaneously extract relational features of words in a sentence at several different levels, thus comprehensively reflecting the sentence semantics. Compared to word2vec, the meanings of words can also be obtained according to the context of the sentence, which would avoid ambiguity.

5 Conclusion

In this paper, we described our system, which is based on the pretrained BERT model, for the text classification task SemEval 2022 Task 4: Patronizing and Condescending Language Detection. We added a classification layer to the pretrained BERT model to address both subtasks. The results generated by the proposed system achieved a relatively good ranking. In the future, we hope to explore other models and methods in the sentiment analysis field.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61966038. The authors would like to thank the anonymous reviewers for their constructive comments.

References

D. Berrar. 2019. Bayes' theorem and naive bayes classifier - sciencedirect. *Encyclopedia of Bioinformatics and Computational Biology*, 1:403–412.

William Cavnar, , William B. Cavnar, and John M. Trenkle. 1994. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova Google, and A I Language. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ari Aulia Hakim, Alva Erwin, Kho I Eng, Maulahikmah Galinium, and Wahyu Muliady. 2014. Automated document classification for news article in bahasa indonesia based on term frequency inverse document frequency (tf-idf) approach. In *Proceedings of 6th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 1–4.

M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Rajib Rana. 2016. Gated recurrent unit (gru) for emotion classification from noisy speech. *arXiv preprint arXiv:1612.07778*.

Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, Wang-Chun Woo, and Hong Kong Observatory. 2015. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, volume 28, pages 802–810. Curran Associates, Inc.

Celine Vens, Jan Struyf, Leander Schietgat, Sašo Džeroski, Hendrik Blockeel, C Vens, J Struyf, L Schietgat, H Blockeel, and S Džeroski. 2008. Decision trees for hierarchical multi-label classification. *Machine Learning 2008 73:2*, 73:185–214.

Wojciech Zaremba, Ilya Sutskever, Oriol Vinyals, and Google Brain. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Haowei Zhang, Jin Wang, Jixian Zhang, and Xuejie Zhang. 2017. YNU-HPCC at SemEval 2017 task 4: Using a multi-channel CNN-LSTM model for sentiment classification. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 796–801, Vancouver, Canada. Association for Computational Linguistics.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 73–78, Shanghai, China.

You Zhang, Jin Wang, and Xuejie Zhang. 2018. YNU-HPCC at SemEval-2018 task 1: BiLSTM with attention based sentiment analysis for affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 273–278, New Orleans, Louisiana. Association for Computational Linguistics.

I2C at SemEval-2022 Task 4: Patronizing and Condescending Language Detection using Deep Learning Techniques

Laura Vázquez Ramos
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
laura.vazquez005@alu.uhu.es

Adrián Moreno Monterde
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
adrian.moreno521@alu.uhu.es

Victoria Pachón Álvarez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
vpachon@uhu.es

Jacinto Mata Vázquez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
mata@uhu.es

Abstract

Patronizing and Condescending Language is an ever-present problem in our day-to-day lives. There has been a rise in patronizing language on social media platforms manifesting itself in various forms. This paper presents two performing deep learning algorithms and results for the "Task 4: Patronizing and Condescending Language Detection." of SemEval 2022. The task incorporates an English dataset containing sentences from social media from around the world. The paper focuses on data augmentation to boost results on various deep learning methods as BERT and LSTM Neural Network.

1 Introduction

Nowadays, Patronizing and Condescending Language (PCL) (Pérez-Almendros, Espinosa-Anke, and Schockaert 2022) is used to refer to a forced kindness that derives from a perceived superiority towards another person. A subtle form of bullying, being patronized can leave a person feeling infuriated and impotent.

It is a type of behaviour that cuts across generations. An older person can talk down to a younger colleague, but it can just as easily happen the other way around. Men can patronize women at work and vice versa. But what they have in common is power play, with one individual exerting their authority or seniority over another.

People could feel discriminated by condescending comments. Considering the relevance of equality and respect, and it is important to note that nobody should be treated in such a way as to feel intimidated or different. In SemEval-2022: Task 4 Subtask 1, participants must determine whether a phrase presents PCL or not.

The idea behind the proposed solution was to compare three models based on deep learning. The first model is a Long short-term memory (LSTM) neural network (Zhou et al. 2015). The second, a LSTM neural network with embedding pretrained layer. The third one uses BERT (Devlin et al. 2018). The latter yielded the best results. For all the models we have used data augmentation to balance the training dataset.

The F1-score in our final submission (BERT) was 0.4134 on the test dataset. Compared to the results of the winning team, the difference is not extremely large. Nevertheless, our model obtained a good result of accuracy (0.61) compared to other participants.

2 Background

This paper is focused on Subtask 1: Binary classification. The corpus provided to perform subtask 1 (Pérez-Almendros, Espinosa-Anke, and Schockaert 2020) was composed of 10473 documents with 6 features: "id", "doc id", "keyword", "country code", "text" and "label". The dataset was imbalanced with respect to the class "label". Out of 10473 rows, 9470 were from the negative class and 993 from the positive class. English data augmentation was applied to the original dataset to balance them (Shorten, Khoshgoftaar, and Furht 2021).

After being balanced with data augmentation, the rows of the dataset increased to 19401, 9926 (class 1) and 9470 (class 0). In order to train the three proposed models, we only used the "text" and "label" features. The csv file used follows this structure:

- text: "*The pope as well called on the congregation to reach out…*"
- label: *"1"*

3 Related work

In recent years, several people have been researching this topic. A few years ago, an experiment of the impact of age, race, and stereotypes on perceptions of language performance and patronizing speech was published (Atkinson and Sloan 2017). Indeed, a research of different types of behaviours in healthcare settings (as condescending language) was published to show the impact it has in the world (Felblinger 2009). This is a recent problem and there are significant challenges to be overcome.

4 System overview

4.1 Balancing data techniques

Imbalanced data refers to types of datasets where the target class has an uneven distribution of observations. Sometimes, when the records of a certain class outnumber the other class, our classifier may become biased towards the prediction.

Before considering whether to use balancing techniques, we analyzed the data provided and we trained the models with the original dataset to test the results.

For this purpose, we trained both LSTM neural network and BERT models. The results were as expected. Both models reached similar results. To summarize, they obtained an accuracy of 0.91 and a ROC curve of 0.66. Given the obtained results, we decided to apply some balancing techniques to the original dataset.

Random Under-sampling

Random Under-sampling (Prusa et al. 2015) is a technique to remove examples from the majority class. However, this approach can result in the loss of valuable information during model training. The original dataset was extremely imbalanced, so rows of negative class were removed to achieve a balanced dataset.

The new dataset created using this technique totaled 1986 rows, 993 for each class.

Data Augmentation

Due to the results obtained with the under-sampling method, data augmentation was used to balance the data. Among the most common data augmentation techniques, synonym substitution was used. The synonym augmenter (Wordnet, English) (Miller 1995) to create synonym phrases for the minority class was applied. An example of this kind of substitution is:

- Sentence: *"A quick fox jumps over the lazy dog"*
- Synonym sentence: *"A prompt dodger jumps over the lazy domestic dog"*

Finally, the balanced dataset had a distribution of 9926 (class 1) and 9470 (class 0).

Table 1 shows the results obtained after the application of the two balancing methods. As can be seen, data augmentation produced better results.

Model	**Under sampling**		**Data Augmentation**	
	Accuracy	**ROC Curve**	**Accuracy**	**ROC Curve**
LSTM Neural Network	0.73	0.73	**0.97**	**0.97**
LSTM Neural Network with embedding pretrained layer	0.70	0.70	0.96	0.96
BERT-base-uncased	0.81	0.81	**0.97**	**0.97**

Table 1: Results obtained using sampling techniques for balancing the dataset

4.2 Models

Based on LSTM Neural Networks and BERT, three different models were implemented.

LSTM Simple Neural Network

LSTM is a special type of recurrent neural network. The main feature of recurrent networks is that information can persist by introducing loops in the network diagram, so they can basically 'remember' previous states and use this information to decide what will be next.

The LSTM was composed of an embedding layer with a size of 200, two bidirectional LSTM layers, a dense layer, a drop layer and, finally, a dense layer with a sigmoid activation.

The parameters used to train the LSTM Neural Network were a batch size of 32 and 10 epochs. Indeed, early stopping was invoked to avoid over-training.

LSTM Simple Neural Network with pretrained embedding layer

A pretrained layer was added to the model described above using GloVe [1] (Pennington, Socher, and Manning 2014, 1532-1543). GloVe is a type of implementation of an inter-contextual model, so that each word that appears in training will have a single vector representation obtained by collapsing all the information available about this word with all its appearances in the data.

Some pretrained word vectors of different sizes were downloaded. Finally, we used a file with a size of 200d. Then we added a weight matrix to the first layer of the recurrent neural network.

BERT: Bidirectional Encoder Representations from Transformers

A BERT-based transformer was used to train our third model. In particular, the model implemented was "BERT-base-uncased", which consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.

A transformer (Vaswani et al. 2017), has an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary.

The model was fine-tuned with the balancing dataset. Before training, every word was set to lower case. The model was trained with a batch size of 32 and 5 epochs.

5 Experimental setup

To set up our models, some libraries were used. Some of them were "NLTK" (Wang and Hu 2021, 1041-1049), "Keras" [2], "TensorFlow" (Joseph, Nonsiri, and Monsakul 2021, 85-111), "Scikit-learn" (Hao and Ho 2019) and "Pandas" (Stepanek 2020).

To test whether the data was balanced, the original training dataset was used. Once the balancing method was decided, the training data was split into two parts – 80% for training and 20% for test, using a stratify approach.

The stratify parameter makes a split so that the proportion of values in the sample produced will be the same as the proportion of values provided for the parameter to stratify.

During the training phase, we evaluated our models with accuracy, ROC curve, precision, recall and F1-score measures.

Once the organizers provided the test data, the models with the original training dataset were trained without splitting.

6 Results

The two models submitted were LSTM Neural Network with pretrained embedding model and BERT-base-uncased. According to the official metrics (F1-score for the positive class), a result of 0.413 and 0.61 of accuracy was obtained. BERT-base-uncased reached the best results.

After training using under sampling and data augmentation methods, we concluded that data augmentation had the best results.

Table 2 shows a summary of the results obtained during the evaluation phase using data augmentation.

7 Conclusions

In this paper we present our approach and system description on Task 4 (Subtask 1) in SemEval 2022: Patronizing and Condescending Language

[1] https://nlp.stanford.edu/projects/glove/

[2] https://keras.io/

Model	Training			Test		
	Accuracy	ROC Curve	F1-score (class 1)	Accuracy	ROC Curve	F1-score (class 1)
LSTM Neural Network	0.97	0.97	0.98	0.95	0.95	0.94
LSTM Neural Network with embedding pretrained layer	0.96	0.96	0.97	0.96	0.96	0.97
BERT-base-uncased	0.96	0.96	0.96	0.96	0.96	0.95

Table 2: Results obtained during the evaluation phase using data augmentation

Detection towards communities in the media. The main aim was to develop three deep learning models using data augmentation to solve imbalanced problem of the original dataset. We implemented three different models. After training and analyzing each model, an F1-score of 0.41 in the evaluation process for class “1” was achieved. For future works, we think the models could be further improved by training with a bigger dataset or using more balancing techniques.

References

Atkinson, Jaye L. and Robin G. Sloan. 2017. *"Exploring the Impact of Age, Race, and Stereotypes on Perceptions of Language Performance and Patronizing Speech." Journal of Language and Social Psychology 36* (3): 287-305. doi:10.1177/0261927X16662967. https://doi.org/10.1177/0261927X16662967.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *"BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding."* CoRR abs/1810.04805.

Felblinger, Dianne M. 2009. *"Bullying, Incivility, and Disruptive Behaviors in the Healthcare Setting: Identification, Impact, and Intervention." Frontiers of Health Services Management 25* (4): 13-23. https://www.proquest.com/scholarly-journals/bullying-incivility-disruptive-behaviors/docview/203882663/se-2?accountid=14549.

Hao, Jiangang and Tin Kam Ho. 2019. *Machine Learning made Easy: A Review of Scikit-Learn Package in Python Programming Language. Vol. 44.* Los Angeles, CA: SAGE Publications. doi:10.3102/1076998619832248. https://journals.sagepub.com/doi/full/10.3102/1076998619832248.

Joseph, Ferdin Joe John, Sarayut Nonsiri, and Annop Monsakul. 2021. *"Keras and TensorFlow: A Hands-on Experience." In Advanced Deep Learning for Engineers and Scientists: A Practical Approach,* edited by Kolla Bhanu Prakash, Ramani Kannan, Alex, S. Albert er and G. R. Kanagachidambaresan, 85-111. Cham: Springer International Publishing. doi:10.1007/978-3-030-66519-7_4". https://doi.org/10.1007/978-3-030-66519-7_4.

Miller, George A. 1995. *"WordNet: A Lexical Database for English." Commun.ACM 38* (11): 39–41. doi:10.1145/219717.219748. https://doi.org/10.1145/219717.219748.

Perez-Almendros, Carla, Luis Espinosa-Anke, and Steven Schockaert. 2020. *"Don't Patronize Me! an Annotated Dataset with Patronizing and Condescending Language Towards Vulnerable Communities.".*

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. *"GloVe: Global Vectors for Word Representation."Association for Computational Linguistics, oct. doi:10.3115/v1/D14-1162".* https://aclanthology.org/D14-1162.

Pérez-Almendros, Carla, Luis Espinosa-Anke, and Steven Schockaert. 2022. *"SemEval-2022 Task 4: Patronizing and Condescending Language Detection."Association for Computational Linguistics .*

Prusa, Joseph, Taghi M. Khoshgoftaar, David J. Dittman, and Amri Napolitano. 2015. *"Using Random Undersampling to Alleviate Class Imbalance on Tweet Sentiment Data.".* doi:10.1109/IRI.2015.39.

Stepanek, Hannah. 2020. *Thinking in Pandas: How to use the Python Data Analysis Library the Right Way*. Berkeley, CA: Apress. doi:10.1007/978-1-4842-5839-2. https://library.biblioboard.com/viewer/087e187a-b660-11ea-a44d-0a7fc7c4e64f.

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

Wang, Meng and Fanghui Hu. 2021. *"The Application of NLTK Library for Python Natural Language Processing in Corpus Research." Theory and Practice in Language Studies* 11 (9): 1041-1049. doi:10.17507/tpls.1109.09.

Zhou, Chunting, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. 2015. *A C-LSTM Neural Network for Text Classification.*

PiCkLe at SemEval-2022 Task 4: Boosting Pre-trained Language Models with Task Specific Metadata and Cost Sensitive Learning

Manan Suri
Netaji Subhas University of Technology
New Delhi, India
manan.suri.ug20@nsut.ac.in

Abstract

This paper describes our system for Task 4 of SemEval 2022: Patronizing and Condescending Language Detection. Patronizing and Condescending Language (PCL) refers to language used with respect to vulnerable communities that portrays them pitifully and is reflective of a sense of superiority. Task 4 involved binary classification (Subtask 1) and multi-label classification (Subtask 2) of Patronizing and Condescending Language (PCL). For our system, we experimented with fine-tuning different transformer-based pre-trained models including BERT, DistilBERT, RoBERTa and ALBERT. Further, we have used token separated metadata to improve our model by helping it contextualize different communities with respect to PCL. We faced the challenge of class imbalance, which we solved by experimenting with different class weighting schemes. Our models were effective in both subtasks, with the best performance coming out of models with Effective Number of Samples (ENS) class weighting and token separated metadata in both subtasks. For subtask 1 and subtask 2, our best models were finetuned BERT and RoBERTa models respectively.

1 Introduction

Patronizing and Condescending Language (PCL) in the context of vulnerable communities refers to language that portrays a sense of superiority or has a tendency to view communities with a pitiful and passionate lens. PCL works subtly and is intricately associated with the way that words and phrases are used. This makes it difficult to classify PCL as compared to overtly offensive language where the nature of words and phrases is clearly negative. Since the harms associated with PCL are not always evident, it is often used inadvertently by actors trying to help these communities. Recognising PCL is important because the expression of PCL leads to a paradigm where discrimination and injustices are routinised and made less visible (Ng, 2007). Use of PCL also feeds into stereotypes (Fiske, 1993), reinforces societal power dynamics and avoids deep-rooted societal problems, providing surface-level solutions for the same (Chouliaraki, 2010). Task 4 of SemEval 2022 (Pérez-Almendros et al., 2022) aims to identify PCL with Subtask 1 working towards binary classification and Subtask 2 working towards the multi-label classification of PCL.

Our strategy involves using state-of-the-art Pre-Trained Language Models (PLMs) and finetuning them for our specific task. We work with BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), DistilBERT (Sanh et al., 2019), ALBERT (Lan et al., 2019) as our models. Additionally, we design effective and simple approaches to optimize our models, by experimenting with different cost-sensitive class weighting methods and working with token separated metadata to enhance performance. With an increasing number of PLMs, each having millions of parameters and being computationally expensive to train, it is essential to make the right model choice. This paper provides a comprehensive analysis of the performance of different models. This can help determine baselines for similar text classification tasks. For model replicability, our code is available online.[1]

2 Background

2.1 Task Description

Patronizing and Condescending Language (PCL) refers to language which may seem kind or helpful but is reflective of a sense of superiority. SemEval 2020 Task 4: Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022) had two subtasks that dealt with the identification of PCL and the categories used to express it. These were seen specifically in reference to communities

[1] https://github.com/MananSuri27/PatronisingAndCondescendingLanguage

	PCL	Non PCL	unb	sha	pre	aut	met	com	the
Training Set	794	7581	574	160	162	192	363	145	29
Development Set	199	1895	142	36	62	38	106	52	11

Table 1: Distribution of categories across the training and development set. The labels 'unb', 'sha', 'pre', 'aut', 'met', 'com' and 'the' refer to 'unbalanced power relations', 'shallow solution', 'presupposition', 'authority voice', 'metaphor', 'compassion', and 'the-poorer-the-merrier' respectively.

being identified as being vulnerable and having unfair treatment in the media.

- **Subtask 1:** Subtask 1 was binary classification, where given a paragraph the system was supposed to predict whether it contains any form of PCL. The basis of evaluation was F1 score on the positive class, PCL.
- **Subtask 2:** Subtask 2 was a multi-label classification task where given a paragraph, we were supposed to predict what categories of PCL the paragraph subscribes to. Pérez-Almendros et al. (2020) determined these categories based on previous works and their research on PCL. The 7 categories considered are 'unbalanced power relations' (unb), 'shallow solution' (sha), 'presupposition' (pre), 'authority voice' (auth), 'metaphor' (met), 'compassion' (com), and 'the-poorer-the-merrier' (the). The basis of evaluation was average F1 score across the given classes.

2.2 Data Description

This task is based on the Don't Patronize Me!(Pérez-Almendros et al., 2020) dataset by the task organizers. For this paper, we have considered the practice split offered by the organizers as the split between train and development set. The train, development and test set contain 8375, 2094 and 3831 rows of data respectively.

The paragraphs in this dataset have been selected from news stories and have been annotated with labels specifying whether they qualify as PCL and the categories of PCL that they belong to. The dataset includes additional information about the paragraphs— including the country of reference and the keyword. The keywords clarify the context of the paragraph. The included keywords are: 'disabled' (dis), 'homeless' (hom), 'hopeless' (hop), 'immigrant' (imm), 'in-need' (need), 'migrant' (mig), 'poor families' (poor), 'refugees' (ref), 'vulnerable' (vul) and 'women' (wom). The dataset includes reports from 20 countries.

Table 1 shows the distribution of labels in the train and dev set. We can observe that the distribution of classes is heavily imbalanced. In the training set, only 9.5% of the samples belong to the PCL class. Similarly, in the multi-label category, 72% of all samples with PCL have the class label of 'unb' for the training set.

3 System Overview

3.1 Finetuning Pre-trained Language Models (PLMs)

Pre-training in NLP is a technique that involves training general-purpose language representations through a large set of unannotated text data. It is beneficial for downstream tasks and avoids training a new model from scratch. Pre-training leads to a better generalization performance and helps in convergence on downstream tasks because it provides a better model initialisation. Most NLP datasets contain limited human-annotated samples, due to which there is a tendency to cause overfitting. Pre-training can be regarded as a kind of regularization, preventing overfitting on smaller datasets (Erhan et al., 2010). Pre-training models followed by fine-tuning them for downstream tasks has shown good performance on many NLP tasks (Dai and Le, 2015; Radford and Narasimhan, 2018; Peters et al., 2018).

Briefly discussing the PLMs we have used for this task:

BERT: BERT refers to Bidirectional Encoder Representations (Devlin et al., 2019). It uses bidirectional transformers (Vaswani et al., 2017) pre-trained using a combination of Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). It learns deep bidirectional representations by jointly conditioning on both left and right context layers.

RoBERTa: RoBERTa refers to A Robustly Optimized BERT Pretraining Approach (Liu et al., 2019). It builds on BERT and modifies key hyperparameters, such as training with larger mini-batches and learning rates. RoBERTa also removes

BERT			
Class Weighting	**precision**	**recall**	**F1**
None	0.672	0.402	0.503
INS	0.456	0.698	0.552
ISNS	0.515	0.598	0.553
ENS	0.541	0.658	0.594

RoBERTa			
Class Weighting	**precision**	**recall**	**F1**
None	0.667	0.462	0.546
INS	0.370	0.779	0.502
ISNS	0.510	0.648	0.571
ENS	0.455	0.678	0.544

DistilBERT			
Class Weighting	**precision**	**recall**	**F1**
None	0.703	0.392	0.503
INS	0.476	0.492	0.484
ISNS	0.564	0.508	0.542
ENS	0.494	0.593	0.539

ALBERT			
Class Weighting	**precision**	**recall**	**F1**
None	0.513	0.296	0.376
INS	0.213	0.764	0.333
ISNS	0.377	0.638	0.474
ENS	0.389	0.739	0.510

Table 2: Subtask1: Binary Classification; The performance of the PLMs we have considered: BERT, RoBERTa, DistilBERT and ALBERT on the dev set, with different class weighting techniques.These systems also included token separated metadata. The class weighting strategies (INS- Inverse Number of Samples, ISNS- Inverse of Square Root of Number of Samples, ENS- Effective Number of Samples) are discussed in section 3.2 .

BERT								
Class Weighting	**unb**	**sha**	**pre**	**aut**	**met**	**com**	**the**	**avg**
None	0.339	0.070	0.270	0.190	0.191	0.314	0.000	0.196
INS	0.409	0.190	0.278	0.218	0.245	0.386	0.098	0.261
ISNS	0.440	0.148	0.279	0.156	0.182	0.384	0.098	0.241
ENS	0.427	0.197	0.277	0.223	0.256	0.396	0.129	0.272

RoBERTa								
Class Weighting	**unb**	**sha**	**pre**	**aut**	**met**	**com**	**the**	**avg**
None	0.838	0.287	0.209	0.085	0.029	0.642	0.000	0.299
INS	0.838	0.324	0.358	0.370	0.328	0.642	0.062	0.417
ISNS	0.834	0.310	0.315	0.317	0.312	0.642	0.109	0.405
ENS	0.838	0.313	0.367	0.379	0.323	0.642	0.079	0.420

DistilBERT								
Class Weighting	**unb**	**sha**	**pre**	**aut**	**met**	**com**	**the**	**avg**
None	0.451	0.092	0.162	0.137	0.121	0.335	0.000	0.185
INS	0.484	0.157	0.255	0.205	0.216	0.394	0.091	0.257
ISNS	0.377	0.147	0.244	0.168	0.184	0.306	0.043	0.210
ENS	0.480	0.217	0.288	0.205	0.212	0.400	0.080	0.269

ALBERT								
Class Weighting	**unb**	**sha**	**pre**	**aut**	**met**	**com**	**the**	**avg**
None	0.826	0.000	0.009	0.000	0.010	0.490	0.000	0.191
INS	0.828	0.100	0.258	0.088	0.079	0.449	0.000	0.257
ISNS	0.548	0.177	0.223	0.147	0.083	0.456	0.000	0.233
ENS	0.436	0.217	0.294	0.238	0.228	0.418	0.000	0.262

Table 3: Subtask2: Category Classification; The performance of the PLMs we have considered: BERT, RoBERTa, DistilBERT and ALBERT on the dev set, with different class weighting techniques.These systems also included token separated metadata. The columns represent the F1 score for different classes and the average F1 score across all classes. The class weighting strategies (INS- Inverse Number of Samples, ISNS- Inverse of Square Root of Number of Samples, ENS- Effective Number of Samples) are discussed in section 3.2 .

the Next Sentence Training (NSP) objective in BERT.

DistilBERT: DistilBERT (Sanh et al., 2019) is a small, fast, cheap and light transformer based model trained by distilling BERT base. It has lesser parameters, and runs faster but still conserves a large proportion of BERT's performance. DistilBERT uses a triple loss combining language model, distillation and cosine distance losses to leverage the advantage gained by larger models during pre-training.

ALBERT: A Lite BERT for Self-supervised Learning of Language Representations (Lan et al., 2019) is a modification of BERT which efficiently allocates model capacity, reducing the training time and memory consumption. It reduces parameters by factorised embedding parameterization- where the embedding matrix is decomposed to a lower dimension and projected. Layer sharing across layers reduced redundancy. Inter-sentence coherence loss based on Sentence Order Prediction (SOP) is also employed by the model.

We finetuned the PLMs for both subtasks by stacking a dropout layer followed by a dense layer on top of the PLM model. The dropout layer before the dense classification layer is added for regularization. In Subtask 1 we use Sigmoid activation to predict binary labels. In Subtask 2 we use sigmoid activation in the final layer rather than softmax as it allows us to deal with non-exclusive labels. For BERT, DistilBERT and ALBERT we use the features of the [CLS] token and for RoBERTa the <s> token. The performance of the models along with the other strategies we have used is present in Table 2 and Table 3 for Subtask 1 and Subtask 2 respectively.

3.2 Utilising metadata

We have attempted to enrich the PLMs with additional metadata provided in context to the paragraphs in the task. In this setup, more data is available to the model. This is based on the idea that more meaningful data leads to better performance on classification. The same has been observed by other researchers who have experimented with including task-specific data in NLP (Ostendorff et al., 2019; Zhang et al., 2019).

Pérez-Almendros et al. (2020) included ten keywords related to potentially vulnerable communities that are widely covered in media and have had the propensity of receiving condescending treatment, namely: disabled, homeless, hopeless, immigrant, in need, migrant, poor families, refugee, vulnerable and women. Since PCL involves a subtle use of language, we believe that contextualizing whether a phrase is PCL also depends on the context of which community or situation is being referred to.

496 @@26214070 refugee hk 3
Hundreds of thousands of Rohingya refugees living in sprawling camps in Bangladesh are celebrating the Muslim holiday of Eid al-Adha, praying for better lives as they wonder if they'll ever again celebrate at their homes in Myanmar. People streamed into makeshift mosques in the camps, the children dressed in new clothing . Those who could afford it feasted on buffalo meat. Muslims often...

350 @@21894186 homeless lk 4
It can not be right to allow homes to sit empty while many struggle to find somewhere to live, others having to sleep rough on pavements during Christmas, hoping against hope, for some charity to provide shelter . The number left homeless and destitute is alarming not necessarily at Christmas ?

Table 4: Two examples from the dataset to get an intuitive sense of the advantage that using keyword might add to contextualise the paragraph. Both paragraphs describe *home*, but one in the context of *refugees* and the other in context of *homelessness*. The data included in the first row of each paragraph includes the serial number, paragraph ID, keyword, country and annotation.

To understand this, let us consider two examples (Table 4) from the dataset, where paragraphs with ID @@26214070 and @@21894186 are tagged with the keywords "refugee" and "homeless" respectively, and use the word "home" in different contexts. With the "refugee" tag, we are given to understand that "home" refers in a very specific way to a place in the actor's country of origin whereas in the "homeless" context, "home" refers to accommodation or the lack there-of. The purpose of including additional metadata thus was to add to the contextualizing abilities of the model.

We include the metadata by adding it to the input string itself as another sentence in itself; separating the metadata from the text with the [SEP] token in case of BERT, DistilBERT and ALBERT and the </s> token for RoBERTA.

Therefore, the input in case of BERT, DistilBERT and ALBERT looks like:

[CLS] keyword [SEP] paragraph [SEP],

and in the case of RoBERTa, looks like:

<s> keyword </s> paragraph </s>.

We make this system design choice based on the following ideas:

- The chosen PLMs are strong enough to learn how the metadata interacts with the input sequence, considering that we have enough training samples available.
- Using token separated metadata rather than concatenating another model reduces the number of additional parameters to be trained.
- Using the [SEP] and </s> tokens help us utilize the power of pre-training which wouldn't have been convenient if we defined new special tokens to separate the metadata instead of using the predefined special tokens.

Subtask 1

Model	**F1 with**	**F1 without**
BERT	0.595	0.556
RoBERTa	0.571	0.566
DistilBERT	0.534	0.510
ALBERT	0.474	0.450

Subtask 2

Model	**Avg F1 with**	**Avg F1 without**
BERT	0.272	0.229
RoBERTa	0.420	0.387
DistilBERT	0.269	0.249
ALBERT	0.262	0.238

Table 5: The performance of the chosen models with and without use the of token separated metadata on the development set. For each model, the same parameters including class weights are used to ensure comparability.

Table 5 includes a comparison of the different models we have used, with and without the token separated metadata. For comparability, the same parameters including class weights are used for each model.

3.3 Cost Sensitive Learning

One of the challenges in the task was a heavy imbalance in the number of samples in the given classes in the training data. The positive class for the binary classification task (Subtask 1) was underrepresented where the number of samples with PCL was only around 9.5% of the training set. Similarly, subtask 2 which included multi-label classification had a large proportion of samples from only 2 classes- 'unb' and 'met' (72% and 45% of all samples with PCL respectively), and some classes such as 'the' were heavily underrepresented(3% of training samples with PCL). This varying distribution is a significant issue while training because it becomes a challenge for us to ensure that our model learns the characteristics of the minority classes as well.

Class imbalance is a common issue in many real-world datasets, and many techniques have been developed to mitigate this problem: changing the data (undersampling the majority class, oversampling the minority class, data augmentation by using synonyms or other such methods) or adjusting the model (like changing the performance metric). We found in our experiments that data manipulation techniques only provide a marginal performance boost, and the same has been observed by other researchers working on transformer-based models in text classification tasks (Tayyar Madabushi et al., 2019).

The technique that we have used is cost-sensitive learning (Elkan, 2001), which is based on the statistical concept of importance sampling. Importance sampling refers to weights being assigned to samples in a way that matches the given distribution of data. Mathematically, the loss function is modified to account for a multiplier that represents the weight of the class. This method doesn't modify the distribution of the imbalanced data directly.

For a single prediction x with a gold label of a given class, the loss function is then modified as:

$$loss(x, class) = weight[class]\Theta \qquad (1)$$

where Θ represents the original loss function.

This can be interpreted as adjusting the cost of misclassification of the given classes, practically increasing the cost for misclassification of an important class such as a minority class by assigning a larger weight to it.

We explored different cost weighting strategies which are discussed below. Table 6 describes the count as well as class weights of different cate-

	PCL	Non PCL	unb	sha	pre	aut	met	com	the
Count	794	7581	574	160	162	192	363	145	29
INS	10.55	1.10	3.65	13.09	12.93	10.91	5.77	14.44	72.21
ISNS	297.22	96.19	1.91	165.55	164.52	151.12	109.91	173.90	388.85
ENS	11.85	2.80	16.30	20.32	20.22	19.01	16.68	21.18	64.27

Table 6: Count of different categories in the training set and the calculated weights according to the Inverse of Number of Samples (INS), Inverse of Square-root of Number of Samples(ISNS) and Effective Number of Samples (ENS) schemes.

gories for subtask 1(PCL, No PCL) and subtask 2 (unb, sha, pre, aut, met, com, the) according to the different weighting schemes we discuss below.

1. Inverse of Number of Samples (INS)

For finding the class weight for a given class, we simply take the inverse of the number of samples in the class. It is a simplistic way of ensuring that under-represented classes have a higher weight and classes with a large number of samples have a lower weight. INS class weighting can be described by the following equation:

$$weight[class] \propto \frac{1}{n_{class}} \tag{2}$$

where n_{class} is the number of samples in that class.

2. Inverse of Square Root of Number of Samples (ISNS)

The INS method increases the recall by decreasing the number of false negatives, we observed that because the weight of the majority class had been highly diminished, the precision is still low because of a higher number of false positives. The ISNS method mathematically is the inverse of the root of class frequency. Mathematically, this means that the class weights are larger numeric quantities here than in the INS method and more importantly, the problem of the weights of the majority class being highly diminished for our dataset is mitigated by this method. The ISNS class weighting strategy can be summarised by the following equation:

$$weight[class] \propto \frac{1}{\sqrt{n_{class}}} \tag{3}$$

where n_{class} is the number of samples in that class.

3. Effective Number of Samples (ENS)

A third class weighting strategy that we consider is the Effective Number of Samples (ENS) method which was described by Cui et al. (2019). The authors argue that as the number of samples of a class increases, the benefit added by each new sample will diminish. Instead of considering individual rows of data as singular points in the space,

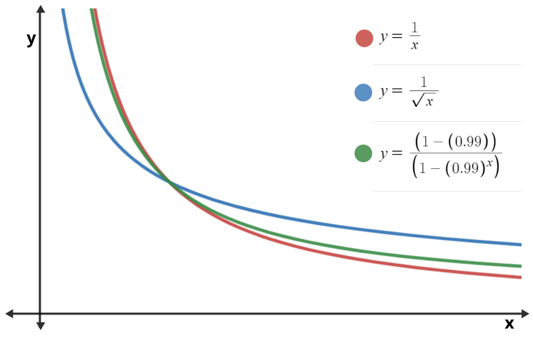


Figure 1: The plots of the mathematical functions that define INS, ISNS and ENS class weighting schemes. The actual class weight may involve additional constants.

this model considers them as small neighbouring regions, the effective number of samples is then calculated mathematically as the effective volume of samples, given by the simple formula:

$$ENS[class] = \frac{1-\beta^{n_{class}}}{1-\beta} \tag{4}$$

where β is a parameter that can take values as 0.9, 0.99, 0.999 and so on, ENS refers to the effective number of samples and n_{class} is the number of samples in the given class.

The weight of the class is then defined as being the inverse of the effective number of samples.

$$weight[class] \propto \frac{1-\beta}{1-\beta^{n_{class}}} \tag{5}$$

For a very high value of β , this class weight comes very close to the INS class weight.

Figure 1 is a graphical representation of the mathematical functions that define the class weights we have discussed above. The performance of the PLMs we have considered with different class weighting strategies can be seen in Table 2 and Table 3 (Subtask 1 and Subtask 2 respectively).

Subtask 1: Binary Classification				
RANK	TEAM NAME	PRECISION	RECALL	F1-SCORE
38	Team PiCkLe	0.46	0.5804	0.513

Subtask 2: Categorical Classification									
RANK	TEAM NAME	UNB	SHA	PRE	AUT	MET	COM	THE	F1 AVG
35	Team PiCkLe	0.1091	0.2254	0.1439	0.2101	0.1916	0.0651	0.1151	0.1515

Table 7: PiCkLe's Results on the official leaderboard for subtask 1 and subtask 2.

4 Experimental Setup

To ensure comparability, all models are trained on the same train, dev and test split. Further, the train-dev splits are the same splits provided by the task organizers in the practice splits.

The models were developed on Keras[2] (Chollet et al., 2015), and implemented using the transformers library by HuggingFace[3] (Wolf et al., 2019). We experimented with learning rates of 1e-5,2e-5 and 5e-5 for all models, finding the best results at 2e-5. For all the models, we fixed the max length parameter at 256 tokens and the batch size parameter to 16. The finetuning for the models was performed on Google Colab GPU. We trained each model for 1-2 epochs and found the best results at 2 epochs. The value of β for ENS class weighting was taken as 0.9997 for Subtask 1 and 0.99 for Subtask 2 based on experiments with different values.

Class weighting was implemented using the *class_weight* parameter during model fitting. We used the Autotokenizer offered by HuggingFace's transformers library to tokenize our inputs. We implemented the token separated metadata by setting the *add_special_tokens* parameter of the tokenizer as True and using the *text_pair* parameter. We used the Adam (Kingma and Ba, 2014) optimiser by Keras. The loss function used is binary cross-entropy and categorical cross-entropy for Subtask 1 and Subtask 2 respectively.

5 Results and Analysis

Based on the performance of different PLMs with different configurations (Table 2 and Table 3) on the development set, for Subtask 1 we submitted a finetuned BERT model with ENS class weighting and token separated metadata. For Subtask 2 we submitted a finetuned RoBERTa model with ENS class weighting and token separated metadata.

Our results in the given subtasks on the test set are shown in Table 7 for subtask 1 and subtask 2. We have ranked 38 on subtask 1 with an F1 score of 0.513. For subtask 1, our best performing model on the test set is BERT with token separated metadata and ENS class weighting. This model performs better than the baseline RoBERTa model and falls in the top half of all the models entered into the competition. For subtask 2, our submitted model for the evaluation phase was RoBERTa with token separated metadata and ENS class weighting. This model seems to have performed very well on the development set however it has failed to give the same performance on the evaluation set. While we saw an average F1 of 0.419 on the development set, we get a lower F1 of 0.1515 on the evaluation set. This model still ranks 35 on the leaderboard and has performed better than the baseline model. Moreover, this model is amongst the top 20 models in terms of the F1 scores on 'the' class which was in the smallest proportion in the training set, showing how the cost-sensitive learning that we have performed has been effective in taking into account the minority classes.

While we recognise that our model has performed well for subtask 1, the model still lacks in terms of learning what exactly represents PCL. Recognising PCL is an inherently difficult task because of the subtle nature of the language used and the lack of an exact benchmark of what constitutes PCL. With further tuning of parameters and attempts at improving paragraph representations, we may improve the performance of our existing model.

For Subtask 2, on analysis, we believe that a possible issue with the chosen model for submission is that its performance on the development set is heavily biased by the distribution of labels in the training and development set. This is evident by

[2]https://keras.io/
[3]https://huggingface.co/docs/transformers/index

the fact that 'unb' which is in the highest proportion in Subtask 2 has a development F1 score of 0.838 on the 'unb' class (which also raises the average F1), signifying that a large number of samples have been correctly classified as 'unb' in the development set. We believe that since this model has a high tendency to classify samples as 'unb', it gained a high F1 for 'unb' on the development set where this class was statistically well represented with 71% of the samples with PCL having the class 'unb' in the development set, however, the same distribution may not present in the evaluation set revealing a pitfall of our model.

6 Conclusion

The task of predicting Patronizing and Condescending Language (PCL) is relatively new in the field of Natural Language Processing and comes with its challenges as discussed in this paper. We used a finetuning approach to build models to identify and classify PCL and explored the performance of various models as well as training variations and present them as a comparative in this paper. We explore techniques to deal with class imbalance, which is a rampant problem in real-world datasets by considering various class weighting techniques which work based on cost-sensitive learning. We also explore the idea of using metadata to optimize our model by adding a context that represents the target community or situation being referred to in a given paragraph.

In the future, we would like to explore other options that utilise the power of task-specific metadata. We would also like to work with other transformer-based models such as T5 (Raffel et al., 2019) and ELECTRA (Clark et al., 2020). We would also like to work on improving the ability of our model to recognise the subtle use of language which is embodied by PCL.

Acknowledgements

We would like to thank Dr Vijay Kumar Bohat and Aniruddha Chauhan for their support during writing this paper. We would also like to thank the organisers of SemEval-2022 for conducting this competition.

References

Francois Chollet et al. 2015. Keras.

Lilie Chouliaraki. 2010. Post-humanitarianism: Humanitarian communication beyond a politics of pity. *International Journal of Cultural Studies*, 13(2):107–126.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. 2019. Class-balanced loss based on effective number of samples. *CoRR*, abs/1901.05555.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. 28.

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

Charles Elkan. 2001. The foundations of cost-sensitive learning. *Proceedings of the Seventeenth International Conference on Artificial Intelligence: 4-10 August 2001; Seattle*, 1.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, pages 625–660.

Susan Fiske. 1993. Controlling other people: The impact of power on stereotyping. *The American psychologist*, 48:621–8.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

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

Sik Hung Ng. 2007. Language-based discrimination: Blatant and subtle forms. *Journal of Language and Social Psychology*, 26(2):106–122.

Malte Ostendorff, Peter Bourgonje, Maria Berger, Julian Moreno-Schneider, Georg Rehm, and Bela Gipp. 2019. Enriching bert with knowledge graph embeddings for document classification.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pretraining.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle. 2019. Cost-sensitive BERT for generalisable sentence classification on imbalanced data. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 125–134, Hong Kong, China. Association for Computational Linguistics.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: enhanced language representation with informative entities. *CoRR*, abs/1905.07129.

ML_LTU at SemEval-2022 Task 4: T5 Towards Identifying Patronizing and Condescending Language

Tosin Adewumi, Lama Alkhaled, Hamam Mokayed, Foteini Liwicki
and Marcus Liwicki
Machine Learning Group
EISLAB, SRT
Luleå University of Technology
firstname.lastname@ltu.se

Abstract

This paper describes the system used by the Machine Learning Group of LTU in subtask 1 of the SemEval-2022 Task 4: Patronizing and Condescending Language (PCL) Detection. Our system consists of finetuning a pretrained Text-to-Text-Transfer Transformer (T5) and innovatively reducing its out-of-class predictions. The main contributions of this paper are 1) the description of the implementation details of the T5 model we used, 2) analysis of the successes & struggles of the model in this task, and 3) ablation studies beyond the official submission to ascertain the relative importance of data split. Our model achieves an F1 score of 0.5452 on the official test set.

Pérez-Almendros et al. (2020) introduced the dataset for the SemEval-2022 Task 4 (Pérez-Almendros et al., 2022)[1]. The dataset covers the English language. It is meant to support Natural Language Processing (NLP) models in identifying PCL towards vulnerable communities, such as poor families and refugees. The dataset is designed for 2 subtasks in the competition. Subtask 1 is a binary classification task of predicting the presence of PCL while subtask 2 is a multi-label classification task of predicting PCL categories. We address subtask 1 in this system paper.

PCL is an expression that depicts someone in a compassionate way or shows a superior attitude of the speaker (Pérez-Almendros et al., 2022). PCL identification is important because PCL has been shown to have harmful effects on vulnerable groups (Fox and Giles, 1996; Morris, 2007; Bell, 2013; Wang and Potts, 2019). This task of identifying and categorizing PCL is apparently more challenging than some other types of harmful language because it is subtle and generally used with good intentions (Wang and Potts, 2019; Gilda et al., 2022).

The main strategy of our system, to address the challenge, was to use a recent SoTA model (T5) in a simple, novel way to reduce out-of-class predictions. We discovered that our system achieves a relatively good performance on the task and PCL identification is a challenging task, due to its subtle nature. It achieved an F1 score of 0.5452 on the test set while the best score was 0.651. This made us rank 27 (66th percentile) out of 78 and we surpass the official RoBERTa baseline. We perform error analysis and ablation studies to evaluate the strengths and weaknesses of the model. We contribute the model checkpoint publicly on the HuggingFace hub[2] and the T5 code [3]

The rest of this paper is organized as follows. Section 1 gives a brief background of related work in PCL. Section 2 gives the system overview of what we used for the task. Section 3 describes the experimental setup for the task and the additional experiments beyond the official submission. Section 4 gives the tables of results and discusses relevant observations from the results. We share concluding remarks in section 5.

1 Background

Work on various sorts of harmful language in NLP has mostly concentrated on explicit aggressive and brazen phenomena (Pérez-Almendros et al., 2022). Scholars are striving to distinguish between harmful and unhealthy language by identifying the fundamental characteristics of unhealthy language. Price et al. (2020) proposed one of the most recent efforts in this regard. The research introduced a new dataset containing 44,000 comments with the unhealthy category sub-classified as either (1) hostile; (2) antagonistic, insulting, provocative or trolling; (3) dismissive; (4) condescending or patronizing; (5) sarcastic; and/or (6) an unfair generalisation. In their work, it is assumed that the language with a PCL tone will assume an attitude of superiority, implying that the other speak-

[1] semeval.github.io/SemEval2022/tasks

[2] huggingface.co/tosin/pcl_22

[3] github.com/tosingithub/pcl (after another competition)

ers/listeners are ignorant, naive, or unintelligent. In such scenarios, the language will usually imply that the other speaker should not be taken seriously.

Similarly, Morris (2007) explains the high likelihood of using PCL language when there is discussion between two persons with different mental health conditions. He demonstrated that patronizing language is common when a discussion occurred between a cashier with no cognitive issue and a customer who suffers from cognitive disability. Overall, PCL does not have an obvious negative or critical language and there is the challenge of limited, high-quality labelled data.

There have been different efforts at automatically detecting PCL. Wang and Potts (2019) showed that models with contextual representations are much better at identifying PCL and this bolstered the hypothesis that context is essential for PCL detection. They implemented the BERT model, which deploys a Transformer-based encoder architecture, on the TALKDOWN corpus they introduced. Both the base and large versions of the BERT model are implemented and evaluated over the new proposed corpus for balanced and imbalanced data. Price et al. (2020) added more context to their work by comparing the performance of BERT to human performance in order to better understand the model's performance. In their experiments, they observed that the BERT model detects PCL with a 78% accuracy, whereas the average over human annotators does so with a 72% accuracy. (Warholm, 2021) also finetuned a BERT model to classify the unhealthy comments in Norwegian data. This model was subjected to a variety of finetuning approaches to distinguish between condescending and non-condescending cases and in the binary classification subtask, the best accuracy was 0.862.

1.1 Data

"Don't patronize me" is an annotated dataset of PCL by (Pérez-Almendros et al., 2020) through crowdsourcing. It is a collection of texts which targets vulnerable communities. The dataset is extracted from News On Web (NoW) corpus[4], containing web articles from over 20 English-speaking countries. It contains 10,637 paragraphs. In addition to the words (*disabled, homeless, hopeless, immigrant, in need, migrant, poor families, refugee, vulnerable and women*) for identifying PCL for annotation in paragraphs, the following traits are also identified as indicators and used for acquiring the dataset:

- Words expressing feeling of pity towards the vulnerable community. For example: *god bless the victims , all those people and their poor families , and i feel so sorry but i want to tell them it was n't my son who did this , it was a different seifeddine*

- Words describing the vulnerable community as lacking certain privileges, knowledge or experience. For example: *After Vatican controversy, McDonald's helps feed homeless in Rome*

- Expressions that present members of the vulnerable community as victims. For example: *the biggest challenge is the no work policy . i think that refugees who come here , or asylum seekers , they 're unable to work and they have kids here – their kids are stateless . that 's really the cause of a lot of stress in the community*

The dataset was annotated by 3 expert annotators. It has two-level classification of PCL: binary classification used to determine if a paragraph has PCL or not, and then categorical label for those with PCL. The categorical classification has three higher-level categories: saviour, expert and poet. "Other" category is the final category to classify all paragraphs with PCL but that do not fit any of the previous categories. The saviour category represents text in which the author is in a privileged class as opposed to the target community. It has two subcategories: unbalanced power relations and Shallow solutions. The expert category is for text where the author is also in a privileged position and presents themselves as knowing better than the target group what their needs are. It also has two subcategories: presupposition and authority voice. The final category "Poet" is identified by how the author frames the community with a literary style writing. It has three subcategories: Metaphor, Compassion and The poorer the merrier.

2 System Overview

The T5 architecture (Raffel et al., 2020) is very similar to the originally proposed architecture of the Transformer by Vaswani et al. (2017). We use the pretrained base version of the model from the

[4]english-corpora.org/now/

HuggingFace hub (Wolf et al., 2020). Input sequence of tokens are mapped to embeddings and then passed to the encoder, which has alternating set of multi-head attention and feed-forward layers. The attention mechanism (Bahdanau et al., 2015) replaces each element of a sequence by a weighted average of the remaining sequence (Raffel et al., 2020). In addition to each self-attention layer of the decoder, there is the standard attention mechanism. As self-attention is order-independent, relative position embeddings are used in the architecture.

The training method (for both pretraining and finetuning) uses maximum likelihood objective (i.e. teacher forcing) and a cross entropy loss (Raffel et al., 2020). The model was pretrained on 34B tokens. Adam optimizer is used for optmization during finetuning. The model has 12 layers each in the encoder and decoder blocks and a total of 220M parameters (Raffel et al., 2020). When we refer to T5, we mean the base model, except where explicitly stated otherwise. The size of the model meant that a batch size of 64 or 32 required more memory than what is available on a single V100 GPU, so we lowered the batch size to 16. T5 takes a hyperparameter called a task prefix. We, hence, use 'classification: ' as the task prefix.

We introduced a correction to the out-of-class prediction of the model, as shown in the flow chart in Figure 1. Raffel et al. (2020) mentioned this issue as a possibility but they did not experience it. The issue appears to be because all the tasks the T5 model is trained on are framed as "text-to-text" before training. Hence, sometimes, the model might predict tokens seen during training but that do not belong to the category of classes in a classification task. This behaviour seems more common in the initial epochs of training and may not even occur sometimes. We further observed that replacing target labels with numbers and explicitly typecasting them as string reduces this occurrence, as the model becomes more stable with predictions.

We split 10% of the training set for validation (dev set) for both of our submissions to the competition. We explored different sizes, however, in further ablation studies, as explained in the next section. The 2 submissions of prediction files are based on 2 adaptive optimizers: Adam and AdamW (Loshchilov and Hutter, 2019). The predictions based on Adam had the better F1 score. Each experimental run was for 3 epochs and the model checkpoint with the lowest validation loss was saved and used to make prediction on the test set. The initial learning rate and scheduler for both submissions are 2e-4 and linear schedule with warmup, respectively.

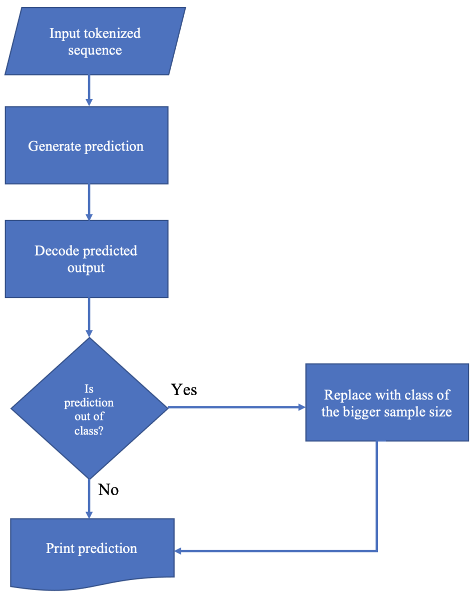


Figure 1: Flowchart of out-of-class code section for the T5 model during prediction.

3 Experimental Setup

All the experiments were conducted on a shared DGX-1 cluster of 8 × 32GB Nvidia V100 GPUs. The server runs on Ubuntu 18 OS and has 80 CPU cores. The experiments were conducted in a Python (3.6.9) virtual environment with the PyTorch framework (1.8.1+cu102). We use both the training & test data provided by Pérez-Almendros et al. (2020). Besides the 2 submissions of prediction files, we perform ablation studies over the training/dev set split ratio (95%/5%, 90%/10%, 85%/15%, and 80%/20%). The training set was shuffled before splitting each dev set. We evaluate all the models using macro F1 scores, precision (P) and recall (R). In the absence of the ground truth of the test set, we perform error analysis by constructing the confusion matrix on a split of the dev set (20%). Further to that, in order to have a basis of comparison of the T5 model's strengths and struggles with the official RoBERTa baseline, we removed the 10 examples provided in Table 5 by Pérez-Almendros et al. (2022) from the training set and concatenated them with the dev set before training and evaluation. The predictions of 9 of the samples are given in Table 3.

Evaluation of the available data, by code, before and after running the script provided by Pérez-Almendros et al. (2022) to categorize the labels into 0 (neg) and 1 (pos) (for subtask 1) reveals that there are a total of 10,469 samples. The script treated paragraphs with the original labels 0 and 1 as 0 (instances not containing PCL) and paragraphs with the original labels 2, 3 and 4 as 1 (instances containing PCL). After running the script, the following are obtained: 9,476 samples classified as 0 and 993 classified as 1 in the training set. The test set has 3,832 samples. Before training, the following preprocessing steps were applied to all splits of the data:

- Emails & URLs are removed.
- All the characters are made lowercase.
- Extra spaces are removed.
- Special characters such as hashtags(#) and emojis are removed.
- Numbers & IP addresses are removed.

4 Results and Discussion

Our model performed relatively well with an F1 score of 0.5452 in the official assessment. This made it rank 27 (the 66th percentile) out of the 78 scores. All the F1 scores we report are macro scores. Our model has 11% advantage over the RoBERTa baseline, which achieved 0.4911, as shown in Table 1. Indeed, our second submission, based on the AdamW optimizer, also performs better than the baseline, achieving an F1 score of 0.5282, precision and recall of 0.5976 and 0.4732, respectively. The T5 model may have performed even better in the official rankings but for the shortcoming we described in section 2. In ablation studies, as shown in Table 2, we observe that training/dev set split ratio affects the performance of the system. All the results are based on submissions to the official evaluation system[5]. Using 5% of the training set as the dev set gave the worst F1 score but we observe improvements as the size is increased, though not linearly. We observe a sharp rise in F1 score when we increase the split from 5% to 10% but the rate of increase falls for subsequent increases.

[5]competitions.codalab.org/competitions/34344

Model	Rank	P	R	F1
best	1	0.646	0.6562	0.651
T5 (ours)	27	0.5801	0.5142	0.5452
RoBERTa baseline	43	0.3935	0.653	0.4911
worst	78	0.1059	0.0284	0.0448

Table 1: Abridged official result ranking for subtask 1.

Model (dev split)	P	R	F1
T5 (5%)	0.0725	0.8643	0.1339
T5 (10%)	0.6725	0.3628	0.4713
T5 (15%)	0.6067	0.4574	0.5216
T5 (20%)	0.5818	0.5047	0.5405

Table 2: Ablation studies results on the test set for subtask 1. Hyperparameters are the same for all model modifications. The T5 (10%) model is retrained afresh like the others, to avoid test/dev set feedback because of the samples in table 3.

4.1 Error Analysis

Since the ground truth labels of the test set are not available, we perform error analysis on the dev set. The T5 (20%) model achieves an F1 score of 0.7405 on the dev set (20%). However, the confusion matrix, as depicted in Figure 2, reveals that the model predicted 0 (neg) correctly 96.4% of the time while struggling to make the correct predictions when it came to 1 (pos), making only 47.8% of predictions correctly. This is very likely due to data imbalance, as 90.5% of the total training set contains samples labeled as 0 (neg). Ways of mitigating this may include data augmentation, possibly in a similar strategy to that used by Sabry et al. (2022), where an autoregressive model was deployed (Adewumi et al., 2022). A more careful stratification of the data split may also be helpful in this case.

Pérez-Almendros et al. (2022) report that the models they considered struggled to detect certain categories of PCL. We observe a similar challenge though our model achieves a better performance than the official baseline. For example, our T5 (20%) model's predictions for the same examples shown by Pérez-Almendros et al. (2022) for subtask 1 reveal that our model correctly predicts 5 out of the 9 displayed in Table 3, unlike the 3 correct predictions out of the 10 by the official baseline. The reason the T5 (20%) may have misclassified 2 of the samples labeled 0 (neg) in Table 3 may be because of tokens such as *vulnerable patients* and *hopelessly*, since they belong to the keywords used for annotating paragraphs with PCL, as discussed

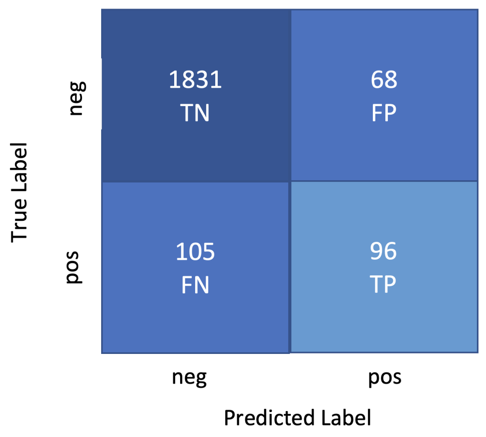


Figure 2: Confusion matrix of T5 (20%) on the dev set (20%). Macro F1 (0.7405): [0.9549 (neg) 0.5260 (pos)]

in section 1.

5 Conclusion

We describe the system involving the pretrained T5 model, which we use for our submission for the subtask 1 of the SemEval-2022 Task 4. We split 10% of the training set as dev set for hyperparameter evaluation in our official submission. Typecasting integer values, which represent classes, as string before feeding the T5 model and adjusting for out-of-class predictions improved the stability of the model in making predictions. Furthermore, in the post-competition phase, we performed ablation studies on the relative importance of dataset split by experimenting with different ratios of the training/dev set and showed what the model struggles with. Our results show that the encoder-decoder T5 model is competitive in this binary task and can obtain better performance with more hyperparameter tuning.

Acknowledgements

The authors wish to thank the anonymous reviewers for their feedback and the task organisers for their prompt attention whenever it was required.

Pred.	Paragraph	Gold
pos	from his personal story and real-life encounters with poor families , manpower correspondent toh yong chuan suggested shifting the focus from poor parents who repeatedly make bad decisions to their children ("""" lifting families out of poverty : focus on the children ; last thursday) .	pos
pos	he said their efforts should not stop only at creating many graduates but also extended to students from poor families so that they could break away from the cycle of poverty	pos
neg	smyth told hkfp : """" the biggest challenge is the no work policy . i think that refugees who come here , or asylum seekers , they 're unable to work and they have kids here – their kids are stateless . that 's really the cause of a lot of stress in the community .	pos
neg	the people of khyber pakhtunkhwa are resilient . i did not see hopelessness on any face , """" he said .	pos
pos	teach kids to give back : when kang runs summer camps with kids , she includes """" contribution fridays """" – the kids work together as a team to make sandwiches for the homeless and dole out the food in shelters .	pos
pos	these shocking failures will continue to happen unless the government tackles the heart of the problem – the chronic underfunding of social care which is piling excruciating pressure on the nhs , leaving vulnerable patients without a lifeline .	neg
neg	lilly-hue : his ability to make sure our family is never in need – his sacrificial self .	neg
pos	"any kenyan small-scale farmer with such an income could not be said to be hopelessly mired in agrarian destitution . but of course , nothing in life is ever so simple as to allow for neat and precise answers ."	neg
neg	"selective kindness : in europe , some refugees are more equal than others"	neg

Table 3: Example predictions by T5 based on the dev set

References

Tosin Adewumi, Rickard Brännvall, Nosheen Abid, Maryam Pahlavan, Sana Sabah Sabry, Foteini Liwicki, and Marcus Liwicki. 2022. Småprat: Dialogpt for natural language generation of swedish dialogue by transfer learning. In *5th Northern Lights Deep Learning Workshop, Tromsø, Norway*, volume 3. Septentrio Academic Publishing.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations, ICLR 2015*.

Katherine M Bell. 2013. Raising africa?: Celebrity and the rhetoric of the white saviour. *PORTAL: Journal of Multidisciplinary International Studies*, 10(1):1–24.

Susan Anne Fox and Howard Giles. 1996. Interability communication: Evaluating patronizing encounters. *Journal of Language and Social Psychology*, 15(3):265–290.

Shlok Gilda, Luiz Giovanini, Mirela Silva, and Daniela Oliveira. 2022. Predicting different types of subtle toxicity in unhealthy online conversations. *Procedia Computer Science*, 198:360–366.

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

Vann Morris. 2007. Patronizing speech in interability communication toward people with cognitive disabilities.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ilan Price, Jordan Gifford-Moore, Jory Flemming, Saul Musker, Maayan Roichman, Guillaume Sylvain, Nithum Thain, Lucas Dixon, and Jeffrey Sorensen. 2020. Six attributes of unhealthy conversations. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 114–124, Online. Association for Computational Linguistics.

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

Sana Sabah Sabry, Tosin Adewumi, Nosheen Abid, György Kovacs, Foteini Liwicki, and Marcus Liwicki. 2022. Hat5: Hate language identification using text-to-text transfer transformer.

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

Zijian Wang and Christopher Potts. 2019. TalkDown: A corpus for condescension detection in context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3711–3719, Hong Kong, China. Association for Computational Linguistics.

Joakim Warholm. 2021. Detecting unhealthy comments in norwegian using bert. Master's thesis, UiT Norges arktiske universitet.

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

Acronyms

BERT Bidirectional Encoder Representations from Transformers. 2

LTU Luleå University of Technology. 1

NLP Natural Language Processing. 1

PCL patronizing and condescending language. 1, 2, 4

RoBERTa Robustly optimized BERT approach. 1, 3, 4

SoTA state-of-the-art. 1

T5 Text-to-Text-Transfer Transformer. 1–5

Xu at SemEval-2022 Task 4: Pre-BERT Neural Network Methods vs Post-BERT RoBERTa Approach for Patronizing and Condescending Language Detection

Jinghua Xu
University of Tübingen
jinghua.xu@student.uni-tuebingen.de

Abstract

This paper describes my participation in the SemEval-2022 Task 4: Patronizing and Condescending Language Detection. I participate in both subtasks: Patronizing and Condescending Language (PCL) Identification and Patronizing and Condescending Language Categorization, with the main focus put on subtask 1. The experiments compare pre-BERT neural network (NN) based systems against post-BERT pretrained language model RoBERTa. This research finds that NN-based systems in the experiments perform worse on the task compared to the pretrained language models. The top-performing RoBERTa system is ranked 26 out of 78 teams (F1-score: 54.64) in subtask 1, and 23 out of 49 teams (F1-score: 30.03) in subtask 2.

1 Introduction

An entity is considered to engage Patronizing and Condescending Language (PCL) when its language use presents a superior attitude towards others or depicts them in a compassionate way (Pérez-Almendros et al., 2020). Such language is often used toward vulnerable communities such as women, refugees, and homeless people. These unfair treatments of the vulnerable groups are believed to result in further exclusion and inequalities in society. Compared to other types of harmful language (e.g. hate speech), PCL is considered more subtle and unconscious. Given the negative effects of PCL on society and its subtle nature, enabling computers to identify and categorize PCL presents an interesting technical challenge to the NLP community.

This paper describes my participation in both subtasks of the SemEval-2022 Task 4: Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022). Subtask 1, Patronizing and Condescending Language Identification is a binary text classification task to predict whether a given paragraph contains PCL or not. Subtask 2, Patronizing and Condescending Language Categorization is a multi-label classification task to identify the categories of a given paragraph according to the taxonomy defined in Pérez-Almendros et al. (2020), which categorizes PCL into 7 types: *1) Unbalanced power relations 2) Shallow solution 3) Presupposition 4) Authority voice 5) Metaphor 6) Compassion 7) The poorer, the merrier*. The dataset (Pérez-Almendros et al., 2020) contains annotated paragraphs in English, collected from news stories in 20 English-speaking countries.[1]

The focus of my experiments is primarily on subtask 1, meanwhile, this paper also proposes a solution to subtask 2. For subtask 1, the experiments compare pre-BERT neural network (NN) based systems including a majority voting system of NN models against pretrained language model RoBERTa. The experiments start with building individual NN models from the most basic artificial neural network (ANN) to long short-term memory network (LSTM) models following previous work on NN for text classification. It was found that the NN-based systems in the experiments perform worse on this task in comparison to the pretrained language models. The best-performing NN-based voting system could not outperform the RoBERTa baseline model. For subtask 2, this paper simply proposes a RoBERTa solution.

The code is released at: github.com/JINHXu/PCL-Detection-SemEval2022-task4.

2 Background

Numerous previous research had been conducted on the treatment of condescension and patronization. The studies range in various areas from so-

[1] 19 countries and Hong Kong: Australia, Bangladesh, Canada, Ghana, Ireland, India, Jamaica, Kenya, Sri, Lanka, Malaysia, Nigeria, NewZealand, Philipines, Pakistan, Singapore, Tanzania, UK, United States, South Africa, and the special administrative region of China, Hong Kong.

ciolinguistics (Irisawa et al., 1993) to medicine (Komrad, 1983). Whereas in the field of natural language processing, automatically identifying or categorizing PCL has been an understudied area. Most research on harmful language detection has focused on more explicit and aggressive topics such as hateful speech (MacAvaney et al., 2019), offensive language (Zampieri et al., 2019), and fake news (Conroy et al., 2015). Only in recent years, few in the research community have started to show interest in enabling computers to identify condescending language. Prior to this shared task, for instance, Wang and Potts (2019) proposed an annotated TalkDown corpus of condescending language from social media.

3 Dataset

The corpus used for this shared task, the Don't Patronize Me! dataset is described in Pérez-Almendros et al. (2020). The corpus consists of 10,637 paragraphs extracted from the News on Web (NoW) corpus (Davies, 2013). The paragraphs were selected according to 10 keywords related to potentially vulnerable communities (disabled, homeless, hopeless, immigrant, in need, migrant, poor families, refugee, vulnerable, and women). And for each keyword, a similar number of paragraphs were chosen for each of the 20 English-speaking countries.[2] Each paragraph is annotated with a true/false label indicating whether it contains PCL or not, and the ones that contain PCL are annotated with a category label. Each category label is a set of 7 binary predictions, each prediction indicates the existence of a specific type of PCL.

The dataset is highly imbalanced. The POS:NEG ratio is approximately 1:10, which poses a challenge to the predictive modeling process. In the experiments of this paper, various strategies were employed in order to deal with the imbalance in data. The following sections will describe these strategies in detail.

Additionally, the shared task provides a comparable 80/20 split of the training data for development. In the experiments of this paper, the same split was used to train models and generate predictions in the development stage.

[2]Except for Hong Kong, which is not a country.

4 Model

4.1 Preprocessing

In the preliminary experiments, it was found that preprocessing data by removing stop words decreases model performance. Thus in the following model training process, the text data are used as-is. Furthermore, in order to deal with data imbalance, various approaches including data oversampling, undersampling, and setting class weights were experimented with. The neural network models were found in preliminary experiments to work the best with the original data, with class weights set to 10:1 according to the POS:NEG ratio in data. Whereas the RoBERTa models present the best performance with oversampled data with default class weights.

4.2 Neural Network Models

The experiments start with exploring NN models for the binary text classification subtask. Previous work has shown that linear classic machine learning models such as Linear SVM (Suthaharan, 2016), Bernoulli Naive Bayes (Webb et al., 2010), and Logistic Regression (Wright, 1995) have advanced performance on binary text classification with proper feature engineering. This paper is, however, interested in exploring neural network solutions to the task, given the sufficient size of the dataset.

Neural network models have been regarded to be capable of achieving remarkable performance on text classification. In addition to the popular LSTM (Hochreiter and Schmidhuber, 1997) models, some basic ANN (McCulloch and Pitts, 1943) models have also been proved in previous work to perform well on the task of binary text classification. The experiments of this paper start with building individual NN models from the most basic ANN models to the more sophisticated LSTM models. Furthermore, in order to continue improving system performance from the individual models, a majority voting system that uses the predictions of both of the best-performing ANN and LSTM models was built.

4.2.1 Basic ANNs

Common basic ANN architectures for binary text classification tasks typically consist of an Embedding layer, a pooling layer of different types (average, minimum, maximum), and various dense layers. Following the previous work, the experiments start with building a baseline ANN model using a GloVe (Pennington et al., 2014) embedding layer for word representation, with a `GlobalAveragePooling1D` built on top of it, followed by a ReLU layer, and a sigmoid layer to generate predictions. In the preliminary experiments, various types of pooling were tried, and the model with the `GlobalAveragePooling1D` layer presented the best performance. The confidence threshold was initially set to 0.5, which resulted in low precision and high recall. Thus the threshold was gradually increased in experiments, with 0.7 found to generate the best predictions in the development stage.

In order to improve the ANN model from the baseline, more dense layers were added to the network gradually. Since there are no rules of thumb in building a neural network, the strategy employed in this experiment is to continue adding dense layers of tanh and ReLU to the baseline model before the output layer. The F-score reaches a peak value after two additional tanh layers with a ReLU layer in between were added to the baseline model. Adding more dense layers did not further help increase the model performance in the experiments.

4.2.2 LSTM

The LSTM model uses the same GloVe embedding layer for word representation, with a single layer of LSTM units (output dimension size 60) built on top of it. A `GlobalMaxPool1D` layer is built on top of the LSTM layer, followed by a ReLU layer, and a sigmoid layer to generate predictions. The dropout rate is set to 0.1. With the confidence threshold set to 0.5, relatively even precision and recall were obtained.

4.2.3 NN voting system

NN model performance can be unstable in each run, this was also confirmed in the experiments of this paper. In order to handle the instability, also to continue increasing system performance from the individual models, a majority voting system based on both NN models was built. The system considers the predictions of both the ANN and the LSTM models in two separate runs, which results in four votes in total. The systems prediction for each paragraph is then based on the majority vote of the four votes produced by both models in two runs.

All NN models in the experiments are implemented using `tensorflow.keras` (Chollet et al., 2015). During training, each model uses 10% of data for validation, with class weights set to 10:1 as mentioned in a previous section.

The hyperparameter tuning process in this experiment focuses on batch size and the number of training epochs. For each model, batch sizes of 16, 32, 64, 128, and training epochs of 10, 50, 100 were tried. All models present the best performance with the number of training epochs set to 50. The LSTM model works the best with training batch size of 128, and ANN models with 32.

4.3 Pretrained Language Model: RoBERTa

For both shared tasks, this paper proposes a RoBERTa (Liu et al., 2019) solution. RoBERTa is regarded as an improved pretraining procedure from BERT (Devlin et al., 2018), and it is able to match or exceed the performance of all post-BERT methods. All pre-trained language models in the experiments are implemented using the `simpletransformers` library (Rajapakse, 2019).

In subtask 1, the shared-task provides a baseline `roberta-base` model with default configurations, trained on undersampled data. On top of this work, I further tuned the hyperparameters (mainly the number of training epochs, in the search space: 1, 2, 3, 5, 10) and improved model configurations using manually oversampled/undersampled PCL data of various POS:NEG ratios, class weights for fine-tuning. In addition to the `roberta-base` model used in the baseline model, I also experimented with a number community models[3] alternative to `roberta-base`. Among the community models, the experiments mainly focus on BERT- and RoBERTa-based models for toxic language detection and sentiment analysis, given the similarity and relevance of the tasks to PCL detection. Appendix A lists the community models tried in the experiments. However, none of these community models in Appendix A turned out to work better than `roberta-base` in my experiments. I believe the models are too specialized in their own tasks (e.g. sentiment analysis, toxic language detection), therefore resulting in poor performance on the PCL detection task.

The best-performing model in the development stage is a `roberta-base` model, with the number of training epochs set to 1, maximum input sequence length increased to 500.[4] The data was

[3]A list of community models can be found on the website: huggingface.co/models.

[4]The longest paragraph in training data is of the length

balanced by manually repeating all positive data instances 9 times, and keeping the same number of negative data instances for training. The balanced dataset results in 8937 data instances of each class for training. In order to reduce training time, GPUs were used for model training and inference. All RoBERTa-related experiments were conducted on Google Colab.[5]

In subtask 2, the shared-task also provides a RoBERTa baseline model configured with `roberta-base` model, with default configurations. The baseline model is trained on undersampled data (794 positive data instances, 397 negative data instances). I simply increased the maximum input sequence length to 500 from the baseline model and oversampled data to obtain 7146 positive data instances and 7146 negative data instances for training.

5 Results

5.1 Subtask 1: PCL Detection

Model	Precision	Recall	F-score
ANN$^{\text{baseline}}$	32.63	39.19	35.61
ANN	36.50	46.23	40.79
LSTM	41.44	46.23	**43.70**
voting_NN	46.29	40.70	43.32
RoBERTa$^{\text{baseline}}$	40.98	50.25	45.14
RoBERTa-base	51.15	66.83	**57.95**
BERT-emotion	35.93	41.7	38.6
RoBERTa-toxic	37.5	66.33	47.91

Table 1: Model performance on development data.

Table 1 shows the precision, recall, and F1-score of the models in the development stage. Among the neural network systems, the LSTM model presents the most advanced performance with an F-score of 43.7. Meanwhile, the voting system has an F-score (43.32) only slightly lower than the LSTM model. The F-score of the ANN model is lower than the LSTM model, however, the performance gap is not huge. Nevertheless, the best-performing neural network based system LSTM does not outperform the RoBERTa baseline model.

The tuned RoBERTa-base model has the highest score of 57.8 among all systems in the development stage. As mentioned in a previous section, the community models pretrained on sentiment or toxicity language data present poor performance on the PCL data compared to the base model of RoBERTa.[6] Overall, for each model in the development stage, the difference between precision and recall is not vast, except for the tuned RoBERTa model and the toxicity model of RoBERTa.

Model	Precision	Recall	F-score
ANN$^{\text{baseline}}$	28.34	48.90	35.88
ANN	26.62	67.19	38.14
LSTM	38.31	50.16	43.44
voting_NN	48.50	40.69	**44.25**
RoBERTa$^{\text{baseline}}$	39.02	62.78	48.13
RoBERTa-base	46.19	66.88	**54.64**
BERT-emotion	36.54	35.96	36.25
RoBERTa-toxic	25.19	84.54	38.81

Table 2: Model performance on test data.

Table 2 presents model performance on the test data in the evaluation stage. It is notable that among the neural network models, the top-performing system on test data becomes the voting system (F-score: 44.25) instead of the LSTM model, which performs the best in the development stage. The F-score of the voting system in the evaluation stage is also higher than in the development stage. Nonetheless, as the top-performing neural network based system in the evaluation stage, the NN voting system presents an F-score still lower than that of the RoBERTa baseline model (F-score 48.13). Both the LSTM and the ANN model F-score decreased from in the development stage. While the baseline ANN model presents a similar F-score on the test data to that on the development data. In general, in the evaluation stage, the gap between precision and recall is rather big for both the ANN baseline and the ANN model, whereas it is small for the LSTM model and the NN voting system. By comparing the performance of the neural network based systems during the development stage to the evaluation stage, it can be seen that the performance of the ANN models is less stable compared to the LSTM and the voting system.

The tuned RoBERTa-base model is still the top performer among all models in the evaluation stage, with an F-score of 54.64. However, this score decreased from in the development stage. While for the RoBERTa baseline model, the F-score in the

between 400 and 500, in case of future longer data instances to predict on, the model's maximum input sequence length is set to 500.

[5]colab.research.google.com/

[6]Only the performance of two of the community models tried in the experiments are presented in the tables.

evaluation stage is higher than in the development stage. As for the two community models, their performance on test data is also worse than on the development data. Overall, every RoBERTa model shows higher recall than precision with a notable gap. This is also true for the ANN models as mentioned in the previous paragraph. These models produce more false positives than false negatives. While for the BERT-based emotional model, it shows similar precision and recall, although also a low F-score. In general, for every model in the evaluation stage except for the BERT model, the gap between precision and recall further increased from that in the development stage.

5.2 Subtask 2: PCL Categorization

F-score	RoBERTa$^{\text{baseline}}$	RoBERTa
Unb. power rel.	35.35	55.94
Shallow solu.	00.00	31.74
Presupposition	29.63	24.44
Authority voice.	00.00	19.35
Metaphor	00.00	23.88
Compassion	28.78	45.83
The p., the mer.	00.00	15.38
Average	13.40	30.94

Table 3: Model performance on development data.

Table 3 presents the per-class and average F-scores of the RoBERTa baseline model and the proposed RoBERTa model for subtask 2 in the development stage. The proposed RoBERTa model is able to produce a higher F-score for each class with the exception of the *Presupposition* category. Overall, the average F-score is improved from baseline by around 17%.

F-score	RoBERTa$^{\text{baseline}}$	RoBERTa
Unb. power rel.	35.35	54.38
Shallow solu.	00.00	47.06
Presupposition	16.67	26.92
Authority voice.	00.00	24.06
Metaphor	00.00	11.11
Compassion	20.87	46.72
The p., the mer.	00.00	00.00
Average	10.41	30.03

Table 4: Model performance on test data.

Table 4 presents the per-class and average F-scores of the RoBERTa baseline model and the proposed RoBERTa model for subtask 2 in the evaluation stage. As can be seen from the table, the proposed RoBERTa model increased the per-class F-score from the baseline model for each category except for only the *the poorer the merrier* class, for which neither the baseline model nor the proposed model is able to detect. The average F-score of the proposed model is also increased from that of the baseline model. However, the score slightly decreased in the evaluation stage from in the development stage.

6 Conclusion

The experiments of this paper compare some of the pre-BERT neural network based systems against the post-BERT pretrained language model RoBERTa. The experiments start with building individual NN models from the most basic ANN models to the more sophisticated LSTM models, and create a majority voting system based on the individual NN models. It was found that the NN-based systems in the experiments perform worse on the task compared to the RoBERTa baseline model. And the community models pretrained on relevant data such as sentiment and toxicity data turn out to be too specialized in their own task thus resulting in poor performance on the PCL data compared to the RoBERTa-base model.

This paper explores neural network models mainly the basic ANN and LSTM models. Future work should also consider convolutional neural networks (CNN) for PCL detection. In addition to the neural networks, I suggest also investigating classic machine learning models such as Logistic Regression, as well as indicative linguistic features of PCL for feature engineering. Furthermore, on top of the RoBERTa-base model, I propose to pretrain a RoBERTa model using the TalkDown corpus proposed in Wang and Potts (2019), and fine-tune the pretrained model using the PCL data. Finally, future work should run further error analysis of the models to improve performance.

Acknowledgements

I thank Dr. Çağrı Çöltekin for his guidance and help throughout the process.

References

Francois Chollet et al. 2015. Keras.

Nadia K Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. *Proceedings of the association for information science and technology*, 52(1):1–4.

Mark Davies. 2013. Corpus of news on the web (now): 3+ billion words from 20 countries, updated every day. *Retrieved January*, 25:2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

H Irisawa, HF Brown, and W Giles. 1993. Cardiac pacemaking in the sinoatrial node. *Physiological reviews*, 73(1):197–227.

Mark S Komrad. 1983. A defence of medical paternalism: maximising patients' autonomy. *Journal of medical ethics*, 9(1):38–44.

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

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. 2019. Hate speech detection: Challenges and solutions. *PloS one*, 14(8):e0221152.

Warren S McCulloch and Walter Pitts. 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

T. C. Rajapakse. 2019. Simple transformers. `https://github.com/ThilinaRajapakse/simpletransformers`.

Shan Suthaharan. 2016. Support vector machine. In *Machine learning models and algorithms for big data classification*, pages 207–235. Springer.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context.

Geoffrey I Webb, Eamonn Keogh, and Risto Miikkulainen. 2010. Naïve bayes. *Encyclopedia of machine learning*, 15:713–714.

Raymond E Wright. 1995. Logistic regression.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.

A Appendix: List of Community Models

mohsenfayyaz/toxicity-classifier
mohsenfayyaz/roberta-base-toxicity
SkolkovoInstitute/roberta_toxicity_classifier
mohsenfayyaz/toxicity-classifier
DaNLP/da-bert-emotion-binary
siebert/sentiment-roberta-large-english

Amsqr at SemEval-2022 Task 4: Towards AutoNLP via Meta-Learning and Adversarial Data Augmentation for PCL Detection

Alejandro Mosquera
Broadcom Corporation / 1320 Ridder Park Drive San Jose, 95131 California, USA
alejandro.mosquera@broadcom.com

Abstract

This paper describes the use of AutoNLP techniques applied to the detection of patronizing and condescending language (PCL) in a binary classification scenario. The proposed approach combines meta-learning, in order to identify the best performing combination of deep learning architectures, with the synthesis of adversarial training examples; thus boosting robustness and model generalization. A submission from this system was evaluated as part of the first subtask of SemEval 2022 - Task 4 and achieved an F1 score of 0.57%, which is 16 percentage points higher than the RoBERTa baseline provided by the organizers.

1 Introduction

The harmful use of language in social media can have negative and long-lasting effects such as exclusion and unfair treatment, specially when targeting vulnerable communities. For this reason, the detection of toxic, hateful and abusive comments has been the central topic of several workshops and tool evaluations, drawing a lot of attention from the Natural Language Processing (NLP) research community in the last years. However, while toxic language has a clear intent and is usually obvious to the reader, patronizing and condescending language (PCL) is more subtle and likely used in a subconscious manner even in traditional media (Perez Almendros et al., 2020). The aforementioned characteristics and its subjective nature makes PCL harder to identify than abusive comments by both humans (Sap et al., 2019) and NLP applications.

The continuously increasing taxonomies of language misuse poses new challenges to social media platforms, thus not only requiring more effort and cost in order to identify abuse across different languages and textual genres but also having to keep a balance between aggressive and conservative filtering strategies. On the one hand, users eventually devise ways of evading automatic content moderation (Gerrard, 2018), while on the other hand, policing that restricts freedom of speech can lead to distrust (Kirk and Schill, 2021). For these reasons, content filters usually rely on the latest advances in NLP research, dominated in the recent years by deep learning architectures. Despite the competitive scores achieved via transfer learning and models such as the Transformer (Vaswani et al., 2017) in this area, choosing and optimizing the right modeling framework for a given NLP task is still a non-trivial problem.

Automated Natural Language Processing (AutoNLP), the equivalent of Automated Machine Learning (AutoML) for NLP, is a relatively new field of study that aims to automate the iterative components of developing a NLP model given a specific input data and task without requiring any special domain expertise. By building upon existing concepts such as transfer learning, data augmentation and meta-learning the author hypothesizes that is possible to generate strong NLP baselines with minimal human interaction. An analysis of the results of the shared task 4 of SemEval-2022: Patronizing and Condescending Language Detection (Pérez-Almendros et al., 2022) shows that AutoNLP can be successfully applied to PCL classification, obtaining a 16% higher F1 score than the baseline provided by the task organizers.

This paper is organized as follows: In Section 2, the state of the art is reviewed. Further on, Section 3 describes the AutoNLP approach for PCL classification. Next, in Section 4, an in-depth discussion of the results obtained is described, and finally Section 5 concludes this research and outlines future work.

2 Related Work

There have been several research works on the detection of different types of harmful language, not only focused on the most explicit such as hate speech (Zampieri et al., 2019) (Garibo i Orts, 2019) but also more subtle usages such as condescending interactions (Wang and Potts, 2019) and social power implications (Sap et al., 2020). PCL towards vulnerable communities in news articles has also been characterized into 7 categories (Perez Almendros et al., 2020) used in order to label the most comprehensive PCL-annotated corpus to date: the Don't Patronize Me! (DPM) dataset, the official training resource for the shared task 4 of SemEval-2022: Patronizing and Condescending Language Detection.

3 AutoNLP for PCL

Deep neural network modeling techniques have inspired state of the art approaches in various domains, such as image classification and language modeling, thus dominating several benchmarks and shared tasks in the last years. For this reason, NLP applications relying on manually-crafted features have been less popular in comparison with deep learning (DL) architectures (Young et al., 2018), specially where extensive manual feature engineering is required to achieve a similar performance (Mosquera, 2021). However, since building a high-quality DL system for a specific task still relies on human expertise, AutoML offers a promising solution to this problem by automating most of the modeling steps (He et al., 2021).

In order to tackle an arbitrary NLP classification task, in this case PCL detection, a custom end to end AutoNLP solution has been designed and evaluated by using exclusively the DPM dataset provided by the organizers, off-the-shelf pre-trained models and without applying any special pre-processing or feature engineering besides standard tokenization. The main components of the system are described in the following section.

3.1 Adversarial Data Augmentation

Adversarial data augmentation can not only increase model robustness but also improve generalization by increasing the number of training samples (Shorten et al., 2021). This can be specially relevant when using neural networks, which tend to under-perform in a low-data regime (Antoniou et al., 2018). The different data augmentation strategies incorporated in the AutoNLP pipeline are as follows:

- **Backtranslation**: Transformation using TextAttack (Morris et al., 2020) that translates a PCL sentence into a random target language and translates it back to English.
- **Checklist**: TextAttack implementation of the Invariance Testing Method: Contraction, Extension, Changing Names, Number, Location (Ribeiro et al., 2020) applied to the positive class.
- **Wordnet**: Word swap by swapping synonyms in WordNet (Fellbaum, 1998) for PCL paragraphs.
- **Embedding**: Attack that replaces words with synonyms in the word embedding space (Mrkšić et al., 2016) for PCL texts.
- **Counterfactual**: Inspired by the concept of counterfactual augmentation (Kaushik et al., 2020), this manipulation only applies to text from the positive class which is augmented with random texts from the negative class. The resulting paragraph should still have a positive (PCL) label.
- **Shuffle**: Attack that shuffles words in a PCL paragraph.
- **Parrot**: Paraphrased PCL sentences generated with Parrot (Damodaran, 2021).
- **Pegasus**: PCL augmentation by generating paraphrases via conditional augmentation using Pegasus (Zhang et al., 2019).

3.2 Meta-learning

A common approach to meta-learning is stacked generalization (Wolpert, 1992), where a set q of base learners applied to a training set T_{train} : $\{(\tilde{X}_i, c_i)\}_{i=1}^{m}$ to produce q hypotheses $\{h_j\}_{j=1}^{q}$ is redefined into a new set T'_{train} by replacing each vector $\tilde{X}_i$ with the class predicted by each of the q hypothesis on $\tilde{X}_i$. T'_{train} is used as input to a set of meta-learners, producing a new set of hypotheses (Vilalta and Drissi, 2001).

While this approach has been successfully applied in several NLP tasks (Li and Zou, 2017) (Mosquera, 2020), an small variation that deals with skewed datasets and automatically sub-samples the majority class in each base learner (Chan and Stolfo, 1998) was considered instead for this challenge. In order to do this, a pool of 40 base learners was generated by randomly combining different data augmentation approaches, deep learning architectures via transfer learning and sub-sampling factors. Logistic regression was used as meta-learner in the second layer, with probability thresholds and hyper-parameters optimized via cross-validation.

Several pre-trained resources were used for fine-tuning with early stopping including BERT (Devlin et al., 2019), ELECTRA (Clark et al., 2020), GloVe (Pennington et al., 2014) embeddings with capsule networks (Frosst et al., 2018), RoBERTa (Liu et al., 2019) and XLNet (Yang et al., 2019). The number of optimal training epochs was determined via cross-validation. However, for cost mitigation purposes, no model was trained for longer than 10 epochs and most hyper-parameters were left with the default values.

3.3 Model Selection

The maximum relevance and minimum redundancy (MRMR) algorithm (Zhao et al., 2019) was applied as feature selection method, reducing the final number of base learners used by the meta-model to 8.

After analyzing the cross-validation results we can observe that base models fine-tuned with ELECTRA obtained the highest F1 scores. Likewise, the most successful data augmentation was the combination of the Checklist and Backtranslation methods. The final list of base learners, including their cross validation F1 score and logistic regression coefficient is shown in Table 1.

Model	Augmentations	F1	β
BERT	Checklist	0.52	**0.31**
ELECTRA	Checklist	**0.55**	0.25
ELECTRA	Checklist Backtranslation	**0.55**	0.17
ELECTRA	Checklist Backtranslation Embedding Counterfactual Wordnet	0.54	0.13
RoBERTa	Checklist Backtranslation	0.53	0.30
RoBERTa	Parrot	0.54	0.14
RoBERTa	Checklist Backtranslation Embedding	0.54	0.09
RoBERTa	Checklist Backtranslation Embedding Counterfactual Wordnet	0.53	0.13

Table 1: Final list of base learners selected via MRMR with their cross-validation score and regression coefficient estimated during the training phase.

4 Evaluation and Results

Final test set results obtained in the PCL classification task by the AutoML system (amsqr) and the winning submission (hudou) can be found in Table 2. The official RoBERTa baseline and the development set results are also included for comparison purposes.

Model	Precision	Recall	F1
hudou	**0.646**	**0.656**	**0.651**
amsqr (dev)	0.587	0.578	0.582
amsqr (test)	0.547	0.599	0.572
RoBERTa baseline	0.393	0.653	0.491

Table 2: PCL classification results.

The fact that only 42 out of 78 competing teams were able to beat the RoBERTa baseline provided by the task organizers highlights the difficulty of this competition. Besides the nature of the task, other challenging factors were the strong class imbalance and the considered evaluation metric, which required careful tuning of classification thresholds via cross-validation (Lipton et al., 2014). A post-competition analysis in Table 3 shows that the automatically chosen classification threshold of 0.26 during training was also optimal for the test set.

Threshold	Precision	Recall	F1
0.20	0.498	**0.656**	0.566
0.22	0.516	0.634	0.569
0.24	0.532	0.621	**0.573**
0.28	0.558	0.586	0.572
0.30	**0.566**	0.574	0.570

Table 3: Post-competition classification results in the test set for different probability thresholds.

5 Conclusion and Future Work

This paper describes the system developed for the PCL detection task of SemEval 2022. The author demonstrates that the selected AutoNLP approach can produce competitive results by leveraging meta-learning, adversarial data augmentation and pre-trained resources. Automatic hyper-parameter optimization and exploring different meta-learning algorithms are left to a future work.

References

Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2018. Data augmentation generative adversarial networks.

Philip Ka-Fai Chan and S. Stolfo. 1998. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD*.

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

Prithiviraj Damodaran. 2021. Parrot: Paraphrase generation for nlu.

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

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Nicholas Frosst, Sara Sabour, and Geoffrey Hinton. 2018. Darccc: Detecting adversaries by reconstruction from class conditional capsules.

Òscar Garibo i Orts. 2019. Multilingual detection of hate speech against immigrants and women in Twitter at SemEval-2019 task 5: Frequency analysis interpolation for hate in speech detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 460–463, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Ysabel Gerrard. 2018. Beyond the hashtag: Circumventing content moderation on social media. *New Media & Society*, 20:4492 – 4511.

Xin He, Kaiyong Zhao, and Xiaowen Chu. 2021. Automl: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Rita Kirk and Dan Schill. 2021. Sophisticated hate stratagems: Unpacking the era of distrust. *American Behavioral Scientist*, page 00027642211005002.

Wen Li and Liang Zou. 2017. Classifier stacking for native language identification. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 390–397, Copenhagen, Denmark. Association for Computational Linguistics.

Zachary Chase Lipton, Charles Elkan, and Balakrishnan Narayanaswamy. 2014. Thresholding classifiers to maximize f1 score.

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

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Alejandro Mosquera. 2020. Amsqr at SemEval-2020 task 12: Offensive language detection using neural networks and anti-adversarial features. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1898–1905, Barcelona (online). International Committee for Computational Linguistics.

Alejandro Mosquera. 2021. Alejandro mosquera at SemEval-2021 task 1: Exploring sentence and word features for lexical complexity prediction. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 554–559, Online. Association for Computational Linguistics.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Connor Shorten, Taghi Khoshgoftaar, and Borko Furht. 2021. Text data augmentation for deep learning. *Journal of Big Data*, 8.

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

Ricardo Vilalta and Youssef Drissi. 2001. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context.

David Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

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

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2018. Recent trends in deep learning based natural language processing.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Zhenyu Zhao, Radhika Anand, and Mallory Wang. 2019. Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform.

SATLab at SemEval-2022 Task 4: Trying to Detect Patronizing and Condescending Language with only Character and Word N-grams

Yves Bestgen
Statistical Analysis of Text Laboratory (SATLab)
Université catholique de Louvain
Place Cardinal Mercier, 10 1348 Louvain-la-Neuve, Belgium
yves.bestgen@uclouvain.be

Abstract

A logistic regression model only fed with character and word n-grams is proposed for the SemEval-2022 Task 4 on Patronizing and Condescending Language Detection (PCL). It obtained an average level of performance, well above the performance of a system that tries to guess without using any knowledge about the task, but much lower than the best teams. To facilitate the interpretation of the performance scores, the F1 measure, the best level of performance of a system that tries to guess without using any knowledge is calculated and used to correct the F1 scores in the manner of a Kappa. As the proposed model is very similar to the one that performed well on a task requiring to automatically identify hate speech and offensive content, this paper confirms the difficulty of PCL detection.

1 Introduction

This paper presents the SATLab's participation in SemEval-2022 Task 4: Patronizing and Condescending Language Detection. It is a very recent task that aims at identifying passages in texts in which a person is condescending to a vulnerable community (Pérez-Almendros et al., 2020; Wang and Potts, 2019). The task organizers have proposed to identify this type of discourse in paragraphs extracted from news stories published in English-speaking media.

The Patronizing and Condescending Language (PCL) Detection task is part of the large family of tasks that aim at automatically identifying problematic language, whether in media or on social networks. Examples include the detection of hate speech and offensive content, fakenews and hyperpartisan news (Kiesel et al., 2019). The PCL challenge, however, differs from these cases in one important characteristic. As pointed out by (Pérez-Almendros et al., 2020), using PCL is not always conscious and the intention of its author is often positive. Nevertheless, PCL tends to indirectly demean the community in question and reinforce negative stereotypes about it. It is therefore important to develop procedures to identify it.

Recently, the SATLab performed well on a task requiring to automatically identify hate speech and offensive content in tweets using a classical supervised algorithm only fed with character n-grams (Bestgen, 2021b). It was especially effective for resource-poor languages (e.g., Marathi). For English, its performance ($F1 = 0.782$) was average, but nevertheless relatively close to the best team ($F1 = 0.831$) that used complementary resources, pre-trained embeddings and deep learning approaches. These performances suggest that the proposed approach could be a interesting baseline to evaluate the benefits of these more complex approaches.

The goal of SATLab's participation in the PCL detection task was to evaluate whether this finding can be generalized to this new task. A priori, one might think that the performance of the system should be significantly worse here for two reasons. First, this task seems to be much more difficult because of the complex language means of expression on which it relies and the considerable amount of world knowledge and common sense reasoning required to understand this type of language (Pérez-Almendros et al., 2020). Secondly, this task is only proposed for English, a language for which many additional resources are available.

The rest of this document presents the task (see Pérez-Almendros et al. (2022) for more details), the proposed system and the performances reached in the challenge.

2 Task

For this challenge, the organizers extracted paragraphs mentioning predefined vulnerable communities, such as *homeless people*, *migrants* or *poor families*, in news stories from various media in different countries, all in English. Each paragraph was annotated according to whether or not it contained one or more instances of PCL. Annotators were also asked to identify the type of PCLs present in a paragraph using seven categories.

On this materials, the SATLab participated in the two tasks proposed by the organizers. The first task consists in deciding whether a paragraph contains some kind of PCL or not. It is thus a classical binary decision task.

The second task is a fine categorization task in seven positive and one negative categories. The seven positive PCL categories are: *a)*[1] *Unbalanced power relations, b) Shallow solution, c) Presupposition, d) Authority voice, e) Metaphor, f) Compassion, g) The poorer, the merrier*. It is important to note that this categorization is non-exclusive: a single instance can be an example of two, three, four and even five categories. I treated this second problem as Task 1 and thus as seven independent binary problems, the predictions of the seven models being simply concatenated in the final submission. For this second task, the organizers also provided the precise position of the text areas that had been identified by the annotators as warranting the assignment of a given category. This information was not used.

The dataset provided by the organizers to develop the systems consisted of 10,469 instances (but one, id = @@16852855, contained no text). It was highly unbalanced for Task 1 with only 9.5% positive instances. The distribution was even more unbalanced in Task 2 since it contained the same proportion of negative cases. The most frequent positive category (a) represented 6.84% of the total instances and the least frequent only 0.38%.

For the system development phase, the organizers proposed a division of the paragraphs into a learning set (80%) and a development set. The test set used in the final evaluation of the systems, whose responses were therefore unknown, consisted of 3,832 paragraphs.

The measure of effectiveness chosen by the organizers was the F1-score on the positive category for Task 1 and the unweighted average of the F1-scores on the seven PCL categories for Task 2.

[1]The letters are used to identify these categories in Table 2 and 4.

Condition	Char	Word	Combi
F1	0.443	0.440	0.468

Table 1: Performance on the development set

3 System

The proposed system is adapted from the SATLab's participation at HASOC 2021. It is based on the following components.

3.1 Features

The only features used were character and word n-grams. These n-grams were extracted from the lowercased paragraphs. The character n-grams had a length between 1 and 7. The extracted word n-grams contained from 1 to 4 words. The tokenization provided in the materials was used. All n-grams present at least twice in the materials were extracted.

3.2 Weighting schema

BM25 (for Best Match 25) was used to weight the frequency of the features in each paragraph (Robertson and Zaragoza, 2009; Bestgen, 2021a). It is a kind of TF-IDF with specific choices for each of the two components, but above all it takes into account the length of the document. Its classical formula is:

$$\text{BM25} = \frac{tf}{tf + k_1 * (1 - b + b * \frac{dl}{dl - avg_{dl}})} \times \log \frac{N - df + 0.5}{df + 0.5} \quad (1)$$

in which tf refers to the frequency of the term in the paragraph, N is the number of paragraphs in the set, df the number of paragraphs that include the term, dl the length of the paragraph and avg_{dl} the average length of the paragraphs in the set. The parameter k_1 was set to 2 and b to 0.75.

3.3 Regularization

The feature values for each paragraph were regularized using the L2 norm.

3.4 Supervised learning procedure

These character and word n-grams were the only features provided to the supervised learning procedure: the L2-regularized logistic regression as implemented in the LIBLinear package (Fan et al., 2008). This procedure is extremely fast and very

Sub	P	T1	T2a	T2b	T2c	T2d	T2e	T2f	T2g
1	C	3.1	4.75	0.95	0.55	0.35	0.25	0.95	0.014
	w1	180	500	1,600	1,300	700	1,250	850	1,400
2	C	2	3.75	0.90	0.70	0.65	0.40	1.45	0.016
	w1	50	300	2,000	1,500	1,400	750	1,750	1,800

Table 2: Parameters for the two submissions for each task

simple to implement because it only requires the optimization of two parameters: the regularization parameter C and $-wi$ which allows to adjust this parameter C for the positive category, the one which has the most influence in the efficiency measure. It should be noted that during the development period of the system, tests were carried out with an approach much slower and much more complex to optimize: a gradient boosting decision tree as implemented in the LightGBM free software (Ke et al., 2017). But, this approach was abandoned because it did not improve the efficiency of the system.

3.5 Comparison to HASOC

The main difference between this system and the one used for HASOC 2021 is the addition of word n-gram. This decision was made during the development phase of the system, carried out on the basis of the division of the materials into a train set and development set as provided by the organizers. Table 1 shows the F1-scores on the positive category for the models based on each n-gram type and their combination. The parameters were optimized directly on the development set, which obviously raises the concern of overfit, but it can be assumed to be similar for each condition compared. As we can see, the two types of n-grams produce very similar performances and the combination brings a small benefit. The addition of word n-grams means that the system can no longer be considered completely language agnostic like the HASOC system, because it relies on the tokenization provided by the organizers and because a number of characters, such as punctuation marks, are removed.

4 Results

The system just described is identical for the two tasks, the only differences being in the parameters of the logistic regression which were optimized independently for each task and for each target category in Task 2. For the two final submissions, these parameters were optimized by a 5-fold cross-validation procedure applied to the combination of the training and development sets using several steps of exhaustive grid search.

Rank	Id	Prec.	Rec.	F1-Score
1	First	0.646	0.656	0.651
43	Baseline	0.394	0.653	0.491
54	SATLab	0.348	0.552	0.427

Table 3: Results for Task 1 (N = 79)

The parameters employed for each submission are given in Table 2. The parameters for the first submission were those that produced the best overall performance in the cross-validation experiments while those for the second submission corresponded to a model producing close performance, but in which the difference between precision and recall for the target categories was as small as possible.

As can be seen in this table, the parameter values are often very different in the two versions while the performance of the models was very close.

Tables 3 and 4 show the performance of the better of the two submissions on the test set, as provided by the organizers. For Task 1, submission 1 performed better while submission 2 performed better for Task 2. The differences between the two submissions for the two tasks are however very small, less than 0.005. These two tables also give the performance of the first team in the challenge as well as that of the Roberta-based Baseline proposed by the organizers.

The proposed system outperformed the Roberta-based Baseline in Task 2, but not in Task 1. It ranked in the middle of the participants at best, very far from the challenge winners. It is unfortunate (for the SATLab at least) that the task was not proposed in languages with less resources and precomputed embeddings, making the use of Deep Learning much more easier.

Rank	Id	a	b	c	d	e	f	g	Mean F1
1	First	0.656	0.529	0.369	0.407	0.359	0.492	0.471	0.469
24	SATLab	0.424	0.331	0.170	0.232	0.175	0.315	0.142	0.256
38	Baseline	0.354	0	0.167	0	0	0.209	0	0.104

Table 4: Results for Task 2 (N = 49)

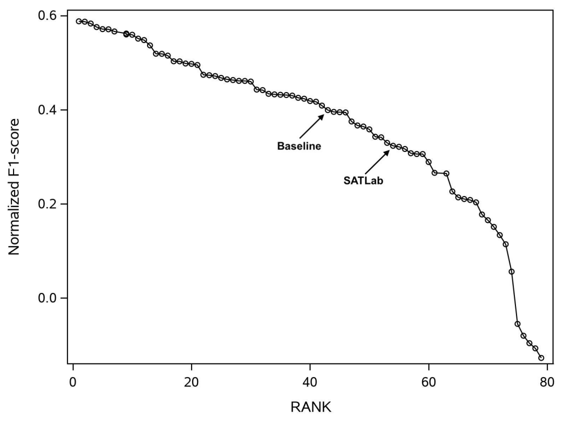


Figure 1: Corrected F1 for all teams in Task 1.

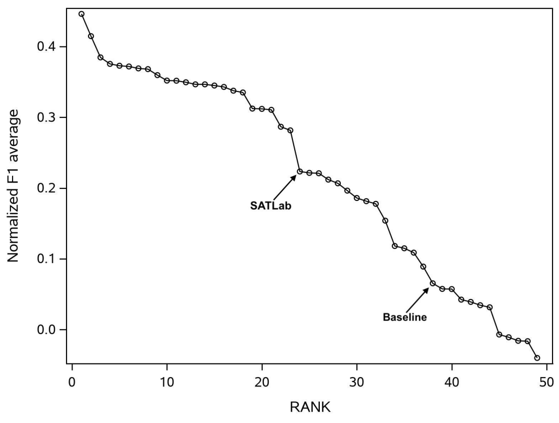


Figure 2: Corrected F1 for all teams in Task 2.

It is interesting to note that the performances obtained on the test set is only slightly lower than those obtained when optimizing the parameters by a cross-validation procedure on the training set. For task 1, this performance on the training set was only 0.03 higher and, on task 2, only 0.008. This suggests that the optimization did not produce an overfit.

The comparison of the system's performance to that of the best team for the seven categories (Table 4) shows that for none of the categories does the system manage to come close to this benchmark. The differences are always approximately 0.20, except *a) Unbalanced power relations*, which is by far the most frequent category, and for *g) The poorer, the merrier*, which is by far the rarest in the learning set, categories for which they are even higher. The more than limited effectiveness of the system does not seem to justify a detailed analysis of its errors.

While the F1-score has become the standard of evaluation for NLP categorization tasks, its interpretation is not obvious. One may wonder whether a Mean F1 of 0.256 (Task 2) represents a prediction of at least an acceptable value. A priori, this does not seem to be the case. On the other hand, the large differences in the frequency of the categories and the imbalance in favor of the negative category may explain the relatively low F1. One way to answer this question more objectively is to determine the best level of performance a system that tries to guess without using any knowledge about the task (Best Guess) can expect to achieve. Ansgar Grüne in his blog post available at https://inside.getyourguide.com/blog/2020/9/30/what-makes-a-good-f1-score shows[2] that in the case of a binary task this is the F1 obtained by a system that always predicts the positive category. If q is the actual proportion of instances belonging to this positive category,

$$\text{F1 Best Guess} = \frac{2q}{q+1}. \qquad (2)$$

This value can also be obtained by submitting the responses of a system that always predicts the positive category, the approach I used here since the challenge participants still ignore the proportion of each category in the test set. In the present challenge, this F1 Best Guess is 0.153 for Task 1 and 0.041 for Task 2. A system that would not exceed these values therefore does no better than a system that always predicts the positive category.

[2] I confirmed empirically this analysis on the task materials by means of a Monte-Carlo procedure in which the proportion of instances in the positive categories were varied between 0 and 1.

SATLab's performance is five times better than this baseline for Task 2, but less than three times better for Task 1. In absolute value, the gain compared to the Best Guess is 0.274 for Task 1 and 0.205 for Task 2. So the n-grams bring some information, but it is obvious that it is not enough to perform well in this task.

This best guest can also be used to correct participants' scores on the task using a formula derived from the Kappa coefficient proposed by Cohen (1960) to adjust the degree of agreement between two judges for agreement due to chance alone. The formula for this correction is:

$$\text{Normalized F1} = \frac{\text{System F1} - \text{F1 Best Guess}}{1 - \text{F1 Best Guess}}. \tag{3}$$

Figures 1 and 2 show the normalized performance of all teams on both tasks. The best system for task 1 achieved a corrected F1 slightly below 0.60. It thus performed 60 % of the way between doing nothing (i.e., the Best Guess) and a perfect performance. This again highlights the complexity of the task. The normalized F1 of the SATLab is only slightly higher than 0.30. For task 2, all the corrected performances are even worse. Figure 2 shows also that the SATLab normalized F1 is quite far from the team directly ahead of it.

5 Conclusion

The SATLab's proposed system for SemEval-2022 Task 4: Patronizing and Condescending Language Detection relies solely on the character and word n-grams present in the paragraphs to be categorized. It does not use any additional data and employs a classical supervised learning procedure (i.e., logistic regression). It obtained an average level of performance, well above the performance of a system that tries to guess without using any knowledge about the task, but much lower than the best teams.

Compared to the performance obtained in the HASOC task (Bestgen, 2021b) and in the detection of hyperpartisan news articles (Bestgen, 2019), this system is clearly further from these best teams. These results confirm, if it were necessary, the much greater difficulty of PCL detection compared to hate speech and offensive content identification (Pérez-Almendros et al., 2020, 2022). They thus suggest, unless other teams using a similar approach were more successful, that the use of much more complex approaches is essential for the PCL task.

Acknowledgements

The author wishes to thank the organizers of this shared task for putting together this valuable event and for their availability throughout the task. He is a Research Associate of the Fonds de la Recherche Scientifique - FNRS (Fédération Wallonie Bruxelles de Belgique).

References

Yves Bestgen. 2019. Tintin at SemEval-2019 task 4: Detecting hyperpartisan news article with only simple tokens. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1062–1066, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Yves Bestgen. 2021a. Optimizing a supervised classifier for a difficult language identification problem. In *Proceedings of the Eigth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 96–101.

Yves Bestgen. 2021b. A simple language-agnostic yet strong baseline system for hate speech and offensive content identification. In *Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation*, CEUR Workshop Proceedings. CEUR-WS.org.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 Task 4: Hyperpartisan News Detection. In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context.

Taygete at SemEval-2022 Task 4: RoBERTa based models for detecting Patronising and Condescending Language

Jayant Chhillar
C2FO
Noida, India
IIIT-Delhi
Delhi, India
`jayant17154@iiitd.ac.in`

Abstract

This work describes the development of different models to detect patronising and condescending language within extracts of news articles as part of the SemEval 2022 competition (Task-4). This work explores different models based on the pre-trained RoBERTa language model coupled with LSTM and CNN layers. The best models achieved 15th rank with an F1-score of 0.5924 for subtask-A and 12th in subtask-B with a macro-F1 score of 0.3763.

1 Introduction

The use of Patronising and Condescending Language (PCL) in text or speech can affect healthy communication channels adversely. The effect of PCL on the vulnerable sections of society have been widely studied. PCL acts as a catalyst for discriminatory behaviour (Mendelsohn et al., 2020) against various vulnerable groups. It has been observed to promote exclusion and discrimination among communities and provide a conducive environment for rumour spreading and misinformation (Nolan and Mikami, 2013). These negative effects of PCL are unaffected by the intent of the writer/speaker who might have unknowingly used PCL. These reasons provide a strong argument for developing methods that can identify and prevent unwanted use of PCL in news articles, blogs, and other pieces of text.

In subtask-A of the Patronizing and Condescending Language Detection task at Semeval-2022 (Pérez-Almendros et al., 2022), the goal is to develop a model which takes a sample text as an input and outputs a label indicating the presence or absence of PCL. In subtask-B, the model was required to identify the correct set of PCL categories. The model takes in a sample text as an input and return seven separate outputs each indicating the presence or absence of the pre-defined seven categories. The dataset for the task was shared by the task organisers in the English language. To tackle these tasks, RoBERTa based models were developed. Different variations of the models involved the use of feed-forward layers, LSTMs, CNN and their combinations. For subtask-A RoBERTa with LSTM, CNN and feed-forward layers outperformed all the other variations with an F1 score of 0.5924. In subtask-B RoBERTa with feed-forward layers got the best F1 score of 0.3763 as compared to the other variations. For subtask-A, this work achieved 15th rank in the leader board and 12th rank for subtask-B details of which are discussed in section 3.2.

2 Background

Identification and analysis of PCL in text is well explored in linguistics (Aggarwal and Zhai, 2012), politics (Huckin, 2002), sociolinguistics (Thapar-Björkert et al., 2016) and other fields. However, in NLP it is still heavily unexplored and starting to gain traction. In the past topics such as sentiment analysis (Feldman, 2013), offensive speech identification (Safaya et al., 2020) and fake news identification (Shu et al., 2017) have been significantly worked upon. One major roadblock in exploring PCL in the text is the lack of well structured and labelled dataset. Recently, some new work has been developed to tackle this issue. Wang et al. (Wang and Potts, 2019) developed a model for identifying the condescending language in Reddit threads and also developed an annotated dataset for the same.

2.1 Dataset

For training and development of the model presented in this work "The Don't Patronize Me!" dataset (Perez-Almendros et al., 2020) was used. The dataset contains paragraphs in the English language extracted from the News on Web (NoW) corpus. It comprises 10469 samples out of which 993 have been classified as positive samples, i.e. they contain PCL. The dataset categorizes PCL into 7 different sub-categories, namely, Unbalanced power relations (UPR), Shallow solution

Sentence	Keyword	Label
In September , Major Nottle set off on foot from Melbourne to Canberra to plead for a national solution to the homeless problem .	homeless	1
10:41am - Parents of children who died must get compensation , free medicine must be provided to poor families across UP : Ram Gopal Yadav	poor-families	1
Today , homeless women are still searching for the same thing . A place to sleep and be safe	homeless	0
For refugees begging for new life , Christmas sentiment is a luxury most of them could n't afford to expect under shadow of long-running conflicts	refugee	0

Table 1: Sample text and keyword pairs along with corresponding labels.

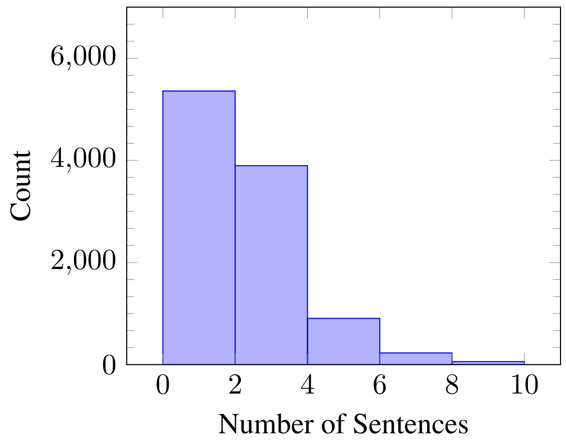


Figure 1: Distribution of the number of sentences per sample.

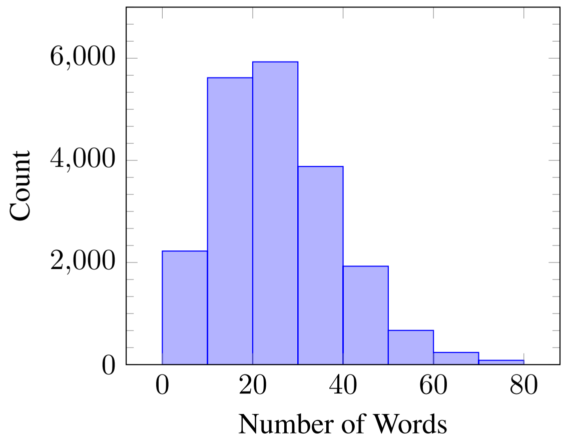


Figure 2: Distribution of the number of words per sentence in a sample.

(SSL), Presupposition (PS), Authority voice (AV), Metaphor (MTP), Compassion (CMP), The poorer - the merrier (PM). Each positive sample can belong to any combination of these categories. The distribution of each category out of all the positive samples is described in Figure 4.

For developing the models for subtask-A the dataset also provides binary labels (0 or 1) to signify the presence or absence of PCL in the text. Along with the paragraphs, the dataset includes the country of origin of the original article and keywords that occur in the paragraph under consideration. These keywords comprise the following, Disabled, Homeless, Hopeless, Immigrant, In need, Migrant, Poor Families, Refugee, Vulnerable and Women. These keywords are usually present in texts that concern the vulnerable sections of society (refer Table 1).

3 System Overview

This section describes the different model designs explored for Task A and Task B and the pre-processing techniques employed. Section 3.1 describes the pre-processing techniques and how they tackle the challenges offered by the dataset. The different models are described under section 3.2 along with a description of the different sub-components and the underlying intuition.

3.1 Data pre-processing

The "The Don't Patronize Me!" dataset offers primarily three major challenges, which are, low number of samples, high class imbalance and the low context in the textual data (smaller sentence length). To deal with high class-imbalance and lower number of samples data augmentation techniques, loss weighting strategies were adopted and to address the low context issue, keywords shared in the dataset were used to provide added context to the models.

3.1.1 Tokenisation

Each sample was tokenised using RoBERTa (Liu et al., 2019) tokenizer. To identify the optimal Tokenisation length analysis was done on the distribution of the number of a sentence per sample (Figure 1) and the distribution of the number of words for

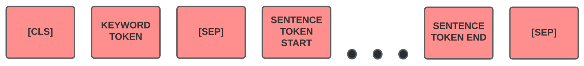


Figure 3: An example of tokenisation

each sentence (Figure 2). On analysing the two distributions length of 50 to 60 tokens seemed a viable candidate for Tokenisation operation. However, on further analysis, it was found that out of 993 positive samples, 193 (19.43 %) had more than 75 words. Thus, to prevent loss of information Tokenisation was done with a length of 100.

Each tokenised sentence was prepended with a tokenised keyword corresponding to that sample separated by the SEP token. Finally, the Tokenisation process was completed by adding a CLS and SEP token at the beginning and the end respectively (refer Figure 3).

3.1.2 Data augmentation

For data augmentation back-translation method was explored. Back-translation is the process of using a language model to translate a text from its parent language to another language, generally using a language model. The new text is then translated back to its parent language. This method introduces slight changes in the structures of the text while retaining the underlying context. This method has been shown to boost the performance of models trained over smaller datasets.(Sennrich et al., 2016). Helsinki-NLP models [1] were used to translate a sentence from English to French and back to English. Only 30 per cent (randomly sampled) of the positive samples from the dataset were back-translated.

3.1.3 Loss weighting

Initial exploratory analysis of the dataset has shown high class imbalance. To address this issue cost-sensitive re-weighting technique developed by Cui et al (Cui et al., 2019) and suggested by Jurkiewicz et al (Jurkiewicz et al., 2020) was adopted. The weighting factor for each class was identified as per the following definition:

$$(1-\beta)/(1-\beta^{n_i}) \tag{1}$$

where β is a hyper-parameter in [0,1), and n_i is the number of samples belonging to the class i. Using these weights the updated softmax cross-entropy loss is given as:

[1]https://huggingface.co/Helsinki-NLP

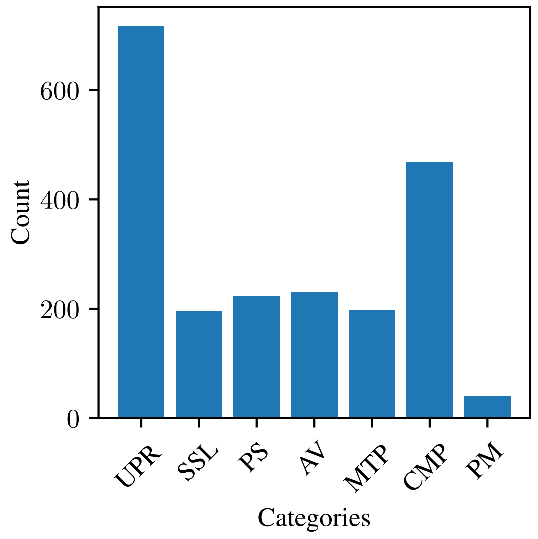


Figure 4: Number of samples for each of the seven PCL classes

Model	BASIC	AUG	WT
RB-FNN	**0.6177**	0.6301	0.6080
RB-BiLSTM	0.6140	**0.6305**	0.6258
RB-CNN	0.5879	0.5954	0.6037
RB-BLS-CNN	0.6059	0.6095	**0.6318**

Table 2: Analysis of the models trained under WT, AUG and BASIC setting for subtask-A

$$L(z,y)=\frac{1-\beta}{1-\beta^{n_i}}\log\left(\frac{\exp(z_y)}{\sum_{j=1}^{C}\exp(z_j)}\right) \tag{2}$$

where $z=[z_1,z_2,...,z_C]$ is the predicted output of the model for C classes and y being one of the possible class labels, i.e. $y \in C$

3.2 Model description

This work explores four different model designs. Each design includes RoBERTaLARGE (Liu et al., 2019) as it's base layer. The output of the last hidden state (shape = 106 X 1024) is then further fed down the network to get the final prediction. For subtask-A, all the models perform binary prediction (0 = no PCL, 1 = contains PCL) to identify if the input text contains PCL, while for subtask-B each model produces 7 binary predictions, one for each possible PCL category. The design of the four models remains the same for both tasks except for the number of outputs generated by them (Figure 5). Binary Cross-Entropy (BCE) loss was used for

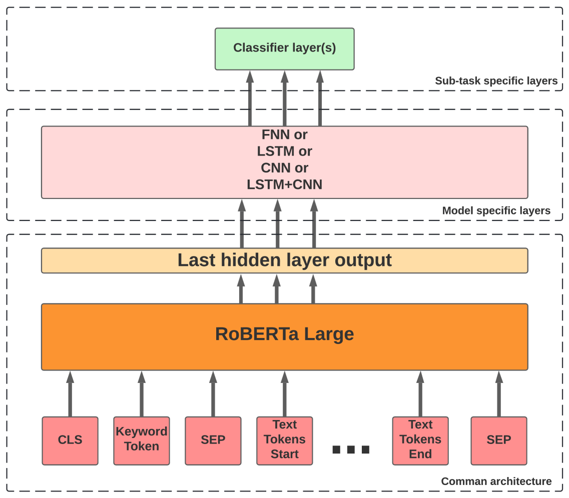


Figure 5: Generalised architecture of the models developed. For subtask-A classifier layers consist of single FNN with 2 units. For subtask-B classifier layers consist of 7 FNN layers each with 2 units.

all the outputs. Adam optimizer was utilized with a learning rate set to 1e-6 and epsilon at 1e-6.

3.2.1 RB-FNN

The model employs the use of two feed-forward layers added on top of $\text{RoBERTa}_{\text{LARGE}}$. The output of the last hidden layer is flattened and passed down the model. The initial feed-forward layer has 106 units. For subtask-A, the output of this hidden layer is passed on to a single feed-forward layer with 2 units for binary prediction, while for subtask-B the output is shared by seven feed-forward layers each with 2 units predicting the presence of each sub-category of PCL.

3.2.2 RB-BiLSTM

LSTM is a type of recurrent neural network (RNN) that allows the model to learn underlying features in temporal data without the added drawbacks of general RNN models such as exploding or vanishing gradients. LSTM allows the model to capture the long term dependencies in the data and identify the underlying temporal nature of the data(Tang et al., 2015). LSTMs have shown to achieve state of the art performance in different text classifications tasks (Tang et al., 2015) and (Li et al., 2020). Shi and Lin (Shi and Lin, 2019) also showed that using LSTM coupled with BERT can improve the performance compared to BERT by itself. For this model the output of the last hidden layer of $\text{RoBERTa}_{\text{LARGE}}$ model is fed into a Bi-Directional LSTM layer with 106 units. The output of the BiLSTM layer is then fed down to two FNN layers with 106 and 2 units respectively (subtask-A). For subtask-B, the output of the first FNN layer is fed to seven feed-forward layers each with 2 units.

3.2.3 RB-CNN

CNN based models have been shown to perform well for various text classification problems (Chen, 2015) (Safaya et al., 2020). CNN layers are able to capture the semantic relationships within the textual data and given the structured nature of the embeddings obtained from $\text{RoBERTa}_{\text{LARGE}}$ model it seemed beneficial to use CNN layers to extract the hierarchical features within the data (Rodrigues Makiuchi et al., 2019). In this model, the last layer embeddings of the $\text{RoBERTa}_{\text{LARGE}}$ model are fed to two CNN layers coupled with a max-pooling layer. The first CNN layer comprises 64 10X10 filters with stride 1 and the second layer comprises 32 5X5 filters with stride 1. After each CNN layer, a two-dimensional max-pooling operation is done with a shape of 2X2. The output of the last max-pooling operation is fed to an FNN layer with 106 units which is followed by an FNN layer with 2 units (subtask-A). For subtask-B, the output of this FNN layer is fed to seven feed-forward layers each with 2 units.

3.2.4 RB-BLS-CNN

To get the model to learn both temporal and hierarchical features within the data a hybrid model was developed employing both LSTM and CNN layers. This model is created as an amalgamation of the RB-BiLSTM and RB-CNN models. The last layer $\text{RoBERTa}_{\text{LARGE}}$ embeddings are fed to an LSTM layer with 106 units. The output of the LSTM layer is then further fed to the CNN architecture defined in the RB-CNN model. The final FNN layer with 106 units is then further fed to a single FNN layer with 2 units for subtask-A and to seven separate FNN layers each with 2 units for subtask-B.

4 Experimental setup

To gauge the effect of data augmentation and loss weighting techniques on the performance of models for each subtask four experiments were carried out (Table 6). The goal was to identify how both the techniques interacted with each other and to find the right combination for each subtask. For each experiment, the model was trained on 80 per cent of the data as the training set and 20 per cent

Model	Macro F1	UPR F1	SSL F1	PS F1	AV F1	MTP F1	CMP F1	PM F1
Baseline	0.1041	0.3535	0	0.1667	0	0	0.2087	0
RB-FNN	**0.3763**	**0.5969**	**0.4578**	**0.3333**	**0.2178**	**0.3043**	**0.536**	**0.1875**

Table 3: F1 score comparison on evaluation dataset for subtask-B between the RoBERTa baseline shared by task organisers and the RB-FNN under AUG experimental settings.

Model	F1 score	Precision	Recall
Baseline	0.4911	0.3935	0.653
RB-BLS-CNN	**0.5924**	**0.5357**	**0.6625**

Table 4: F1 score comparison on evaluation dataset for subtask-A between the RoBERTa baseline shared by task organisers and the RB-BLS-CNN under WT experimental settings.

Exp	Augment	Loss Weighting
BASIC	No	No
AUG	Yes	No
WT	No	Yes
AUG+WT	Yes	Yes

Table 6: Different experiments carried out on each model.

Model	BASIC	AUG	WT
RB-FNN	**0.4054**	**0.4082**	0.3158
RB-BiLSTM	0.3643	0.3818	0.2880
RB-CNN	0.3594	0.3903	**0.3180**
RB-BLS-CNN	0.3599	0.3519	0.2871

Table 5: Analysis of the models trained under WT, AUG and BASIC setting for subtask-B

as the validation set. The 80-20 split shared by the task organisers was used. F1 score for subtask-A and macro F1 score for subtask-B were chosen by the task organisers as the criteria to identify the best performing model, thus the same was used to evaluate the performance of different models created for the two subtasks under different experimental settings. For each experiment, training was done for 20 epochs with a batch size of 8. The best version of the model from each experiment was used to generate predictions for the evaluation dataset.

5 Results

For subtask-A RB-BLS-CNN under WT experiment achieved the highest F1 score of 0.5924 with a precision of 0.5357 and recall of 0.6625 on the evaluation dataset. While on the validation dataset the same model received an F1 score of 0.6318 with a precision of 0.5685 and recall of 0.7109.

For subtask-B RB-FNN performed best out of all the models under the AUG experimental settings. The model achieved a macro F1 score of 0.4006 on the validation dataset and 0.3763 on the evaluation dataset.

The minute difference in the F1 scores of the best models for the evaluation dataset and the validation dataset shows that the model did not overfit during the training phase despite a large number of training epochs.

The effect on class re-weighting (refer 3.1.3) and data augmentation was also explored (refer Table 2 and Table 5). It was found that for subtask-A a majority of the four models received a boost in the F1 score when class re-weighting was applied as compared to the BASIC experimental setting. However, this trend was absent for all the models of subtask-B. Rather class-weighting had a detrimental effect on the models for subtask-B as shown in Table 5. The low number of samples for each of the seven sub-classes coupled with the added complexity of the task as compared to subtask-A could have been the underlying cause behind this observation.

Similarly, the effect of data augmentation on model performance was also explored (refer to Table 2 and Table 5). For subtask-A, all the different models received a boost as compared to the models without augmented data. The same trend persisted for the majority of models trained for subtask-B.

Another interesting find in subtask-B was the significantly poor performance of the LSTM and CNN based models as compared to the vanilla RoBERTa model i.e. RB-FNN. This is not in line with the trend observed for the models in subtask-A (Table 2). The reason for this result could be similar to the unexpected trend observed for the models of subtask-B in class re-weighting experiments. Especially for LSTM based models as a large number of samples are required to train models that employ LSTMs in their design.

	UPR	SSL	PS	AV	MTP	CMP	PM
Precision	**0.6076**	0.3684	**0.5667**	**0.3428**	**0.5652**	**0.6521**	**0.6666**
Recall	0.5563	**0.3889**	0.2741	0.3157	0.25	0.4245	0.1818

Table 7: Precision and Recall values for RB-FNN model under AUG experimental settings on test data.

6 Conclusion

This work explored the design and training of different RoBERTa based models for PCL detection in text. The added benefits of using CNN and LSTM layers along with RoBERTa in boosting model performance was also shown. This work also explored the effects of using back translation as a data augmentation technique along with a class re-weighting technique to deal with low sample size and high class imbalance. Finally, the challenges offered by the models under different problem statements were explored which gives a deeper insight into the impacts of different design methodologies. The best models achieved 15th and 12th rank for subtask-A and subtask-B respectively.

Future work can include expanding the dataset with more data as the current dataset includes 10469 samples. Also, the original article can be provided against each sample which can be fed to the model as added context. This added context should significantly boost model performance.

7 Acknowledgments

The author is grateful for the support of Dr Rajiv Ratn Shah and for the guidance offered by Aradhya Mathur. The author would also like to thank the anonymous reviewers for their insightful feedback.

References

Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer.

Yahui Chen. 2015. Convolutional neural network for sentence classification. Master's thesis, University of Waterloo.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277.

Ronen Feldman. 2013. Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. *Discourse studies in composition*, 155:176.

Dawid Jurkiewicz, Łukasz Borchmann, Izabela Kosmala, and Filip Graliński. 2020. Applicaai at semeval-2020 task 11: On roberta-crf, span cls and whether self-training helps them. *arXiv preprint arXiv:2005.07934*.

Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S Yu, and Lifang He. 2020. A survey on text classification: From shallow to deep learning. *arXiv preprint arXiv:2008.00364*.

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

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A framework for the computational linguistic analysis of dehumanization. *Frontiers in artificial intelligence*, 3:55.

David Nolan and Akina Mikami. 2013. 'the things that we have to do': Ethics and instrumentality in humanitarian communication. *Global Media and Communication*, 9(1):53–70.

Carla Perez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mariana Rodrigues Makiuchi, Tifani Warnita, Kuniaki Uto, and Koichi Shinoda. 2019. Multimodal fusion of bert-cnn and gated cnn representations for depression detection. In *Proceedings of the 9th International on Audio/Visual Emotion Challenge and Workshop*, pages 55–63.

Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. 2020. KUISAIL at SemEval-2020 task 12: BERT-CNN for offensive speech identification in social media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2054–2059, Barcelona (online). International Committee for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, 19(1):22–36.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Suruchi Thapar-Björkert, Lotta Samelius, and Gurchathen S Sanghera. 2016. Exploring symbolic violence in the everyday: misrecognition, condescension, consent and complicity. *Feminist review*, 112(1):144–162.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. *arXiv preprint arXiv:1909.11272*.

CS/NLP at SemEval-2022 Task 4: Effective Data Augmentation Methods for Patronizing Language Detection and Multi-label Classification with RoBERTa and GPT3

Daniel Saeedi
University of Tehran
saeedi.danial@ut.ac.ir

Sirwe Saeedi
Western Michigan University
sirwe.saeedi@wmich.edu

Aliakbar Panahi
C3 AI
ali.panahi@c3.ai

Alvis C.M. Fong
Western Michigan University
alvis.fong@wmich.edu

Abstract

This paper presents a combination of data augmentation methods to boost the performance of state-of-the-art transformer-based language models for Patronizing and Condescending Language (PCL) detection and multi-label PCL classification tasks. These tasks are inherently different from sentiment analysis because positive/negative hidden attitudes in the context will not necessarily be considered positive/negative for PCL tasks. The oblation study observes that the imbalance degree of PCL dataset is in the extreme range. This paper presents a modified version of sentence paraphrasing deep learning model (PEGASUS) to tackle the limitation of maximum sequence length. The proposed algorithm has no specific maximum input length to paraphrase sequences. Our augmented under-represented class of annotated data achieved competitive results among top-16 SemEval-2022 participants. This paper's approaches rely on fine-tuning pretrained *RoBERTa* and *GPT3* models such as *Davinci* and *Curie* engines with extra-enriched PCL dataset. Furthermore, we discuss Few-Shot learning technique to overcome the limitation of low-resource NLP problems.[1]

Keywords: Natural Language Processing, Transformers, Data Augmentation, RoBERTa, GPT-3, Curie and Davinci Engines.

1 Introduction

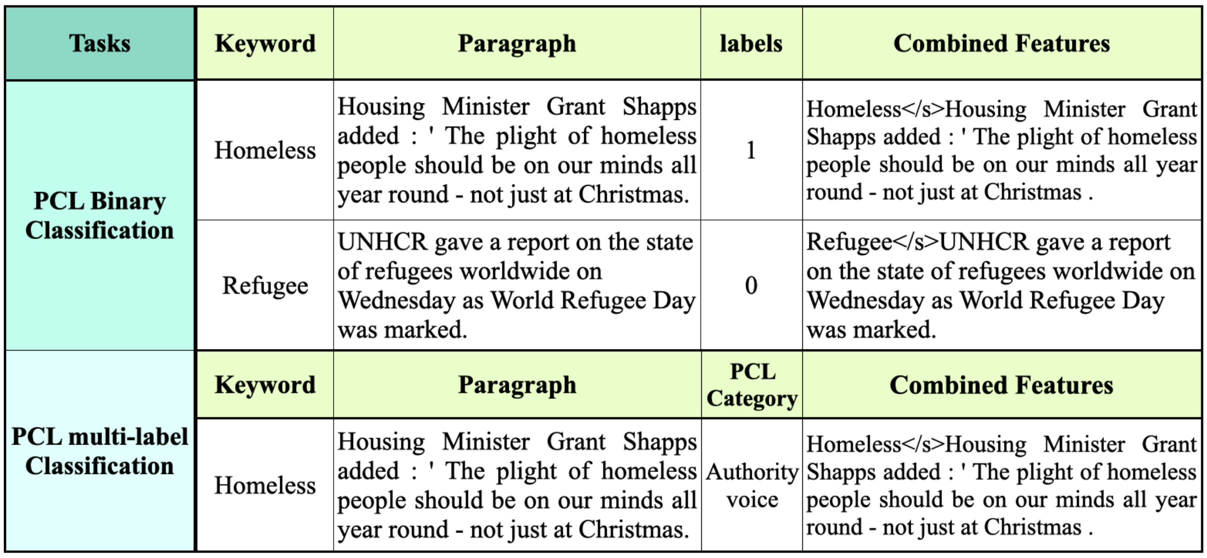
Natural Language Understanding (NLU) and Interpretation (NLI) is a branch of Natural Language Processing (NLP) in Artificial Intelligence (AI), which involves understanding and analyzing human language in-depth. Recent advances in Deep Neural Networks (DNNs) have enabled NLP research scientists to achieve state-of-the-art results for tasks that were extremely difficult, if not impossible Devlin et al. (2019), Lan et al. (2020). However, understanding human emotions, reactions, and uncovering hidden insights from unstructured text data such as news stories channel is still challenging.

Language attitudes and intentions extracting in response to the support for the marginalized and vulnerable communities is one of the emergent NLP applications. Patronizing and condescending language (PCL) is a type of behavior that projects a sense of superiority to vulnerable populations Pérez-Almendros et al. (2020). Furthermore, biases and discrimination can result from patronizing attitudes, causing some people to feel unfairly treated, inadequate, unintelligent, and possibly infuriated Saeedi et al. (2021).

Since raw text data extracting from web is a common data collection method, language models can learn different forms of harmful language Heidari and Jones (2020). The PCL understanding is inherently different from sentiment analysis because positive/negative hidden attitudes in the context will not necessarily be considered positive/negative for PCL tasks. It is difficult due to the fair amount of world knowledge and commonsense reasoning required to understand this kind of language Saeedi et al. (2020). The fine-grained idea of PCL detection towards vulnerable communities was presented by Pérez-Almendros et al. (2022). They evaluated baseline results of NLP techniques to detect the presence of PCL and classify PCL types at the text span level.

In this paper, we describe systems participating in the SemEval-2022, PCL detection competition, multiple tasks of language interpretation. The competition is divided into binary classification and multi-label categorization tasks. Data quality analysis led us to explore several NLP data augmentation techniques and state-of-the-art DNN architectures for these challenging tasks. Our attempts to improve the performance of previous

[1]Our implementation is publicly available at https://github.com/daniel-saeedi/PCL_Detection_SemEval2022



Tasks	Keyword	Paragraph	labels	Combined Features
PCL Binary Classification	Homeless	Housing Minister Grant Shapps added : ' The plight of homeless people should be on our minds all year round - not just at Christmas.	1	Homeless</s>Housing Minister Grant Shapps added : ' The plight of homeless people should be on our minds all year round - not just at Christmas .
	Refugee	UNHCR gave a report on the state of refugees worldwide on Wednesday as World Refugee Day was marked.	0	Refugee</s>UNHCR gave a report on the state of refugees worldwide on Wednesday as World Refugee Day was marked.
PCL multi-label Classification	**Keyword**	**Paragraph**	**PCL Category**	**Combined Features**
	Homeless	Housing Minister Grant Shapps added : ' The plight of homeless people should be on our minds all year round - not just at Christmas.	Authority voice	Homeless</s>Housing Minister Grant Shapps added : ' The plight of homeless people should be on our minds all year round - not just at Christmas .

Figure 1: PCL data for binary and multi-label classification problems. Labels 0 and 1 are corresponding to not containing and containing PCL, respectively. "Authority voice" is the PCL category of paragraph. Training model on combined features as the concatenation of keyword and paragraph with RoBERTa separation token "`</s>`".

efforts ranked us 16 among 79 NLP research teams with very competitive results on the PCL detection task. Our system's performance achieved 80% and 58% F1-score on the training and test datasets, respectively. In comparison, the winning system's F1-score was 65%. Also, the in-depth dataset analysis revealed multi-label classification techniques commonly confused in the PCL categorization task.

This paper is organized as follows. In Section 2, we introduce two PCL tasks, an in-depth analysis of their datasets, and the challenges of these tasks. In Section 3, we describe our different strategies to tackle discovered challenges of data quality. Next, we explored text augmentation methods to fine-tune the Transformer-based model for each individual task. In Section 4, we discuss our applied models, the experimental setup for fine-tuning models, and their performance. Finally, we conclude the paper in Section 5.

2 Tasks Definition and Dataset Analysis

As discussed, PCL competition consists of two classification tasks, each focused on the different objectives of PCL towards underprivileged communities. Figure 1 shows samples of data, their salient features, and annotated labels in training set for both tasks. The first task aims to classify a paragraph that contains PCL as an act of appearing kind or helpful but internally feeling superior to others. The second task is the investigation of the text categorization problem, where each PCL-containing paragraph may belong to several PCL categories[2].

2.1 Data Analysis of Binary Classification

For the PCL binary classification task, we had access to 10469 human-labeled paragraphs for training our models. Two annotators consider their disagreement on borderline cases as not containing PCL. Our exploratory data analysis reveals not containing PCL paragraphs with label '0' make up a large proportion of dataset (90.4%), and target class '1' as containing PCL is the minority class.

The imbalance degree of PCL binary classification dataset can be measured in moderate to extreme range Leevy et al. (2018). The highly imbalanced data would be problematic because models are mostly trained on non-PCL data and will not learn enough from the PCL samples. In this case, a non-PCL outcome is almost always predicted by the trained model. Our experiment shows models yield inaccurate results, see Section 4.

To combat imbalanced training data and misleading classification results, we investigate several techniques in Section 3. Furthermore, this problem is highly challenging because the nature of PCL detection is different from other domains, such as hate speech, inappropriate and fake content detection.

[2]Seven categories for different traits of PCL: Unbalanced power relations (unb), Shallow solution (shal), Presupposition (pre), Authority voice (aut), Metaphor (met), Compassion (com), The poorer, The merrier (merr).

Words that might have positive connotations in sentiment analysis will not necessarily be considered positive in PCL.

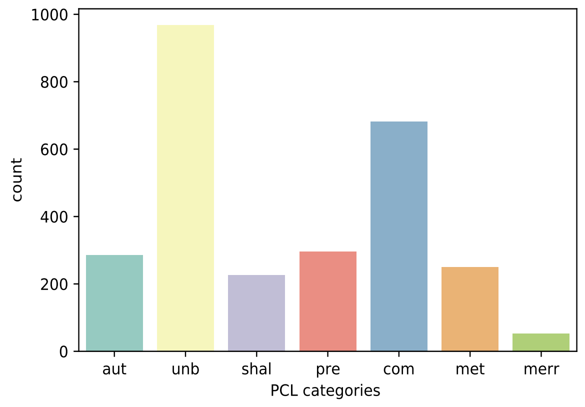


Figure 2: Imbalanced data representation. This chart illustrates the number of observations per PCL category is not equally distributed because in the first task not containing PCL class can obviously discriminate the minority class.

2.2 Data Analysis of Multi-label Classification

For identifying PCL types, the number of manually labeled samples in the datasets is 2760, including all PCL positive data from the previous task. Each text span within the containing PCL paragraph can represent one or more PCL categories.

We were challenged to build a multi-label deep learning model capable of detecting different types of PCL. The unevenly distributed labels, also in the case of multi-label classification, could be problematic. Figure 2 illustrates the number of paragraphs associated with "unb" and "com" are the dominant categories. In Sections 3, we present different methods to combat these challenges, and we describe our efforts of training model on the proper distribution to handle imbalanced dataset in Section 4.

3 Tackling Data Imbalance

Taken together, these challenges led us to approach skewed class proportion problems in the PCL dataset with various Data Augmentation (DA) techniques in NLP Wei and Zou (2019). Like many other NLP techniques, DA is not an exact science, and understanding both dataset and task is essential. We conducted an ablation study to measure the impact of DA on the performance of the system.

We aimed to enhance the size of dataset to reduce the side effect of data imbalance. Before trying text augmentation methods, we preprocessed the data by removing HTML-tags and non-alphabetic characters. Then, we expanded English language contractions, e.g., from "you've" to "you have." The following subsections explain our DA methods.

3.1 Synonym Replacement

Synonym Replacement (SR) is a simple operation that randomly chooses some non-stop words from the sentence and replaces them with one of their synonyms chosen at random. We applied *wordnet* database from *nltk* library to identify synonyms of a given word within the paragraph Miller (1995). As SR is a lightweight and efficient way of performing DA, we tried to replace 1 to 3 words at a time to create diverse PCL samples. Table 1 illustrates scores achieved by training $RoBERTa_{Large}$ model on the augmented dataset. Regardless of the approach taken, the model performance did not spike as expected. As shown later, this approach has been mixed with other text augmentation methods in training models.

3.2 Oversampling

Since containing PCL samples are underrepresented, we considered oversampling (OS) Paduraru and Breaban (2019). Oversampling randomly duplicates data in the minority class by a factor of 8 and adds them to the PCL training dataset, so the number of samples in each class becomes almost equal. The performance of the training pre-trained model with augmented training data by far exceeded the baseline result. (See Table 1)

3.3 Back Translation

We applied back translation (BT) to treat the problem of underrepresented class and boost model performance. In this case, we used a powerful augmenter method of BT in *nlpaug* library and *FairSeqMachineTranslation* Wang et al. (2020) model from *HuggingFace* [3] Transformers. The aim was to generate more PCL samples and then train model on the true distribution. BT translates all PCL samples from English to German, then translates the previously translated text back into the source language. We reused our best-performing model on OS and SR methods. However, later experiments showed that this technique led the model heavily to overfit the augmented training data (Figure 3).

[3] https://huggingface.co/docs/transformers/model_doc/fsmt

Note that we did no model validation using augmented data but did training with a mixture of OS, SR, and BT approaches. Although the improvement offered by BT is not so intelligible, statistical analysis is remarkable. The results are shown in Table 1.

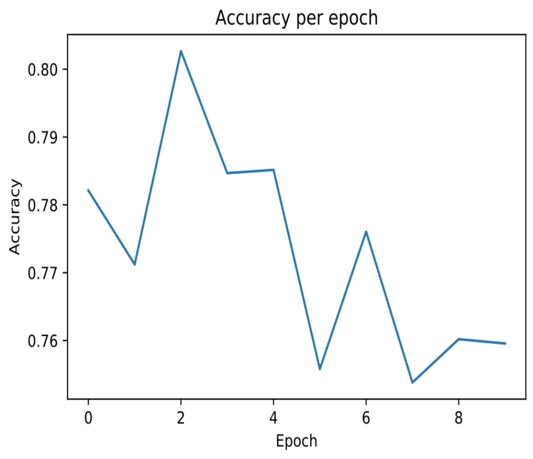


Figure 3: 80% F1-score was achieved at epoch 3. We can see a clear sign of overfitting after this epoch.

3.4 PEGASUS Paraphrasing

Paraphrase generation was the last effort in DA. Paraphrase generation models (in an encoder-decoder form) learn to reconstruct the input using different words and retaining the same meaning while paraphrasing. Paraphrasing can act as a regularizer and reduce the overfitting during the training process Fu et al. (2020).

To leverage PCL dataset efficiently, we performed paragraph paraphrasing along with SR to come up with a less imbalanced dataset. PEGASUS (Pre-training with Extracted Gap-sentences for Abstractive Summarization) Zhang et al. (2020) is a self-supervised Transformer model that masks important sentences from the input and then generates them as one output sequence from the remaining sentences.

The original PEGASUS is limited by the length of text and does truncation on long texts input. The maximum length of PCL paragraphs is 5493 tokens, while the longest input of original PEGASUS model can be 60 tokens. Therefore, we need to handle the limitation of Transformers on the size of the text while training Liu et al. (2019). We proposed an algorithm, multi-sentence PEGASUS, to modify PEGASUS model for arbitrarily long document paraphrasing. This algorithm separates each paragraph into sentences, and then multi-sentence PEGASUS generates ten paraphrased sentences from each individual sentence. The main challenge is to retrieve the original paragraph, because the number of paraphrased sentences for each paragraph was different due to different number of sentences in each sample data. This algorithm can concatenate paraphrased sentences to get the original paragraph in efficient time (The implementation is available at our GitHub). Multi-sentence PEGASUS generates a new dataset containing PCL paragraph over ten times larger than the original containing PCL training data. The following example is a containing PCL data and its corresponding paraphrased context:

Original Paragraph: Shepherding in America has always been an immigrant's job, too dirty, too cold and too lonely for anyone with options.

PEGASUS Paraphrased: In America, shepherding has always been an immigrant's job, dirty, cold, and lonely.

After multi-sentence PEGASUS paraphrasing, two words in each generated text are replaced by their respective synonyms from *wordnet* corpus. The hyper-parameters values for PEGASUS model have been selected by trial and error. We set a number of times the model searches for the most optimal follow-up word within the text to 10 and played with the parameter that regulates the chances of appearance of high/low probability words.

4 Model Description

Our system is based on pre-trained transformers models on the augmented PCL dataset. We focused on exploiting superior performance of *RoBERTa* and *GPT3* models.

4.1 Fine-tuning RoBERTa

To simulate the baseline result, we first did regular fine-tuning *RoBERTa* for each PCL task on the concatenated features of dataset (keyword and paragraph). Submitted systems on the SemEval-2022 leaderboard were evaluated on the F1-score metric. The (73%) F1-score was achieved by training the model with parameter values of $1e-5, 2, 400$ for learning rate, number of epochs, and warm-up steps, respectively while the baseline is 70.63% (See Table 1).

For the next step, we fine-tuned *RoBERTa* on the augmented datasets via each method mentioned

Models	Original Dataset	SR	OS	BT	Peg	SR/Peg	SR/OS/BT/Peg
Task1 $RoBERTa_{Large}$	73%	73%	79%	73%	76%	77%	81%
Task2 $RoBERTa_{Large}$	32%						57%

Hyper-parameters	LR	WS	Epoch
	{1e-5, 5e-6}	{400, 800, 3000, 4000}	{1, 2, 4, 8, 10}

Models	labels	precision	recall	f1-score	support	accuracy
Task1 $GPT3/Curie$	0	61%	22%	32%	50	
	1	52%	86%	65%	50	54%
Task1 $GPT3/Davinci$	0	58%	30%	39%	50	
	1	53%	78%	63%	50	54%

Table 1: Peg stands for PEGASUS paraphrasing. The training *RoBERTa* on the extra enriched dataset (SR/OS/BT/Peg) outperforms other DA methods. The learning process is controlled by setting hyper-parameters (Learning Rate (LR), Warm-up Steps (WS), and Number of Epochs) in the defined range. *GPT3* model with *Davinci* and *Curie* engines yield good performance with small subset of PCL training dataset. Support parameter indicates the number of queries which is the same for both models. 100 queries in total, and 50 queries for each label.

in Section 3, separately. Moreover, we took pre-trained *RoBERTa* and retrained on the extra enriched PCL dataset, which was boosted by a combination of three DA previously explained methods. Same as regular fine-tuning *RoBERTa*, we fed concatenation of keyword and paragraph with $RoBERTa$ special token "</s>" to the model and hyper-parameters are defined in Table 1. Augmented PCL dataset with SR and BT methods led to lower performance of our system compared to a mix of all described DA approaches. Using all DA methods together boosted the model performance to 81%. Figure 3 shows the accuracy of the model in each epoch, and it hits 81% F1-score at epoch 3, and then model start overfitting later. Our system trained and evaluated on the training dataset.

For multi-label classification, the trained *RoBERTa* model on the extra enriched dataset outperformed (57%) the same model trained on the original dataset (32%). However, the model's performance on the test dataset released in the post-evaluation phase was not the same. It is worth mentioning that the F1-average of the winning system (46%) for multi-lable classification task was not better than the random guess model.

4.2 GPT-3 Davinci and Curie

Limitation in the amount of available labeled data can be rectified with Few-Shot Learning technique by providing a few examples at inference time with a large language model. OpenAI *GPT3* Brown et al. (2020) language model uses this technique and also can be applied to PCL binary classification task. *GPT3* has been trained on a huge text dataset from the open internet with billions of parameters.

In this scheme, we considered two offered models of *GPT3* with different capabilities and price points. *Davinci* is the most capable in understanding the intent of a text, the motives of characters, and also the expensive engine. Also, *Curie* is quite faster and lower cost than *Davinci* and capable of tasks like sentiment classification.

We tried both models with Few-Shot learning technique by feeding the model a small amount of PCL training data (with an equal number of labels) as a prompt. The labeled examples were uploaded as a JSON file to OpenAI API for the purpose of classification. *Davinci* and *Curie* leverage a few labeled sets of examples without fine-tuning and enable to understand previously unseen data. We queried the model with a subset of training data to predict the most likely label for each query. In fact, *Davinci* and *Curie* engine classify specified queries using provided labeled data in a JSON file. These engines first search over the labeled data to select the most relevant for a particular query. Our implemented code is publicly available [4].

Table 1 illustrates the performance of *Davinci* and *Curie* models. OpenAI *GPT3* prices are per tokens. Therefore, we just prompted *Davinci* and *Curie* by 1000 and 200 labeled data, respectively. They were evaluated on F1-score with 100 queries of even class distribution. Surprisingly, both models perform well without hyper-parameter tuning and on just a few examples of PCL. *Davinci*'s performance was the same as *Curie*'s result but with

[4]https://github.com/daniel-saeedi/PCL_Detection_SemEval2022

five times fewer labeled examples. OpenAI API offers the ability to fine-tune their model on the desired task, which is quite costly and time-intensive. An interesting future research direction can be exploring GPT3 applications for PCL detection and multi-label classification tasks, regardless of the cost to train the model.

5 Conclusion

This paper presented a system description for PCL detection and multi-label categorization tasks. Our exploratory data analysis revealed annotated PCL dataset is highly imbalanced. We enhanced data quality with a combination of data augmentation methods. We presented a modified version of sentence paraphrasing deep learning model, Multi-sentence PEGASUS, to tackle the limitation of maximum sequence length. The proposed algorithm has no specific maximum input length to paraphrase sequences. We evaluated the performance of the large pre-trained RoBERTa model on the extra enriched PCL dataset. We boosted the baseline performance and achieved competitive results among the top-16 SemEval-2022 participants. Furthermore, we tried two models of GPT3, *Davinci* and *Curie* with Few-Shot learning technique. Our investigation showed both models perform well without hyper-parameter tuning and on just a few examples of PCL. We believe these tasks have many potentials and challenges to further improve current results.

References

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yao Fu, Yansong Feng, and John P. Cunningham. 2020. Paraphrase generation with latent bag of words.

Maryam Heidari and James H Jones. 2020. Using bert to extract topic-independent sentiment features for social media bot detection. In *2020 11th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, pages 0542–0547.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations.

Joffrey L Leevy, Taghi M Khoshgoftaar, Richard A Bauder, and Naeem Seliya. 2018. A survey on addressing high-class imbalance in big data. *Journal of Big Data*, 5(1):1–30.

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

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Cristian Padurariu and Mihaela Elena Breaban. 2019. Dealing with data imbalance in text classification. *Procedia Computer Science*, 159:736–745. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Cheionference KES2019.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Sirwe Saeedi, Saraju P. Mohanty, Steve Carr, Alvis C. M. Fong, and Ajay K. Gupta. 2021. Consumer artificial intelligence mishaps and mitigation strategies. *IEEE Consumer Electronics Magazine*, pages 1–1.

Sirwe Saeedi, Aliakbar Panahi, Seyran Saeedi, and Alvis Cheuk M. Fong. 2020. CS-NLP team at semeval-2020 task 4: Evaluation of state-of-the-artnlp deep learning architectures on commonsense reasoning task. *CoRR*, abs/2006.01205.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. fairseq s2t: Fast speech-to-text modeling with fairseq.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

University of Bucharest Team at Semeval-2022 Task4: Detection and Classification of Patronizing and Condescending Language

Raluca Andreea Gînga, Bogdan Dobre, Tudor-Andrei Dumitraşcu and **Bogdan Radu Silviu Sielecki**
University of Bucharest
`{gingaraluca, dobrebogdan98, tudorandrei.dumitrascu, sieleckiradu}@gmail.com`

Abstract

This paper details our implementations for finding Patronizing and Condescending Language in texts, as part of the SemEval Workshop Task 4. We have used a variety of methods from simple machine learning algorithms applied on bag of words, all the way to BERT models, in order to solve the binary classification and the multi-label multi-class classification.

1 Introduction

The Patronizing and Condescending Language Detection Task (Pérez-Almendros et al., 2022) is based on the paper Don't Patronize Me! (P'erez-Almendros et al., 2020), which is an annotated Dataset with Patronizing and Condescending Language Towards Vulnerable Communities.

The aim of this task is to identify PCL, and to categorize the language used to express it, specifically when referring to communities identified as being vulnerable to unfair treatment in the media.

Participants were provided with sentences in context (paragraphs), extracted from news articles, in which one or several predefined vulnerable communities are mentioned. The challenge is divided into two subtasks.

1. Subtask 1: Binary classification. Given a paragraph, a system must predict whether or not it contains any form of PCL.
2. Subtask 2: Given a paragraph, a system must identify which PCL categories express the condescension. The PCL taxonomy was defined based on previous works on PCL (i.e. Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier.)

2 Background

The dataset used for this SemEval 2022 task was Don't Patronize Me! (P'erez-Almendros et al., 2020), which contains a suite of sentences that mention some vulnerable communities and published in media in a lot of English speaking countries. The paragraphs were manually annotated to show 1) whether the text contains any kind of PCL, and 2) if it contains PCL, what linguistic techniques (categories) are used to express the condescension. The paragraphs, according to (P'erez-Almendros et al., 2020), were extracted from News on Web (NoW) corpus (Davies, 2013), being annotated by three expert annotators, with backgrounds in communication, media and data science.

The dataset for subtask 1 (binary classification) contained a number of 10.636 paragraphs and 2.792 instances were used for the categories classification subtask.

In Figure 1, it can be seen that for the first subtask, there are almost 1000 texts that contain PCL. This means that the dataset is highly imbalanced and this problem needs to be addressed.

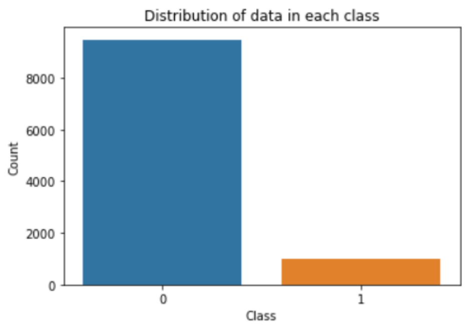


Figure 1: Classes Distribution for Binary Classification problem (Subtask 1)

For task 2, the paragraphs from task 1 are split according to the type of PCL speech category into sentences, resulting in 950 samples.

3 System Overview

1. Subtask 1 (Binary Classification)

   Because the dataset was very imbalanced, we tried different approaches in order to make it balanced:

- Adding a class weight to the models used. In this approach, we computed a metric in which we obtained a class weight according to the imbalance of the dataset. Through this method, we gave some different weights to both the majority and minority classes. This whole process had the purpose to penalize the miss classification made by the minority class by setting a higher class weight and at the same time, reducing the weight for the majority class.
- Using oversampling methods and special ensemble techniques. In this approach, we used methods like SMOTE (Synthetic Minority Over-sampling Technique) (Chawla et al., 2002), Adasyn (Adaptive Synthetic) (He et al., 2008), SVM-SMOTE (Mathew et al., 2015) and SPE (Self-Paced Ensemble) (Liu et al., 2020) that performs strictly balanced under-sampling in each iteration, being very efficient computationally.
- Augmenting the data. Because we notice so little data for label 1, we decided to collect hate speech datasets from Kaggle[1] and add the positive texts into our dataset in order to balance the classes frequency, obtaining a total of 6372 from 795 initial texts with label 1. We will notice in the results section that this collection and generation of new dataset did not provide good results.

The dataset was preprocessed. The preprocessing consisted in: clearing the special characters, lowercasing, tokenization, stopwords removal, removing the words shorter than 3 characters. Then, the resulted (and clean) dataset was split into two preprocessed types: lemmatized cleaned dataset and stemmed cleaned dataset. These two datasets were generated in order to make some comparison between those two techniques and to see which provided the best results.

To extract features from text, we have used TF-IDF (Sammut and Webb, 2010), Keras Tokenizer[2], Word2Vec with Skip-Gram (Mikolov et al., 2013) and, finally, Bert Tokenizer provided by Hugging Face (Wolf et al., 2019).

We have also used a variety of models such as Neural Networks with 3 dense layers, Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) with 64 and 128 neurons with dropout of 0.1 as well, basic Machine Learning algorithms like Logistic Regression, Random Forest, Support Vector Machines as XGBoost. In the end, we decided to try BERT embeddings and a BERT classification model, BertForSequenceClassification [3], that contains a single linear classification layer on top and that provided the best results after all of the other approaches.

Another approach, called "Text shards" made use of the subtask related to multi-class classification as well. For an average text that contains PCL, only some small pieces of them are actually PCL and the rest of the text are not. The assumption is that this confuses the model, because a combination of PCL and non-PCL is labeled as PCL. To address this, the following approach is used:

- negative examples are left as they are
- each positive example is replaced with the actual pieces of PCL inside it that we can get from the categories file
- the positive examples obtained this way are added with the negative examples to obtain a training dataset
- all the sentences are cleaned of characters that are not letters and the words in each sentence are lemmatized
- a Tensorflow Hub pretrained model called Universal Sentence Encoder (Cer et al., 2018) is trained on it
- for each text that we want to predict, we first use the model on the whole text to get an initial label
- a window (of the size of the average length of a cleaned PCL fragment * 2) is slided through the text and the model is used to predict that particular substring. If it is labeled as PCL, then we consider the whole text as PCL.

2. Subtask 2 (Multi-label Multi-class Classification)

[1]Hate Speech datasets
[2]Tokenizer method brought by Keras
[3]Bert for Sequence Classification

Considering the fact that the vocabulary of the is English is large, we have tried to leverage the power of pretrained language models. Therefore we have chosen 3 BERT-based models which were pretrained for hate speech detection and sentiment analysis. The BERT models also provided a tokenizer which split the sentences into tokens and appended the required tokens. The BERT models are used from the transformers library (Wolf et al., 2019).

- BERT (Devlin et al., 2018) Uncased
- BERT Multilingual Uncased
- BERT HateXplain (Mathew et al., 2020): This model was trained to classify text as Hate speech, Offensive or Normal. It was trained on Gab, Twitter and Humain Rationale;
- Distil BERT : This model is a version of Distilled BERT finetuned on the Twitter dataset;
- Distil BERT Multilingual Cased (Sanh et al., 2019)
- Distill RoBERTa : This model is a version of Distilled RoBERTa finetuned on the Twitter dataset;

In the paper describing the dataset (P'erez-Almendros et al., 2020), the authors group the categories into 3 General categories.

(a) The saviour: Unbalanced power relations and Shallow relations
(b) The Expert: Presupposition and Authority voice
(c) The Poet: Compassion, Metaphor and The poorer the merrier

From this idea, we tried to train the models to predict those 3 categories, and save the hidden features to a fixed latent space. Then these learned features can be used when training the model to predict the required 7 sub-classes.

Along with those BERT-based model, we also tried to implement models based on Word2Vec (Mikolov et al., 2013) (trained on "Google News") and Machine Learning algorithms based on TF-IDF and BOW:

- LSTM Word2Vec Embeddings (Staudemeyer and Morris, 2019)
- BiLSTM Word2Vec Embeddings (Huang et al., 2015)
- RNN Word2Vec Embeddings (Sherstinsky, 2020)
- SVM TF-IDF
- RandomForest TF-IDF

We also dabbled with the thought of training our own Word2Vec, in order to create a model specialized on hate speech. However we decided against this idea, due to the lack of usable datasets and the computational resources required for this task.

4 Results

1. Subtask 1 (Binary Classification)

   Since we experimented with various techniques and approaches, we decided to split the results based on the experiments made.

   (a) Deep Learning / Machine Learning for Imbalanced and Oversampled dataset
   In table 1, we can notice the results provided by classical Machine Learning algorithms and 3-layer Neural Networks (512, 256 and 128 layers with ReLU activation and using class weight for providing accurate performance in terms of data imbalance) on 4 types of datasets: the original dataset (without proceeding to class balance, but using class weights for controlling the class weights), Random Forest with Self-Paced Ensemble Bootstrap technique (SPE), SMOTE and SVM-SMOTE.
   Logistic Regression gave solid results on all variations of the datasets, providing an *f1_score* of 0.35 on lemmatized dataset and 0.38 on stemmed validation dataset. Neural Networks provided as well good results, but did not manage to obtain the performance of Logistic Regression. We could infer from the tables that Logistic Regression gave the best performance on stemmed dataset.

   (b) Keras Tokenizer & Word2Vec Embeddings + LSTM neural network
   Another experiment that we conducted was the use of Keras Tokenizer and Word2Vec in order to extract the embeddings from the texts. We then applied

Approach \ Dataset	Simple	SPE	SMOTE	SVM-SMOTE
Neural Networks	0.27	-	0.2823	0.3187
Logistic Regression	**0.34**	-	**0.35**	**0.35**
Random Forest	0.067	0.31	0.19	0.16
Support Vector Machines	0.27	-	0.10	0.14
XGBoost	0.15	-	0.23	0.24

(a) Results on Imbalanced and Oversampled Lemmatized dataset

Approach \ Dataset	Simple	SPE	SMOTE	SVM-SMOTE
Neural Networks	0.2698	-	0.289	0.3166
Logistic Regression	**0.35**	-	**0.38**	**0.37**
Random Forest	0.038	0.31	0.21	0.13
Support Vector Machines	0.27	-	0.14	0.20
XGBoost	0.17	-	0.23	0.24

(b) Results on Imbalanced and Oversampled Stemmed dataset

Table 1: Results on Imbalanced and Oversampled Lemmatized & Stemmed dataset. The results are in terms of *f1_score*.

Approach \ Dataset	Augmented dataset
Neural Networks	0.2155
Logistic Regression	0.23
U.S.E. + 2 dense layers	**0.2316**

Table 2: Results on Augmented dataset.

two LSTM models: one with 64 neurons and the other one with 128 neurons.

The results of these two models on both Lemmatized & Stemmed datasets with two variations of created embeddings (Keras Tokenizer and Word2Vec) are provided in table 3. LSTM with 64 neurons provided best results on the datasets that were using the default Tokenizer from Keras, with an f1_score of almost 27% on Lemmatized dataset and 32& on Stemmed dataset. Word2Vec did not seem to provide good results in combination with LSTM networks.

(c) Data augmentation

As we discussed in the previous section, we augmented the data by using the positive texts from different hate speech datasets from Kaggle and adding to our dataset. We then applied TF-IDF vectorizer with 5000 features and fed the embeddings into a 3-layer Neural Network (512, 256 and 128 neurons) and to a Logistic Regression model. Another method used was Universal Sentence Encoder (U.S.E. annotated in table) + 2 dense layers of 128 and 64 neurons.

The results are present in table 2. We can infer that the third method provided the best results, but still insufficient to reach the level and performance of Logistic Regression from (a).

(d) BERT Transformers + BertForSequenceClassification

For Bert Transformers, we obtained a performance of **0.5074**, the best result provided among all of the other models and techniques. This performance was obtained by using Bert Tokenizer for encoding the entire texts, calculating the class weight and providing it to a BERT-base-uncased model with AdamW as optimizer (learning rate of $2e-5$) and 3 epochs for training. The total training time took 2 hours.

(e) Text shards

For "Text shards" approach, we obtained an F1 score of 0.3117.

Overall, for the first subtask, we obtained the best performance using BERT Transformers and fine-tuning a BERT model with an F1 score of 0.5074. The second best-performing algorithm was, surprisingly, Logistic Regression, that provided 0.38 on SMOTE oversampled dataset.

2. Subtask 2 (Multi-label Multi-class Classification)

(a) BERT models approach for classification across 7 classes. Table 4a shows that the model was able to learn only two of the classes. The best model, DistilBERT, obtains F1 score of 0.34.

Approach \ Dataset	Keras Tokenizer	Word2Vec
LSTM (64 neurons)	**0.2693**	0.2109
LSTM (128 neurons)	**0.2317**	0.2308

(a) Results on Lemmatized dataset with class weight

Approach \ Dataset	Keras Tokenizer	Word2Vec
LSTM (64 neurons)	**0.3213**	**0.2093**
LSTM (128 neurons)	0.2789	0.2412

(b) Results on Stemmed dataset with class weight

Table 3: Results on Lemmatized & Stemmed datasets using Keras Tokenizer and Word2Vec as word embeddings. The results are in terms of *f1_score*.

Model \ Class	Unb	Sha	Pre	Aut	Met	Com	Mer	Mean
BERT	0.82	0.0	0.0	0.0	0.0	0.0	0.64	0.21
DistilRoBERTa	0.83	0.0	0.0	0.0	0.0	0.0	0.59	0.20
DistilBERT	0.82	0.0	0.0	0.0	0.66	0.08	0.0	**0.34**

(a) Results of transformers trained directly on 7 classes

Model \ Sub-classes	*Expert*	Aut	Pre	*Saviour*	Sha	Unb	*Poet*	Com	Mer	Met	Mean
BERT	0.44	0.0	0.0	0.85	0.0	0.84	0.69	0.0	0.0	0.59	0.20
DistilRoBERTa	0.54	0.0	0.0	0.85	0.0	0.84	0.69	0.11	0.0	0.65	0.22
DistilBERT	0.42	0.0	0.40	0.75	0.0	0.81	0.61	0.0	0.0	0.67	**0.26**
DistilBERTMLC	0.36	0.0	0.0	0.86	0.0	0.83	0.60	0.0	0.0	0.52	0.19

(b) Results of transformers that were trained on 3 general classes, then finetuned for the desired 7 classes

Table 4: Transformer Results

(b) The general class approach is detailed in table 4b, where the general classes are italicized. It shows that the general classes were learned, but when using the pretrained models and fine-tuning on the specific classes, some of previously learned features are lost. The best results it obtained yet again by the DistilBERT model with an F1 score of .26.

5 Conclusion

In this paper, we presented our solution to the problem posed by SemEval 2022 Task 4: Patronizing and Condescending Language Detection. We applied various methods, including the application of Word Embedings (Bag of Words, Word2Vec, BERT), tokenization, oversampling/undersampling of the datasets.

In the binary classification problem, the approach that gave the best result on the validation dataset was BERT transformers combined with BERT for Sequence Classification, obtaining 0.50 as F1 score, followed by Logistic Regression applied on stemmed SMOTE dataset with a performance of 0.38.

In the multi classification multi label task, the number of labels proved to be a challenge. The results overall are low and the models were only able to learn only a few classes. The general class approach also proved to be inefficient. Perhaps a more suitable approach would be to build more complex models and use models that do not rely on specific pretrained approaches.

Some recommendations for future work could be to have a better approach and introduce more linguistic insight in the approach.

Acknowledgements

We would like to thank our supervisor Ana-Sabina Uban for the guidance and throughout the development of the project.

References

Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Mark Davies. 2013. Corpus of news on the web (now):

3+ billion words from 20 countries, updated every day.

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

Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging.

Zhining Liu, Wei Cao, Zhifeng Gao, Jiang Bian, Hechang Chen, Yi Chang, and Tie-Yan Liu. 2020. Self-paced ensemble for highly imbalanced massive data classification. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection. *arXiv preprint arXiv:2012.10289*.

Josey Mathew, Ming Luo, Chee Khiang Pang, and Hian Leng Chan. 2015. Kernel-based SMOTE for SVM classification of imbalanced datasets. In *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, pages 001127–001132.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Claude Sammut and Geoffrey I. Webb, editors. 2010. *TF–IDF*, pages 986–987. Springer US, Boston, MA.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Alex Sherstinsky. 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Ralf C. Staudemeyer and Eric Rothstein Morris. 2019. Understanding LSTM – a tutorial into long short-term memory recurrent neural networks.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Amrita_CEN at SemEval-2022 Task 4: Oversampling-based Machine Learning Approach for Detecting Patronizing and Condescending Language

Bichu George, S Adarsh, Nishitkumar Prajapati, Premjith B, and Soman K.P
Centre for Computation Engineering and Networking (CEN)
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham, India
b_premjith@cb.amrita.edu

Abstract

This paper narrates the work of the team Amrita_CEN for the shared task on Patronizing and Condescending Language Detection at SemEval 2022. We implemented machine learning algorithms such as Support Vector Machine (SVM), Logistic regression, Naive Bayes, XG Boost and Random Forest for modelling the tasks. At the same time, we also applied a feature engineering method to solve the class imbalance problem with respect to training data. Among all the models, the logistic regression model outperformed all other models and we have submitted results based upon the same.

1 Introduction

Discriminatory language on the social media is lately creating hostile environment towards the vulnerable communities especially women and minorities. These are reflected in day to day conversations happening on popular social media sites. It is a high time now to build a technological solution to counter the discrimination against vulnerable communities. Here in this task, we consider one such issue known as "Patronizing and Condescending Language (PCL) Detection". When someone's language conveys a pompous attitude toward others or portrays them or their circumstances in a compassionate manner, eliciting feelings of sympathy and compassion, they are patronising or condescending. This is why it is important to develop a computational model to predict whether there is patronizing content in social media or not (Pérez-Almendros et al., 2020). This challenge can be solved by the applying Natural Language Processing (NLP) concepts. The Social media platforms reaches a huge audience, which might contribute to increased exclusion and inequity among vulnerable groups. Despite the fact that harmful language behaviour (such as hate speech, abusive language, fake news, rumour propagation, or disinformation) (Sreelakshmi et al., 2020), (Sreelakshmi et al., 2021) has been extensively investigated in NLP, PCL has remained a neglected field of research.

We implemented seven machine learning models which include three classical machine learning algorithms and four ensemble models: Support Vector Machine (SVM), Logistic regression, Naive Bayes, XG Boost and Random Forest for modelling the tasks (Soman et al., 2009), (Premjith et al., 2019), (Premjith and Kp, 2020). The class imbalance problem was dealt by a minority oversampling technique called SMOTE and comparative analysis of our algorithm was done by various evaluation metrics such as precision, recall and F1 score.

The remaining parts of the paper are described as follows: Section 2 contains dataset description along with works related to that. Section 3 describes the system overview. Section 4 explains the experimental setup. Section 5 discusses result and the paper is concluded in Section 6.

2 Related works

This section provides a brief review of the literature published for the detection of various offensive and abusive contents pertained to violence, cyberbullying etc. shared on the social media.

Adithya et.al (Bohra et al., 2018) analysed the hate speech data in code-mixed form and proposed classification models for the detection. They created a dataset consisting of Hindi-English code-mixed tweets. Machine learning algorithms like SVM, Random forest were used for the classification of tweets into different categories. Conroy et.al (Rubin et al., 2016) reported the problem of fake news detection in their paper and their study offered a classification of different types of truthfulness evaluation methods that fall into two categories: linguistic cue with machine learning and network analysis approaches. Zampieri et al (Zampieri et al., 2019) predicted the nature and victim of offensive content shared on social media. They used the Offensive Language Identification Dataset (OLID) for the analysis. They compared the performance of different machine learning models on this dataset. Wang and Potts (Wang and Potts, 2019) used a corpus called TALKDOWN for detecting the condescension in a text by incorporating the context. The dataset consist of annotated social media messages. They explored the issue of modelling condescension in direct communication from an NLP perspective. They used BERT-based models for developing the baseline models.

3 Task and Data Description

3.1 Task1

The competition mainly consisted of 2 sub tasks (Pérez-Almendros et al., 2022). The objective of the subtask 1 is to develop a model, which could predict whether a given paragraph contain condescension or not, which is a binary classification problem. The dataset used for subtask 1 consists of 10469 paragraphs. Each of the paragraphs describes the people belonging to vulnerable social categories. It contains excerpts from news items from 20 English-speaking nations that feature at least one of the following terms relating to potentially weaker sections of the society: vulnerable or women, refugee, hopeless, migrant, immigrant, in need, homeless, poor families, disabled, with Patronizing and Condescending Language (PCL) comments.

3.2 Task2

The objective of the subtask 2 is to develop a model, which could predict whether a given paragraph comes under any of the top 7 PCL taxonomies namely, Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier, which is a multi-label classification problem. The dataset used for subtask 2 consists of 993 paragraphs. Each of the paragraphs describes the people belonging to vulnerable social categories. It contains excerpts from news items from 20 English-speaking nations that feature at least one of the following terms relating to potentially weaker sections of the society: vulnerable or women, refugee, hopeless, migrant, immigrant, in need, homeless, poor families, disabled, with Patronizing and Condescending Language (PCL) comments.

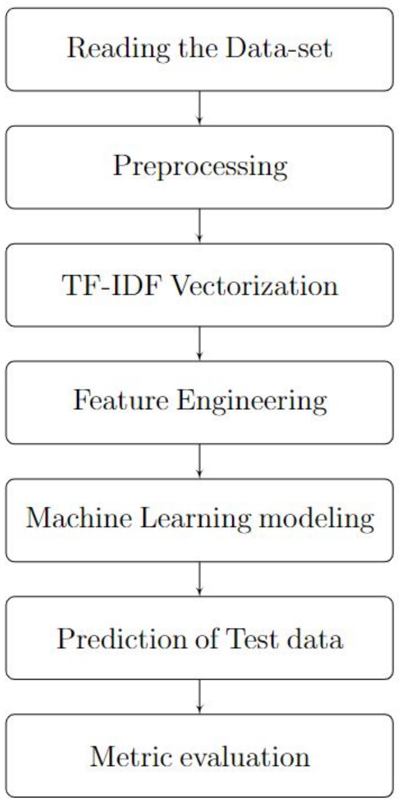


Figure 1: Flowgraph of the methodology

4 System Overview

This section discusses the procedure followed for developing models for each subtasks in completion. Figure 1 represents the block diagram of the workflow of the methodology.

This section explains the steps followed for developing models for the PCL shared tasks.

4.1 Preprocessing

Initially, we cleaned the data by removing stopwords, URLs and special characters. The cleaned texts were tokenized and lemmatized to obtain the root form of the word. It helped to reduce the vocabulary in the corpus, which further reduce the dimension of the sentence vector obtained using Term Frequency-Inverse Document Frequency (TF-IDF) vectorizer algorithm.

4.2 Feature Engineering

We represented the textual data as vectors using TF-IDF for the further processing. In addition to that, we employed SMOTE (SMOTE: Synthetic Minority Over-sampling Technique) (Chawla et al., 2002), an oversampling algorithm to address the problem of class imbalance in the data. The

SMOTE algorithms synthetically generates random data for the minority classes to increase the size of the minority classes. It is done by selecting one or more of random k-nearest neighbour for each minority instances. We employed SMOTE after converting texts into vector using Term Frequency-Inverse Document Frequency vectorizer algorithm.

4.3 Machine Learning modelling

The dataset for subtask 1 consists of total 10469 instances and for subtask 2 it is 993 instances. We considered a train-test split ratio of 80:20. The parameter stratify was used for the purpose of making a split so that the share of values in the sample produced will be the equal to the proportion of values provide to parameter stratify. For prediction, we have a total of 2094 test instances in which 1895 belongs to class 0 and 199 belonging to class 1 in subtask 1 and 198 test instances. For logistic regression model, hyper parameter tuning was done using sklearn's GridSearchCV function [1]. The parameters that was given for tuning was penalty $=l1, l2$ and value of $C = array([0.01, 0.1, 1, 10, 100])$. After hyperparameter tuning using GridSearchCV, the best parameters were found to be $C(regularization\_term) = 10$ and $Penalty = l2$. For subtask 2, we set the class_weight hyperpaameter to be 'Balanced'. To predict the multi-label output, we used the 'MultiOutputClassifier' function from Scikit-learn [2]. For models other than logistic regression, we used default parameters available in Scikit-learn for classification.

4.4 Evaluation

The evaluation measures used for this work were macro average F1,precision and recall. Recall is ratio of correct positive predictions to the total number of positives and Precision is ratio of correct positive predictions to the total number of positive predictions. F1 score is the harmonic mean of precision and recall. Macro average is defined as the average of precision, recall, F1 score on different classes.

5 Results

In both the sub tasks we used three classical ML models and four ensemble techniques for classification. The three ML models were logistic regression ,SVM and DecisionTreeClassifier and the ensemble techniques were Bagging classifier, Random forest, GradientBoost and XGBoost. Validation dataset was used to get a comparative analysis of our algorithm. In this analysis we used evaluation metrics such as precision, recall and F1 score. The official evaluation metric was F1 score for positive class for subtask 1. For the validation dataset an F1 score of 0.41 was achieved for positive class and in case of test dataset an F1 score of 0.39 was obtained and our final rank for subtask 1 in the competition was 60. For subtask 2 we got a macro_average F1 score of 0.45 during the post evaluation phase.

Model	Recall	Precision	F1
Log Reg	0.73	0.64	0.66
SVM	0.53	0.75	0.53
Dec Tree	0.57	0.57	0.57
Bagging	0.54	0.61	0.55
Random For	0.51	0.64	0.49
GradBoost	0.53	0.75	0.54
XGBoost	0.56	0.68	0.58

Table 1: Comparitive analysis of our ML models for subtask 1 considering macro averages

Model	Recall	Precision	F1
Log Reg	0.48	0.45	0.45
SVM	0.30	0.58	0.32
Dec Tree	0.35	0.36	0.35
Bagging	0.29	0.41	0.33
Random For	0.27	0.56	0.31
GradBoost	0.28	0.45	0.32
XGBoost	0.33	0.47	0.36

Table 2: Comparitive analysis of our ML models for subtask 2 considering macro averages

From the Tables 1 and 2 we can clearly see that the macro F1 score of Logistic regression stood out among all the other models. Moreover the execution time for logistic regression was less compared to other models especially the ensemble techniques. Hence this model was used for the final prediction of the test dataset.

6 Conclusion

This paper narrates the work of Amrita_CEN with respect to SemEval 2022 Task 4 competition named " Patronizing and Condescending Language Detection ". A total of seven machine learning algorithms were used which include three classical ML models and four ensemble techniques. The problem

[1]GridSearchCV: `https://rb.gy/lajkio`
[2]MultiOutputClassifier: `https://rb.gy/52vpax`

of class imbalance was dealt with minority oversampling technique called SMOTE. Considering macro F1 score for both the sub tasks, logistic regression performed the best and the results were submitted using the same model. Coming to the future work, implementation using deep learning and BERT approaches can give better results compared to classical machine learning models.

7 Acknowledgements

With the invaluable guidance of our faculty Dr Sowmya V, this work and the research has been feasible. Her constant guidance, attitude of perfection and passion of detailing were an inspiration and helped us to achieve our objective.

References

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of hindi-english code-mixed social media text for hate speech detection. In *Proceedings of the second workshop on computational modeling of people's opinions, personality, and emotions in social media*, pages 36–41.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

B Premjith and Soman Kp. 2020. Amrita_cen_nlp@ wosp 3c citation context classification task. In *Proceedings of the 8th International Workshop on Mining Scientific Publications*, pages 71–74.

B Premjith, KP Soman, M Anand Kumar, and D Jyothi Ratnam. 2019. Embedding linguistic features in word embedding for preposition sense disambiguation in english—malayalam machine translation context. In *Recent Advances in Computational Intelligence*, pages 341–370. Springer.

Victoria L Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. 2016. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of the second workshop on computational approaches to deception detection*, pages 7–17.

KP Soman, R Loganathan, and V Ajay. 2009. *Machine learning with SVM and other kernel methods*. PHI Learning Pvt. Ltd.

K Sreelakshmi, B Premjith, and Soman Kp. 2021. Amrita_cen_nlp@ dravidianlangtech-eacl2021: deep learning-based offensive language identification in malayalam, tamil and kannada. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, pages 249–254.

K Sreelakshmi, B Premjith, and KP Soman. 2020. Detection of hate speech text in hindi-english code-mixed data. *Procedia Computer Science*, 171:737–744.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. *arXiv preprint arXiv:1909.11272*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.

JCT at SemEval-2022 Task 4-A: Patronism Detection in Posts Written in English using Pre-processing Methods and various Machine Learning Methods

Yaakov HaCohen-Kerner, Ilan Meyrowitsch, Matan Fchima

Department of Computer Science, Jerusalem College of Technology, Lev Academic Center
21 Havaad Haleumi St., P.O.B. 16031, 9116001 Jerusalem, Israel
kerner@jct.ac.il, meyrowitsch@gmail.com, matanf1992@gmail.com

Abstract

This paper describes our submissions to SemEval-2022 subtask 4-A - "Patronizing and Condescending Language Detection: Binary Classification". We developed different models for this subtask. We applied 11 supervised machine learning methods and 9 pre-processing methods. Our best submission was a model we built with BertForSequenceClassification. Our experiments indicate that pre-processing stage is a must for a successful model. The dataset for Subtask 1 is highly imbalanced. The F1-scores on the oversampled imbalanced training dataset were higher than the results on the original training dataset.

1 Introduction

The explosion of social media in recent years also enables an increasing the number of patronizing and condescending language (PCL). Patronizing language is best described as expressions that are agreeable and show kindness to a person or group in a condescending manner, indicating that the person or group is inferior (McCune and Matthews, 1978).

The discourse of condescension has three main characteristics: (1) It does not contain anything openly critical or negative, and often contains insincere praise; (2) it assumes a difference in status and worth between the writer and the person who wrote about him; and (3) this assumed difference is disputed by the listener (Huckin, 2002).

PCL can harm individuals or groups of people and may cause harmful effects on society. Therefore, it is important to develop efficient computerized systems capable of detecting PCL (Lo and Wei, 2006).

PCL detection is not a simple problem because it requires understanding the context of the situation, the relevant culture, and indirect clues. In social media texts, the problem is harder due to the different levels of ambiguities in natural language and the noisy nature of such texts.

In contrast to the offensive language or hate speech detection field, where there has been relatively an extensive research (e.g., Basile et al., 2019; Zampieri et al., 2019; Zampieri et al., 2020), PCL is still a relatively new and open field of study in Natural language processing (NLP) and machine learning (ML) (Pérez-Almendros et al., 2020).

Pérez-Almendros et al. (2020) introduced the Don't Patronize Me! Dataset. This dataset contains paragraphs extracted from news stories, which have been annotated to indicate the presence of PCL at the text span level.

This paper describes our research and participation in subtask 4-A for patronism detection in posts written in English. The full description of task 4 in general and 4-A, in particular, is given in Perez-Almendros et al. (2022).

The structure of the rest of the paper is as follows. Section 2 introduces a background concerning patronism detection, text pre-processing, and TC with imbalanced classes. Section 3 describes subtask 4-A and its training dataset. In Section 4, we present the submitted models and their experimental results. Section 5 summarizes and suggests ideas for future research.

2 Related Work

Various NLP methods have been applied in the detection of several types of harmful language such as offensive language or hate speech detection (Basile et al., 2019; Zampieri et al., 2019; Zampieri et al., 2020). Previous NLP tasks have generally focused on explicit, aggressive, and flagrant phenomena such as fake news detection (Conroy et al., 2015).

During the last three years, several studies on PCL have appeared. Wang and Potts (2019) introduced the task of modeling condescension and developed an annotated dataset of social media messages. Sap et al. (2019) discussed various implications behind certain uses of language. Mendelsohn et al. (2020) analyzed, from a computational linguistics viewpoint, how language has dehumanized minorities in media news.

2.1 Text preprocessing

Text preprocessing is an important step of TC in general and in social text documents in particular. Classification of text dataset that has not been carefully cleaned or preprocessed might lead to misleading results.

HaCohen-Kerner et al. (2019) investigated the impact of all possible combinations of six preprocessing methods (spelling correction, HTML tag removal, converting uppercase letters into lowercase letters, punctuation mark removal, reduction of repeated characters, and stopword removal) on TC in three benchmark mental disorder datasets. In another study, HaCohen-Kerner et al. (2020) explored the influence of various combinations of the same six basic preprocessing methods on TC in four general benchmark text corpora using a bag-of-words representation. The general conclusion was that it is always advisable to perform an extensive and systematic variety of preprocessing methods because it contributes to improving TC accuracy.

2.2 Text classification with imbalanced classes

The problem with TC with imbalanced classes is that there are too few examples of the minority class to effectively learn a good predictive TC model. There are various methods to cope with this problem (e.g., Liu et al., 2004). The main idea is to change the dataset until a more balanced distribution is reached. Two well-known sampling methods that enable such a change are oversampling and undersampling (e.g., Yap et al., 2014). Random oversampling means randomly duplicating examples in the minority class. Random undersampling means randomly deleting examples in the majority class.

An additional frequent method is to generate synthetic samples, which means randomly sampling the attributes from instances in the minority class (Zhu et al., 2017). There are several algorithms that support the generation of synthetic samples. The most popular one is called the Synthetic Minority Over-sampling Technique (SMOTE) (Chawla, 2002). This method is an oversampling method that creates synthetic samples from the minor class instead of creating copies. This method selects two or more similar instances and perturbs an instance one attribute at a time by a random amount within the difference to the similar instances.

Other possible methods are to try a variety of different types of machine learning (ML) methods in general and penalized variants of these methods that charge an additional cost on the model for making classification mistakes on the minority class during training.

Readers interested in expanding and deepening the topic of solutions to TC with imbalanced classes are referred to the following articles (Chawla et al., 2002; He and Ma, 2013; Krawczyk, 2016; Brownlee, 2020).

3 Task and Training Dataset Description

We only participated in subtask 4-A - “Patronizing and Condescending Language Detection: Binary Classification", which deals with the classification of each post as a patronizing or condescending language (PCL) or not in the English language. Table 1 presents various statistical details about the data set.

	not patronize	is patronize	total
Documents	9,476	993	10,469
% Docs	90.5	9.5	100
words	453,690	53,245	506,935
characters	2,514,890	286,435	2,801,325
avg word per doc	47.87	53.62	50.745
avg chars per doc	265.39	288.45	276.92
words std	32.77	28.62	30.695
chars std	158.36	175.52	166.94

Table 1: Details of the training set.

The analysis of the details presented in Table 1 shows that the dataset is highly imbalanced with a ratio of about 91:9 (not patronize: is patronize). We changed this rate to 77:23 by the creation of new partial 'patronized' posts extracted from various posts that belong to different categories of positive patronized labels available from TASK 6-2 (multi-label classification). We also evaluated an equal split (50:50) by duplication of the patronized sentences. However, the experimental results using the equal split lead to results that were lower than the results using unequal ratios. All the python code lines used for improving the ratio, preprocessing methods, and the different models are available on Github at https://github.com/meyrow/pcl-detection-task4-semeval2022.

4 The Submitted Models and Experimental Results

We applied 11 supervised ML methods to the training dataset. Seven of them were classical ML methods: Random Forest (RF), K Nearest Neighbours (KNN), Support Vector Classifier (SVC), XGBoost Classifier, Logistic Regression (LR), Decision Tree (DT), Naive Bayes (NB), and four of them were deep learning (DL) methods: Bert, DistilBert, Roberta, and Albert.

In our various models, we applied nine sub-types of preprocessing methods: remove newlines, remove HTML Tags, remove Links, remove White spaces, remove accented characters, conversion to lower-case, reduce repeated characters, and punctuations, expand contractions, and remove special characters.

These methods were applied using the following tools and information sources:

- The Python 3.7.3 programming language[1].
- Scikit-learn – a Python library for ML methods[2].
- Numpy – a Python library that provides fast algebraic calculous processing, especially for multidimensional objects[3].

In our experiments, we tried to find the best combination of ML method, preprocessing methods, and oversampling methods. The training set was split into 80:20 train: test and the training test was using 90 percent in every epoch to train and 10 percent of the training set was used for validation.

Figure 1 presents training and validation loss curves of our BERT model with 20 epochs showung that training and validation continuously improved themselves. We noticed that a gap between training and validation began to grow, therefore we had to stop the model after 20 epochs, otherwise, the model will be overfitted. Figures 2 and 3 present the confusion matrices of our BERT model with 20 epochs and the decision tree model, respectively. The confusion matrix of both models demonstrates that the dataset is imbalanced as shown in Table 1. We also noticed that the ratio of 77:23 after improving the original dataset is close to the ratio shown in the confusion matrix. That indicates that our models are well trained. To select the best model we compared the F1-score.

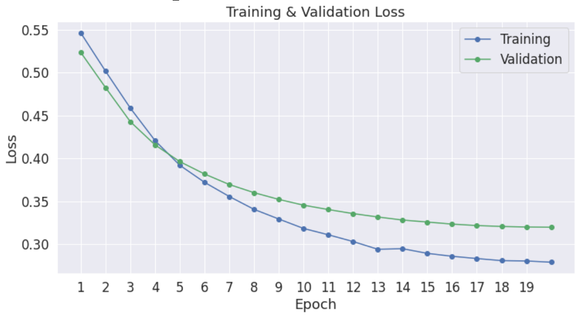


Figure 1: Training and validation loss curves of our BERT model with 20 epochs.

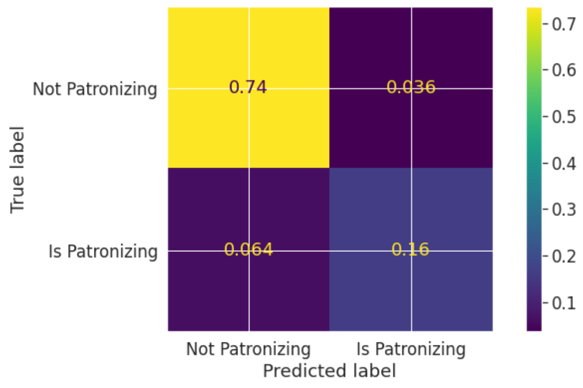


Figure 2: Confusion matrix of our BERT model with 20 epochs.

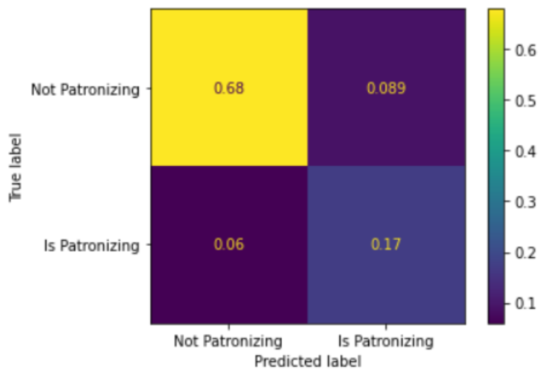


Figure 3: Confusion matrix of our Decision Tree model.

[1] https://www.python.org/downloads/release/python-373/

[2] https://scikit-learn.org/stable/index.html

[3] https://numpy.org

Our best submission was a model called Matan-bert that we built using a function called BertForSequenceClassification. This BERT model includes 768 layers. Its values of the learning rate, epsilon, number of epochs, and batch size were 2e-7, 1e-8, 20, and 16, respectively. This model was ranked the 62nd position. Its F1-score over the PCL class, precision, and recall results are 0.377, 0.3536, and 0.4038, respectively.

Table 2 presents the results of the submitted models. The F1-score over the PCL class on the training dataset of our best model was 0. 77 while the F1-score over the PCL class on the test dataset of our best model was only 0.377. Currently, the posts' labels of the test dataset are unknown. Therefore, we do not have any definite explanation(s) for such a large decrease in the results. Possible explanations might be: (1) The training dataset is different in its balance rate than the balance rate of the competition test dataset and (2) the content of a relatively high number of news items in the competition test dataset is fundamentally different from the content of the news in the training dataset.

Our code is available on Github at https://github.com/meyrow/pcl-detection-task4-semeval2022. Our models are available for reproducibility with comments that explain the code and parameters such as epsilon, learning rate batch, and epochs.

5 Conclusions and Future Research

In this paper, we describe our submissions to subtask 4-A of the SemEval-2022 contest. We submitted the models that achieved the best results while trying to choose two models that applied different supervised learning methods.

Future research ideas include (1) Acronym disambiguation that will extend and enrich the social text and might enable better classification (e.g., HaCohen-Kerner et al., 2008; HaCohen-Kerner et al., 2010A); (2) use of skip character n- to overcome problems such as noise and sparse data (HaCohen-Kerner et al., 2017); (3) use of stylistic feature sets (HaCohen-Kerner et al., 2010B) and key phrases that can be extracted from text files (HaCohen-Kerner et al., 2007).

Model Name	split mode	ML Method	Features	F1-score over the PCL class on the training dataset	F1-score over the PCL class on the test dataset
Matan_bert	80:20	BertForSequence Classification	Layers: 768 Learning rate: 2e-7 Epsilon: 1e-8 Epochs: 20 Batch-size: 16	0.77	0.377
Matan_ Decision_Tree	70:30	Decision Tree	criterion= 'entropy' random_state = 0	0.7	was not published

Table 2: Results of the submitted models.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In Proceedings of SemEval.

Jason Brownlee. 2020. Imbalanced classification with python: Better metrics, balance skewed classes, cost-sensitive learning. Machine Learning Mastery.

Nitesh V. Chawla, Kevin W Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.

Niall J. Conroy, Victoria L. Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. Proceedings of the Association for Information Science and Technology, 52(1):1–4.

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

Haibo He and Yunqian Ma (Eds.). 2013. Imbalanced learning: foundations, algorithms, and applications.

Thomas Huckin. 2002. Critical discourse analysis and the discourse of condescension. Discourse studies in composition, 155, 176.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J. Carman. 2017. Automatic sarcasm detection: A survey. ACM Computing Surveys (CSUR), 50(5), 1-22.

Yaakov HaCohen-Kerner, Ariel Kass, and Ariel Peretz. 2008. Combined one sense disambiguation of abbreviations. In Proceedings of ACL-08: HLT, Short Papers, Association for Computational Linguistics, pages 61-64, Columbus, Ohio, Association for Computational Linguistics. URL: https://aclanthology.org/P08-2.

Yaakov HaCohen-Kerner, Ittay Stern, David Korkus, and Erick Fredj. 2007. Automatic machine learning of keyphrase extraction from short HTML documents written in Hebrew. Cybernetics and Systems: An International Journal, 38(1), 1-21.

Yaakov HaCohen-Kerner, Dror Mughaz, Hananya Beck, and Elchai Yehudai. 2008. Words as classifiers of documents according to their historical period and the ethnic origin of their authors. Cybernetics and Systems: An International Journal, 39(3), 213-228.

Yaakov HaCohen-Kerner, Ariel Kass, and Ariel Peretz. 2010A. HAADS: A Hebrew Aramaic abbreviation disambiguation system. Journal of the American Society for Information Science and Technology, 61(9), 1923-1932.

Yaakov HaCohen-Kerner, Hananya Beck, Elchai Yehudai, and Dror Mughaz. 2010B. Stylistic feature sets as classifiers of documents according to their historical period and ethnic origin. Applied Artificial Intelligence, 24(9), 847-862.

Yaakov HaCohen-Kerner, Ziv Ido, and Ronen Ya'akobov. 2017. Stance classification of tweets using skip char Ngrams. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 266-278). Springer, Cham.

Yaakov HaCohen-Kerner, Yair Yigal, and Daniel Miller. 2019. The impact of Preprocessing on Classification of Mental Disorders, in Proc. of the 19th Industrial Conference on Data Mining, (ICDM 2019), New York.

Yaakov HaCohen-Kerner, Daniel Miller, and Yair Yigal. 2020. The influence of preprocessing on text classification using a bag-of-words representation, PloS one, vol. 15, p. e0232525.

Bartosz Krawczyk. 2016. Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence, 5(4), 221-232.

Ven-hwei Lo, and Ran Wei. 2006. Perceptual differences in assessing the harm of patronizing adult entertainment clubs. International Journal of Public Opinion Research 18.4: 475-487.

Yang Liu, Elizabeth Shriberg, Andreas Stolcke, and Mary Harper. 2004. Using machine learning to cope with imbalanced classes in natural speech: Evidence from sentence boundary and disfluency detection. In Eighth International Conference on Spoken Language Processing.

Shirley D. McCune and Martha Matthews. 1978. Implementing Title IX and attaining sex equity: a workshop package for elementary-secondary educators: the community's role: outline and participants' materials for application sessions for community group members. Department of Health, Education, and Welfare,[Education Division], Office of Education.

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A framework for the computational linguistic analysis of dehumanization.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't patronize me! An annotated dataset with patronizing and condescending language towards vulnerable communities. In Proceedings of the 28th International Conference on Computational Linguistics. pages 5891-5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2019. Social bias frames: Reasoning about social and power implications of language. arXiv preprint arXiv:1911.03891.

Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the

9th International Joint Conference on Natural Language Processing

Bee Wah Yap, Khatijahhusna Abd Rani, Hezlin Aryani Abd Rahman, Simon Fong, Zuraida Khairudin, Nik Nairan Abdullah. 2014. An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets. Proceedings of the first international conference on advanced data and information engineering (DaEng-2013). Springer, Singapore.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 Task 6: Identifying and categorizing offensive language in social media (offenseval). arXiv preprint arXiv:1903.08983.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Cagrı Coltekin. 2020. In Proceedings of SemEval Semeval-2020 task 12: Multilingual offensive language identification in social media.

Tuanfei Zhu, Yaping Lin, and Yonghe Liu. 2017. Synthetic minority oversampling technique for multiclass imbalance problems. Pattern Recognition, 72, 327-340.

ULFRI at SemEval-2022 Task 4: Leveraging uncertainty and additional knowledge for patronizing and condescending language detection

Matej Klemen   **Marko Robnik-Šikonja**
University of Ljubljana, Faculty of Computer and Information Science
Večna pot 113, Ljubljana, Slovenia
{matej.klemen, marko.robnik}@fri.uni-lj.si

Abstract

We describe the ULFRI system used in the Subtask 1 of SemEval-2022 Task 4 Patronizing and condescending language detection. Our models are based on the RoBERTa model, modified in two ways: (1) by injecting additional knowledge (coreferences, named entities, dependency relations, and sentiment) and (2) by leveraging the task uncertainty by using soft labels, Monte Carlo dropout, and threshold optimization. We find that the injection of additional knowledge is not helpful but the uncertainty management mechanisms lead to small but consistent improvements. Our final system based on these findings achieves $F_1 = 0.575$ in the online evaluation, ranking 19th out of 78 systems.

1 Introduction

Despite invaluable contributions to the society, the internet can also serve as an infrastructure for a rapid spread of hurtful language, in part due to the anonymity it commonly provides (Burnap and Williams, 2015). The spread of such language can have a serious impact on individuals, such as the increased development of mental health problems in children (Munro, 2011). To prevent this, the society has to establish moderation mechanisms. Fully manual content moderation is infeasible both due to the large scale of the web as well as the possible negative psychological effects on human moderators (Arsht and Etcovitch, 2018). Much attention has been devoted to the automatic detection of offensive language within the field of natural language processing (NLP). Some examples include the detection of hate speech (Davidson et al., 2017), toxic language (Pavlopoulos et al., 2021), and cyberbullying (Dadvar et al., 2013), which use a relatively explicit form of hurtful language (Waseem et al., 2017). In contrast, patronizing and condescending language (PCL) is more implicit in nature. PCL can roughly be described as an expression of a superior attitude towards others, possibly unconsciously. Perez Almendros et al. (2020) have shown that large language models are able to detect PCL to a various degree, but consistently better than random guessing or a machine learning approach using the bag-of-words representation. Based on that, the authors propose SemEval-2022 Task 4 (Pérez-Almendros et al., 2022), which aims to encourage further research and improvements in the detection of PCL.

We present our attempts at modeling PCL, based on the RoBERTa model (Liu et al., 2019) and following two main lines:

1. **Injection of additional knowledge**. We experiment with the injection of additional knowledge on coreferences, named entities, dependency relations, and sentiment.

2. **Leveraging uncertainty present in the task**. We experiment with the use of soft labels in the form of the target label probability distribution, and with the Monte Carlo dropout as a means for more accurate estimation of the label posterior probability (Gal and Ghahramani, 2016).

Our first set of modifications aims to guide the model to better follow the definition of PCL. The additional coreference and named entity knowledge may help the model to focus on detecting an imbalance between entities in the text, while the dependency relations and sentiment knowledge may help the model discover more subtle linguistic patterns used in PCL. We inject different forms of knowledge as the second input sequence to the model, combining it with the primary text representation during training of the PCL detector. This is motivated by the Factored Transformer (Armengol-Estapé et al., 2021).

The second set of modifications aims to capture the subjectivity and uncertainty that is inherently present in the task and is reflected in the annotator disagreement. This is motivated by the data perspectivism paradigm (Basile et al., 2021) which argues the disagreements are not necessarily errors.

We participate in Subtask 1, and our best model ranks 19th out of 78 systems[1]. In our analysis, we find that (1) injection of additional knowledge does not increase the F_1 score significantly and (2) leveraging uncertainty in the task leads to small but consistent increase in the F_1 score.

The remainder of the paper is structured as follows. In Section 2, we describe the details of the task. In Section 3, we describe our approach, and analyze its performance in Section 4. In Section 5, we summarize our work and provide ideas for further work.

2 Task Description

Given an updated version of the *Don't Patronize Me!* dataset (Perez Almendros et al., 2020), the goal of SemEval-2022 Task 4 Subtask 1 is the detection of patronizing and condescending language (PCL). The provided dataset consists of 10 469 paragraphs annotated by three annotators: two were tasked with annotating the examples as not containing PCL (0), containing PCL (2), or borderline (1). The third annotator resolved *complete* disagreements, i.e. examples annotated as $\{0, 2\}$ by the two annotators. The annotations are aggregated into a five-point fine-grained class y_F:

- $y_F = 0$ if both annotators assigned the label 0,
- $y_F = 1$ if one annotator assigned the label 0 and the other assigned 1,
- $y_F = 2$ if both annotators assigned the label 1,
- $y_F = 3$ if one annotator assigned the label 2 and the other assigned 1,
- $y_F = 4$ if both annotators assigned the label 1.

We provide the distribution of these fine-grained class labels in Figure 1. For the task evaluation, the fine-grained five label class is binarized into a coarse-grained class y_C, where $y_C = 1$ if $y_F \in \{2, 3, 4\}$, and $y_C = 0$ otherwise. Although the final evaluation uses binary labels, the fine-grained labels can provide additional information in the form of label uncertainty. We leverage this information in one of our modifications, described next.

[1]Our code is available at `https://github.com/matejklemen/pcl-detection-semeval2022t4`.

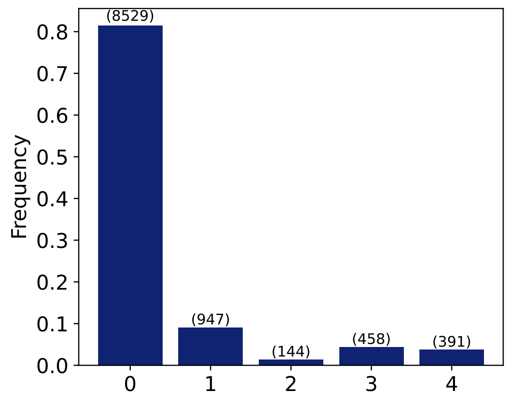


Figure 1: Fine-grained PCL label distribution. The numbers above bars indicate the number of examples for each label.

3 Methods

In this section, we describe our methodology. First, we describe RoBERTa, which we use as the baseline model. Then, we describe how additional knowledge is injected into the model in Section 3.2. In Section 3.3 we describe how we leverage the task uncertainty in our model.

3.1 RoBERTa

RoBERTa (Liu et al., 2019) is a robustly optimized BERT model (Devlin et al., 2019), composed of multiple transformer layers that use the self-attention mechanism to construct a text representation. It is first pre-trained on a general corpus using the masked language modeling objective, after which it can be fine-tuned for a downstream task. Motivated by its strong performance on the PCL detection task shown by Perez Almendros et al. (2020), we use the RoBERTa$_{\text{BASE}}$ model as a baseline.

3.2 Knowledge injection

We inject various types of additional knowledge through a secondary aligned input sequence containing additional knowledge in the form of special tokens. The procedure is shown in Figure 2 for one type of additional knowledge. Using RoBERTa, we independently obtain two representations and combine them using a learned weighted linear combination to obtain a single representation. Lastly, a linear layer transforms the representation into label

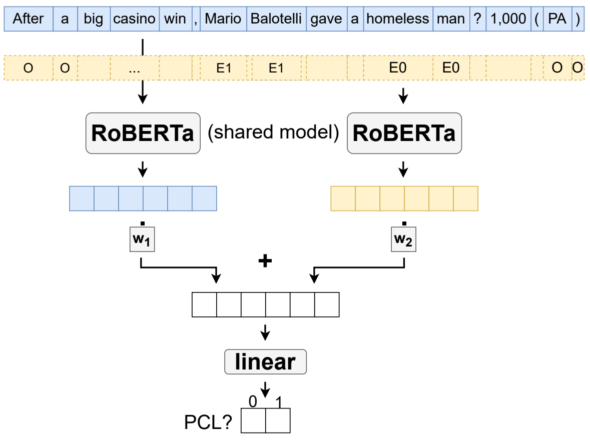

After a big casino win , Mario Balotelli gave a homeless man ? 1,000 (PA)
O O ... E1 E1 E0 E0 O O
RoBERTa (shared model) RoBERTa
w1
w2
+
linear
PCL?
0 1

Figure 2: Injection of additional coreference knowledge for PCL detection. The secondary sequence (in yellow) consists of special tokens that denote if a word represents an entity (E_i) or not (O).

scores. The individual sequence representations correspond to the output of the last layer for the <s> token in each sequence.

We experiment with four different types of additional knowledge, one at a time: coreferences (obtained using neuralcoref[2]), sentiment (obtained using SentiWordNet (Esuli and Sebastiani, 2006)), named entities, and dependency relations (obtained using Stanza (Qi et al., 2020)). We provide additional preprocessing details and the used tagsets in Appendix A.

As our modification requires embedding two input sequences instead of one, the memory requirement during training is doubled, and the batch size has to be halved. To minimize differences due to a halved batch size, we accumulate gradients over two half-sized batches before updating the parameters.

3.3 Leveraging uncertainty

We experiment with two ways to leverage the task uncertainty. The first approach trains a model on soft instead of hard (one-hot encoded) labels. We show the comparison between hard and soft labels in Table 1. As described in Section 2 each example is annotated twice. We assign each annotation a probability of the example containing PCL: 0.0 if the annotation is 0, 1.0 if it is 2, and 0.5 if it is 1 (borderline). To obtain the final soft labels, we then take the mean of the two annotations. In this way, we transform a five-class problem into a binary one while approximately preserving information about label differences. Additionally, we potentially avoid issues when a label has few training examples.

[2]https://github.com/huggingface/neuralcoref

Table 1: Conversion scheme from fine-grained annotations into hard and soft binary target vector.

Label type	Fine-grained annotation (y_F)	Binary target vector
hard	0, 1	[1.00, 0.00]
	2, 3, 4	[0.00, 1.00]
soft	0	[1.00, 0.00]
	1	[0.75, 0.25]
	2	[0.50, 0.50]
	3	[0.25, 0.75]
	4	[0.00, 1.00]

The second approach uses the Monte Carlo dropout (MCD) (Gal and Ghahramani, 2016) to sample the label distribution during the prediction phase. Instead of determining the target label using a single prediction, we obtain multiple non-deterministic predictions while applying dropout (Srivastava et al., 2014), and then aggregate them into a single prediction (in our case, using the mean) (Miok et al., 2022).

Both modifications transform the target label probability distribution, so using the PCL probability threshold of 0.5 may no longer be suitable. For this reason, we also experiment with the decision threshold optimization, i.e. we select the threshold based on the validation set F_1 score.

4 Evaluation

In this section, we evaluate our methodology and compare it to the baseline. We start by describing the experimental settings in Section 4.1, and continue with the results in Section 4.2.

4.1 Experimental settings

We select the hyperparameters for the training of RoBERTa using the validation set F_1 score in preliminary experiments on a single 80%:10%:10% split into the training, validation and testing set. In our main experiments, we use the learning rate 10^{-5}, maximum sequence length of 158, and batch size of 48. The latter two were selected in a way to allow training on an 11GB GPU.

In the evaluation, we use 10-fold cross validation and report the means across folds. In each cross validation iteration, we use 10% of the training set for tuning and early stopping. Following the official evaluation, we use three metrics: precision, recall, and F_1 score for the positive (PCL) label. To improve clarity, we only report the mean F_1 score throughout this section, and provide other metrics and standard deviations of the results in Appendix B. We statistically test the differences in F_1 score between pairs of models using the Wilcoxon signed-rank test (Wilcoxon, 1945). The same test is applied to the difference between groups of models, where one group uses and one group does not use certain modification (e.g., soft labels). In all cases, we use a confidence level $\alpha = 0.01$ to determine the significance of the differences.

For the online evaluation, we retrain the model using the best parameters on a 90%:10% split into a training and validation set.

4.2 Results

Table 2 shows the F_1 scores of our enhanced models in comparison to the RoBERTa$_{\mathrm{BASE}}$ baseline. We interpret the results below, starting with knowledge-enhanced models in Section 4.2.1 and models leveraging uncertainty in Section 4.2.2.

4.2.1 Knowledge-enhanced models

We first only consider the knowledge injection in isolation, i.e. the scores for each type of knowledge in Table 2a.

We can observe that the addition of knowledge about coreferences ($F_1 = 0.575$), named entities ($F_1 = 0.563$), and dependency relations ($F_1 = 0.567$) increases the performance over the baseline ($F_1 = 0.556$). However, none of the increases are statistically significant due to the variance in performance across the folds.

4.2.2 Knowledge- and uncertainty-enhanced models

Next, we consider the effect of leveraging uncertainty both on the base model as well as the knowledge-enhanced models, i.e. analyze the full results in Table 2. Unless stated otherwise, we compare the modified models with their respective base model without the discussed modifications (i.e. not necessarily always against the roberta-base model without MCD).

The first observation is that training the models on soft instead of hard labels in most cases improves the F_1 score both when not using MCD and when using MCD. Using soft labels increases the

Table 2: Results of the base models and their modifications. The best score in each table is displayed in bold.

(a) Results without MCD.

	train on hard labels	train on soft labels
Model	$\mathbf{F_1}$	$\mathbf{F_1}$
roberta-base	0.556	0.570
+ opt. thresh.	0.550	0.577
+ coreference	0.575	**0.582**
+ opt. thresh.	0.573	0.572
+ sentiment	0.544	0.554
+ opt. thresh.	0.532	**0.582**
+ named ent.	0.563	0.568
+ opt. thresh.	0.563	0.577
+ dep. relations	0.567	0.580
+ opt. thresh.	0.557	0.578

(b) Results using 10 MCD rounds.

	train on hard labels	train on soft labels
Model	$\mathbf{F_1}$	$\mathbf{F_1}$
roberta-base	0.546	0.553
+ opt. thresh.	0.551	0.580
+ coreference	0.550	0.566
+ opt. thresh.	0.573	0.579
+ sentiment	0.526	0.535
+ opt. thresh.	0.540	0.575
+ named ent.	0.556	0.556
+ opt. thresh.	0.553	0.577
+ dep. relations	0.557	0.557
+ opt. thresh.	0.564	**0.583**

(c) Results using 50 MCD rounds.

	train on hard labels	train on soft labels
Model	$\mathbf{F_1}$	$\mathbf{F_1}$
roberta-base	0.546	0.554
+ opt. thresh.	0.556	**0.586**
+ coreference	0.549	0.570
+ opt. thresh.	0.568	0.580
+ sentiment	0.527	0.563
+ opt. thresh.	0.543	0.581
+ named ent.	0.557	0.563
+ opt. thresh.	0.569	0.581
+ dep. relations	0.554	0.560
+ opt. thresh.	0.569	0.577

F_1 score for 13 models and makes no difference for 2 models (named entity and dependency relation enhanced models using 10 MCD rounds). The differences are statistically significant.

Optimizing the threshold after training on soft labels improves the F_1 score for 14 models: for 13 models, the improvement over baseline is larger than without threshold optimization. The only exception where the F_1 score decreases is the coreference enhanced model without MCD, which could be due to overfitting the threshold to the tuning set. Optimizing the threshold after training on hard labels has a mixed effect. For models without MCD, it consistently leads to equivalent or lower F_1 scores compared to the baseline model without the optimized threshold. On the other hand, the threshold optimization leads to the statistically significant increase in the F_1 score for models with MCD. Concretely, it increases the F_1 score for 9 models and decreases it for 1 model.

Using MCD on its own is not helpful. Without also training the model on soft labels or optimizing the threshold, it decreases the F_1 score for all 10 models in comparison to the respective models not using MCD. However, as described previously, using MCD in combination with either or both of these mechanisms mostly leads to an increase in F_1 score.

Lastly, using more MCD rounds starts to bring a diminishing increase in the F_1 score after a certain point. Concretely, using 50 instead of 10 MCD rounds leads to only a small additional increase in F_1 score.

The best F_1 score is achieved by the model without additional knowledge, trained on soft labels, using 50 MCD rounds and the threshold optimization. This model achieves the F_1 score of 0.586 (+0.030 over the baseline) in the offline evaluation, and the F_1 score 0.575 on the official online test set.

5 Conclusion

We have described our approaches for the detection of PCL as part of the SemEval-2022 Task 4. We attempted to inject knowledge into prediction models and leverage the uncertainty present in the task. The injection of additional knowledge did not increase the F_1 score significantly. Leveraging the uncertainty in different ways produced mixed effects. Training the models on soft instead of hard labels consistently increased the F_1 score, while using MCD on its own was not beneficial. However, using MCD in combination with soft labels and threshold optimization brought consistent improvements in the F_1 score and produced our best score.

Both directions of our research have potential for further work. In our knowledge injection experiments, we have only experimented with a single type of additional knowledge at a time. To inject multiple types simultaneously, we would need to create special tokens for each combination, which could lead to overfitting due to relatively small and imbalanced data. Therefore, in further work we will try a different method for knowledge injection considering multiple types of additional knowledge simultaneously. In leveraging uncertainty, we have constructed the soft binary labels from the two annotations per example and aggregated the annotations by weighing them equally. A possible further work would experiment with different weighting schemes.

Acknowledgements

The research was supported by the Slovene Research Agency through research core funding no. P6-0411, project J6-2581 (CANDAS - Computer-assisted multilingual news discourse analysis with contextual embeddings), and the young researcher grant.

References

Jordi Armengol-Estapé, Marta R. Costa-jussà, and Carlos Escolano. 2021. Enriching the transformer with linguistic factors for low-resource machine translation. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 73–78.

Andrew Arsht and Daniel Etcovitch. 2018. The human cost of online content moderation. *Harvard Journal of Law and Technology Digest*.

Valerio Basile, Federico Cabitza, Andrea Campagner, and Michael Fell. 2021. Toward a perspectivist turn in ground truthing for predictive computing. In *Conference of the Italian Chapter of the Association for Intelligent Systems (ItAIS 2021)*.

Pete Burnap and Matthew L. Williams. 2015. Cyber hate speech on Twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2):223–242.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, pages 512–515.

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

Andrea Esuli and Fabrizio Sebastiani. 2006. SENTIWORDNET: A publicly available lexical resource for opinion mining. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059.

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

Kristian Miok, Blaž Škrlj, Daniela Zaharie, and Marko Robnik-Šikonja. 2022. To BAN or not to BAN: Bayesian attention networks for reliable hate speech detection. *Cognitive Computation*, 14(1):353–371.

Emily Munro. 2011. The protection of children online: A brief scoping review to identify vulnerable groups. Accessed: January 27, 2022.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043.

John Pavlopoulos, Jeffrey Sorensen, Léo Laugier, and Ion Androutsopoulos. 2021. SemEval-2021 task 5: Toxic spans detection. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 59–69.

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Table 3: Used tagsets for representing additional knowledge.

Knowledge	Tagset
coreference	O, ENT0, ENT1, . . . , ENT50; 52 tags
sentiment	NEG, OBJ, POS, UNK; 4 tags
named ent.	O, {B-, I-, E-, S-} × {ORG, PER, LOC, MISC}; 17 tags
dep. relations	universal dep. relations (including subtypes) (Nivre et al., 2020); 63 tags

A Additional details of knowledge injection experiment

Table 3 shows the tags used to inject the additional knowledge. The tags are added as special (indivisible) tokens to the tokenizer and used in the secondary input sequence as described in Section 3.2. For sentiment and coreference knowledge, we note additional preprocessing details:

- **Sentiment.** We obtain the sentiment tags using SentiWordNet. Each token is assigned a positive, negative and objectivity score, and

Table 4: Extended results of the base models and their modifications: mean and standard deviation of precision (P), recall (R), and F_1 score across 10 folds.

(a) Results without MCD.

	train on hard labels			train on soft labels		
Model	**P**	**R**	$\mathbf{F_1}$	**P**	**R**	$\mathbf{F_1}$
roberta-base	0.575 (0.051)	0.543 (0.057)	0.556 (0.038)	0.612 (0.040)	0.539 (0.067)	0.570 (0.045)
+ opt. thresh.	0.571 (0.056)	0.537 (0.068)	0.550 (0.048)	0.597 (0.060)	0.570 (0.081)	0.577 (0.051)
+ coreference	0.591 (0.046)	0.567 (0.063)	0.575 (0.032)	0.583 (0.073)	0.594 (0.061)	0.582 (0.031)
+ opt. thresh.	0.566 (0.061)	0.593 (0.066)	0.573 (0.033)	0.572 (0.084)	0.591 (0.077)	0.572 (0.028)
+ sentiment	0.589 (0.056)	0.514 (0.074)	0.544 (0.054)	0.614 (0.061)	0.529 (0.132)	0.554 (0.077)
+ opt. thresh.	0.587 (0.043)	0.494 (0.080)	0.532 (0.056)	0.596 (0.061)	0.585 (0.097)	0.582 (0.047)
+ named ent.	0.538 (0.053)	0.604 (0.076)	0.563 (0.026)	0.569 (0.067)	0.585 (0.081)	0.568 (0.028)
+ opt. thresh.	0.563 (0.021)	0.588 (0.087)	0.563 (0.021)	0.565 (0.044)	0.595 (0.062)	0.577 (0.031)
+ dep. relations	0.537 (0.028)	0.605 (0.076)	0.567 (0.040)	0.581 (0.042)	0.585 (0.064)	0.580 (0.028)
+ opt. thresh.	0.572 (0.061)	0.558 (0.091)	0.557 (0.039)	0.593 (0.037)	0.575 (0.085)	0.578 (0.038)

(b) Results using 10 MCD rounds.

	train on hard labels			train on soft labels		
Model	**P**	**R**	$\mathbf{F_1}$	**P**	**R**	$\mathbf{F_1}$
roberta-base	0.610 (0.062)	0.499 (0.054)	0.546 (0.041)	0.660 (0.054)	0.481 (0.068)	0.553 (0.056)
+ opt. thresh.	0.580 (0.064)	0.534 (0.069)	0.551 (0.040)	0.593 (0.043)	0.577 (0.095)	0.580 (0.059)
+ coreference	0.612 (0.043)	0.507 (0.084)	0.550 (0.051)	0.614 (0.069)	0.537 (0.075)	0.566 (0.038)
+ opt. thresh.	0.537 (0.058)	0.621 (0.055)	0.573 (0.040)	0.572 (0.078)	0.602 (0.085)	0.579 (0.043)
+ sentiment	0.627 (0.063)	0.460 (0.074)	0.526 (0.059)	0.667 (0.075)	0.474 (0.129)	0.535 (0.086)
+ opt. thresh.	0.575 (0.056)	0.514 (0.058)	0.540 (0.047)	0.610 (0.059)	0.553 (0.077)	0.575 (0.042)
+ named ent.	0.589 (0.062)	0.543 (0.087)	0.556 (0.034)	0.621 (0.077)	0.523 (0.098)	0.556 (0.041)
+ opt. thresh.	0.560 (0.062)	0.561 (0.086)	0.553 (0.034)	0.540 (0.056)	0.628 (0.074)	0.577 (0.043)
+ dep. relations	0.577 (0.039)	0.544 (0.063)	0.557 (0.037)	0.633 (0.062)	0.511 (0.100)	0.557 (0.060)
+ opt. thresh.	0.555 (0.044)	0.577 (0.046)	0.564 (0.033)	0.572 (0.059)	0.608 (0.076)	0.583 (0.031)

(c) Results using 50 MCD rounds.

	train on hard labels			train on soft labels		
Model	**P**	**R**	$\mathbf{F_1}$	**P**	**R**	$\mathbf{F_1}$
roberta-base	0.609 (0.055)	0.499 (0.051)	0.546 (0.038)	0.662 (0.052)	0.480 (0.062)	0.554 (0.051)
+ opt. thresh.	0.564 (0.046)	0.558 (0.090)	0.556 (0.046)	0.594 (0.058)	0.587 (0.063)	0.586 (0.034)
+ coreference	0.616 (0.035)	0.503 (0.078)	0.549 (0.046)	0.620 (0.080)	0.538 (0.072)	0.570 (0.043)
+ opt. thresh.	0.561 (0.084)	0.601 (0.099)	0.568 (0.040)	0.582 (0.084)	0.599 (0.101)	0.580 (0.048)
+ sentiment	0.630 (0.066)	0.461 (0.074)	0.527 (0.061)	0.629 (0.080)	0.531 (0.099)	0.563 (0.045)
+ opt. thresh.	0.574 (0.040)	0.523 (0.092)	0.543 (0.055)	0.543 (0.058)	0.633 (0.059)	0.581 (0.038)
+ named ent.	0.589 (0.068)	0.544 (0.086)	0.557 (0.038)	0.629 (0.080)	0.531 (0.099)	0.563 (0.045)
+ opt. thresh.	0.551 (0.038)	0.596 (0.072)	0.569 (0.030)	0.543 (0.058)	0.633 (0.059)	0.581 (0.038)
+ dep. relations	0.575 (0.033)	0.541 (0.071)	0.554 (0.043)	0.636 (0.056)	0.513 (0.091)	0.560 (0.049)
+ opt. thresh.	0.556 (0.043)	0.588 (0.062)	0.569 (0.035)	0.568 (0.058)	0.607 (0.107)	0.577 (0.050)

the final tag is the one with the highest of the three scores. The token UNK is used if a token does not have associated scores.

- **Coreference.** We enumerate the coreference clusters in the document randomly. We find this has a positive effect on the performance, possibly as the model is overtrained on the tags with lower IDs otherwise.

B Extended evaluation results

Table 4 shows the extended results of the base and extended models.

SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene
University of Milano-Bicocca
Viale Sarca 336, 20126, Milan, Italy
elisabetta.fersini@unimib.it, francesca.gasparini@unimib.it
g.rizzi10@campus.unimib.it, a.saibene2@campus.unimib.it

Berta Chulvi, Paolo Rosso
Universitat Politècnica de València
Camino de Vera, Valencia, Spain
berta.chulvi@upv.es
prosso@dsic.upv.es

Alyssa Lees, Jeffrey Sorensen
Google Jigsaw,
111 8th Ave, New York, NY
alyssalees@google.com
sorenj@google.com

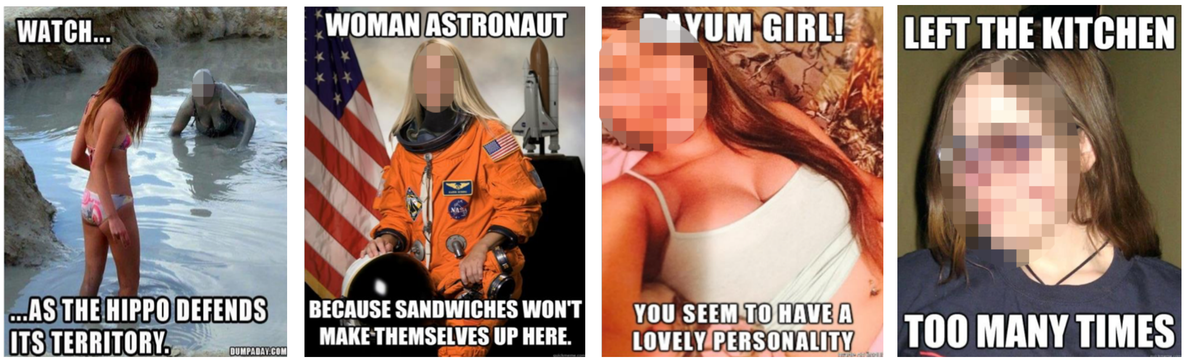
Abstract

The paper describes the SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification (MAMI),which explores the detection of misogynous memes on the web by taking advantage of available texts and images. The task has been organised in two related sub-tasks: the first one is focused on recognising whether a meme is misogynous or not (Sub-task A), while the second one is devoted to recognising types of misogyny (Sub-task B). MAMI has been one of the most popular tasks at SemEval-2022 with more than 400 participants, 65 teams involved in Sub-task A and 41 in Sub-task B from 13 countries. The MAMI challenge received 4214 submitted runs (of which 166 uploaded on the leader-board), denoting an enthusiastic participation for the proposed problem. The collection and annotation is described for the task dataset. The paper provides an overview of the systems proposed for the challenge, reports the results achieved in both sub-tasks and outlines a description of the main errors for a comprehension of the systems capabilities and for detailing future research perspectives.

1 Introduction

Women have a strong presence online, particularly in image-based social media such as Twitter and Instagram: 78% of women use social media multiple times per day compared to 65% of men (Department, 2019). However, while new opportunities for women have been opened on the web, systematic inequality and discrimination offline is replicated in online spaces in the form of offensive contents against them (Frenda et al., 2019; Anzovino et al., 2018; Farrell et al., 2019; Plaza-Del-Arco et al., 2020; Gasparini et al., 2018). A popular communication tool in social media platforms are image macros popularly connoted as "memes" (Shifman, 2013). An internet meme is usually an image communicating pictorial content with an overlaid text that is added *a posteriori* by the meme author, with the main goal of being funny and/or ironic (Shifman, 2013). Although many memes are created with humorous intent, others have political or activist ambitions. Few familiar with the format would be surprised to learn that memes can be used to express hate against women, via sexist and aggressive messages in online environments (Paciello et al., 2021) that subsequently amplify the sexual stereotyping and gender inequality of the offline world (Franks, 2011). In order to counter this phenomenon, the Multimedia Automatic Misogyny Identification (MAMI) shared task has been organised at SemEval-2022 (Emerson et al., 2022). The proposed challenge consists of the identification of misogynous memes, taking advantage of both text and images available as sources of information. The task is organised around two main sub-tasks:
- **Sub-task A:** a basic task of misogynous meme identification, where a meme should be categorised either as misogynous or not misogynous;
- **Sub-task B:** an advanced task, where the type of misogyny should be recognised among potential overlapping categories such as stereotype, shaming, objectification and violence.

Some other tasks related to this topic, but that did not consider the same data and a multimodal approach have been previously organised in the same area of interest, i.e. AMI@Evalita (Fersini et al., 2018a; Elisabetta Fersini, 2020), AMI@IberEval (Fersini et al., 2018b), HatEval (Basile et al., 2019) and OffenseEval (Zampieri


WATCH...
...AS THE HIPPO DEFENDS ITS TERRITORY.
DUMPADAY.COM
WOMAN ASTRONAUT
BECAUSE SANDWICHES WON'T MAKE THEMSELVES UP HERE.
YUM GIRL!
YOU SEEM TO HAVE A LOVELY PERSONALITY
LEFT THE KITCHEN
TOO MANY TIMES

(a) Shaming (b) Stereotype (c) Objectification (d) Violence

Figure 1: Examples of misogynous memes.

et al., 2020). However, the proposed MAMI challenge is a step forward the previous ones for two main reasons: (1) it is focused on multi-modality and (2) the type of misogynous contents are expressed in a completely different form, i.e. in the former challenge the presence of hateful contents was explicit within the text, while here it is often implicit.

2 Dataset and Annotation Process

Candidate memes have been collected by focusing on the following main types of misogyny:

- *Shaming*: The practice of criticising women who violate expectations of behaviour and appearance regarding issues related to gender typology (such as "slut shaming") or related to physical appearance (such as "body shaming") (Van Royen et al., 2018). This category focuses on content that seeks to insult and offend women because of some characteristics of their body or personality.

- *Stereotype*: a stereotype is a fixed, conventional idea or set of characteristics assigned to a woman (Eagly and Mladinic, 1989). A meme can use an image of a woman according to her role in the society (role stereotyping), or according to her personality traits and domestic behaviours (gender stereotyping).

- *Objectification*: A practice of seeing and/or treating a woman like an object (Szymanski et al., 2011).

- *Violence*: A meme that indicates physical and/or a call to violence against women (Andreasen, 2021).

Examples of the above mentioned types of misogynous memes are presented in Figure 1.

The procedure for collecting relevant memes for this shared task consisted of: (1) searching the most popular **social media platforms**, such as Twitter and Reddit; and (2) downloading samples from **websites** dedicated to meme creation and sharing, such as 9GaG, Knowyourmeme and Imgur, by site scraping and manual download. In both cases, in order to collect a proper number of misogynous memes, 4 main activities have been performed: (1) searching for threads dedicated to memes with women as the subject; (2) searching for threads or conversations dedicated to or written by persons who identify as anti-women or anti-feminist (such as the MGTOW website and the related threads on Reddit); (3) exploring discussions in recent events involving famous women (such as *Michelle Obama*); (4) searching by keywords and/or hashtags such as #girl, #girlfriend, #women, #feminist.

The final collection is composed of 15k memes that have been labelled by human annotators (duplicates have been previously removed). Among the labelled memes we obtained an adequate number of misogynous and non misogynous memes. The final benchmark dataset released for the MAMI challenge is composed of 10k memes for training and 1k for testing (balanced between classes). The dataset has been labelled using crowd-sourcing platforms according to the following primary questions[1]:

- Is this meme misogynous or not?

- If the meme is misogynous, what are the main categories to which the meme belongs (shaming,

[1]The prototype of the annotation interface and the annotation guidelines are reported in Appendix A

file_name	misogynous	shaming	stereotype	objectification	violence	Text Transcription
10846.jpg	1	0	1	1	1	SANDWICH!!!!!! don't make me tell you twice woman.

Table 1: Annotation format of the training and testing instances.

stereotype, objectification, violence)?
In the last case, i.e. related to the misogyny category, multiple overlapping labels have been considered. The memes were shown one at a time to avoid bias introduced by the annotators seeing multiple memes simultaneously.

Memes were annotated by 3 observers and the final label was given according to the majority of the labels (2/3). The text of the memes have been transcribed using Google Cloud Vision[2]. We report an example of a meme that has been provided to the participants as training example, which is composed of raw image (Figure 2) and the corresponding labels available through a csv file (Table 1).

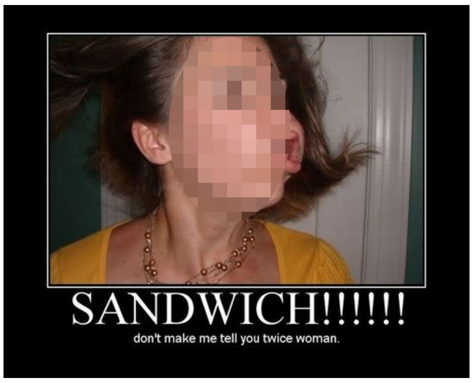


Figure 2: Raw image (10486.jpg)

We estimated the inter-annotator agreement using the Fleiss-κ coefficient (Fleiss, 1971). In particular, we used the traditional Fleiss-κ measure for estimating the agreement related to the misogynous vs not misogynous annotation necessary for Sub-task A, while we adopted the Fleiss-κ with the MASI (Jaccard) index (Passonneau, 2006) to calculate the agreement between annotators on multiple (overlapping) annotations necessary for Sub-task B. Regarding the agreement on the misogynous vs not misogynous annotations, we estimated a coefficient equal to 0.5767, while for the type of misogyny labelling we derived a coefficient equal to 0.3373. We report in Table 2 the details about the dataset provided to the participants. The values of the Fleiss-κ measure suggest that the agreement for the misogynous labelling is moderate, denoting a quite simple task for humans, while the agreement for the type of misogyny annotation is fair, denoting a quite hard task.

[2]https://cloud.google.com/vision/docs/ocr

3 Evaluation Measures and Baseline

Sub-task A. Systems have been evaluated using macro-average F1-Measure. In particular, for each class label (misogynous and not misogynous) the corresponding F1-Measure has be computed, and the final score has been estimated as the arithmetic mean of the two F1-Measures. The baseline models used as benchmark with respect to the participants are:

- **Baseline Text**: a deep representation of text, a fine-tuned sentence embedding using the USE (Cer et al., 2018) pre-trained model;
- **Baseline Image**: deep representation of image content, based on a fine-tuned image classification model grounded on VGG-16 (Simonyan and Zisserman, 2014);
- **Baseline Image_Text**: a concatenation of the previous deep image and text representations through a single layer neural network.

We also used two multi-label models introduced for Sub-task B and detailed in the following paragraph.

Sub-task B. Systems have been evaluated using weighted-average F1-Measure. In particular, the F1-Measure has been computed for each label and then the average has been weighted by the number of true instances for each label. For Sub-task B, the baselines are grounded on:

- **Baseline Flat Multi-label**: a multi-label model, based on the concatenation of deep image and text representations for predicting simultaneously if a meme is misogynous and the corresponding type;
- **Baseline Hierarchical Multi-label**: a hierarchical multi-label model, based on text representations for predicting whether a meme is misogynous or not and, if misogynous, the corresponding type.

4 Participant Systems and Results

MAMI has been one of the most popular tasks in SemEval-2022, with 65 teams that joined Sub-task

	Misogyny Labelling (Sub-task A)			Type of Misogyny Labelling (Sub-task B)				
	Misogynous	**Not Misogynous**	**Fleiss-k Agreement**	**Shaming**	**Stereotype**	**Objectification**	**Violence**	**Fleiss-k Agreement**
Training Set	5000 (50%)	5000 (50%)	0.5767	1274 (25.48%)	2810 (56.20%)	2202 (44.04%)	953 (19.06%)	0.3373
Test Set	500 (50%)	500 (50%)		146 (29.20%)	350 (70.00%)	348 (69.60%)	153 (30.60%)	

Table 2: Dataset characteristics.

A and 41 teams that participated in Sub-task B. We received a total of 4,214 submissions, of which 166 submitted to the leader-board. Among the teams joining the MAMI challenge, 41 groups have provided the details about their participation (team name, number of team members, country, and description of their system). In Appendix B (Table 8), we report features about the teams that have provided team information for further analysis and discussion. On average, the teams are composed of 2 members, varying from 1-person teams (the most frequent case) to 7 members (the largest team). Regarding geographic distribution, the majority of the participants come from India (12 teams), followed by USA and Germany (5), UK and China (4), Italy and Spain (3) and the remaining countries with 1 team each.

As a general overview of the results, we report in Table 3 the mean, standard deviation (StDev), minimum, maximum, median and the first and third quartiles (Q1 and Q3) of the performance achieved by the participant teams.

In Sub-task A, we notice that the maximum value

	Min	Q1	Mean	Median	StDev	Q3	Max
Sub-task A	0.481	0.649	0.680	0.679	0.064	0.722	0.834
Sub-task B	0.467	0.634	0.663	0.680	0.059	0.706	0.731

Table 3: Basic statistics of the results for the participating systems in Sub-task A and Sub-task B, expressed in terms of macro-averaged and weighted-average F_1-score respectively.

(0.834) is much higher than the corresponding one in Sub-task B (0.731), while the difference is less evident when considering the mean (from 0.680 to 0.663) and the median value (from 0.679 to 0.680). When considering the max values, it emerges that Sub-task B seems to be more difficult than Sub-task A, while the median values indicates that for the 50% of the systems both tasks are equally challenging.

In regards to the models adopted by the participants, it has been observed that the majority of the teams exploited pre-trained models, distinguished in text-based, where the most used ones are based on BERT (Devlin et al., 2019) such as RoBERTa (Liu et al., 2019), and image-based models, where the most adopted ones are based on VisualBERT (Li et al., 2020a). Among these systems, considered by 90% of the teams, half of them adopted an ensemble strategy to make the final prediction. The remaining ones adopted either traditional neural networks (30%) or multi-task (20%) approaches to classify the memes. Few teams exploited models, such as CLIP (Radford et al., 2021) and ViLBERT (Lu et al., 2019), to jointly learn the characteristics of misogynous and not misogynous memes, and the related misogyny categories.

4.1 Sub-task A

Sub-task A was attempted by 65 teams, where 47 of them (72%) outperformed the best provided baseline, the Baseline Hierarchical Multi-label model, in terms of macro-averaged F_1-score. The highest score (0.834) has been obtained by the SRCB team (Zhang and Wang, 2022), which defined an ensemble model of deep multi-modal features with Multi Layer Perception (Kubat, 1999), Extreme Gradient Boosting (Chen and Guestrin, 2016) and Gradient-Boosted Decision Trees (Si et al., 2017).

We report in Table 4 the Top-10 teams in Sub-task A, ranked according to macro-average F_1-score (the overall leader-board is reported in Appendix C.) Regarding the top-3 systems, DD-TIG

	Team Name
1	SRCB (Zhang and Wang, 2022)
2	DD-TIG (Zhou et al., 2022)
3	RIT Boston (Chen and Chou, 2022)
4	NLPros
5	ASRtrans (Rao and Rao, 2022)
6	Poirot (Srivastava, 2022)
7	R2D2 (Sharma et al., 2022b)
8	PAIC (ZHI et al., 2022)
	ymf924
	RubCSG (Yu et al., 2022)
9	hate-alert
10	AMS_ADRN (Li et al., 2022)

Table 4: Top-10 teams in Sub-task A, ranked according to macro-average F_1-score.

(Zhou et al., 2022), ranked second place by defining an ensemble of different pre-trained models: (1) ERNIE-Vil (Yu et al., 2021), which incorporates structured knowledge obtained from scene graphs to learn joint representations of vision-language; (2) Uniter (Chen et al., 2020), which learns a joint multi-modal embedding through a Transformer-based architecture over four image-text datasets; (3) VisualBERT (Li et al., 2020a), which is composed of a stack of Transformer layers that implicitly align elements of an input text and regions in an associated input image with self-attention; (4) Oscar (Li et al., 2020b), which exploits object tags detected in an image as anchor point to learn the alignment with the caption fragments.

RIT Boston (Chen and Chou, 2022) ranked third and used OpenAI's CLIP model (Radford et al., 2021) to obtain high-quality multi-modal features and then used a logistic regression (LR) model to make a binary classification. In their model, a data-centric AI principle was used to further improve performance by manually rating a subset of test data and adding this extra data into the train set.

4.2 Sub-task B

Sub-task B was attempted by 41 teams, where 35 of them (85%) outperformed the best MAMI baseline, which also in this case is the Baseline Hierarchical Multi-label model. We report in Table 5 the Top-10 teams in Sub-task B, ranked according to weighted-average F_1-score (the overall leaderboard is reported in Appendix C). The highest result (0.731) has been obtained by three teams, i.e., SRCB (Zhang and Wang, 2022), TIB-VA (Hakimov et al., 2022) and PAIC (ZHI et al., 2022). The SRCB team (Zhang and Wang, 2022) adopted the same ensemble model used for Sub-task A. The system developed by TIB-VA is instead based on a Deep Learning model grounded on CLIP image and text features combined with a LSTM (Hochreiter and Schmidhuber, 1997), while PAIC (ZHI et al., 2022) did not provide any information about their approach. In second place, the ymf924 team did not provide any information about their approach, while in third place is the DD-TIG (Zhou et al., 2022) team with the same approach used for Sub-task A.

	Team Name
1	SRCB (Zhang and Wang, 2022)
	TIB-VA (Hakimov et al., 2022)
	PAIC (ZHI et al., 2022)
2	ymf924
3	DD-TIG (Zhou et al., 2022)
4	NLPros
5	QMUL
6	Unibo (Muti et al., 2022)
7	RubCSG (Yu et al., 2022)
8	AMS_ADRN (Li et al., 2022)
9	taochen (Tao and jae Kim, 2022)
10	ASRtrans (Rao and Rao, 2022)

Table 5: Top-10 teams in Sub-task B, ranked according to weighted-average F_1-score.

In general, the most predominant models for addressing Sub-task B are multi-class approaches, multi-task learning, and/or ensemble methods, where the feature space for learning has been derived either by image and text pre-trained models or by a joint embedding space.

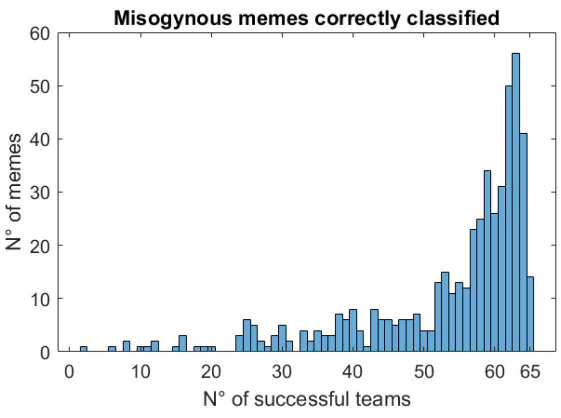
5 Error Analysis

In order to gain deeper insight into the prediction capabilities of the systems and delineate the open issues about the recognition and classification of misogynous memes, we conducted a detailed error analysis on both sub-tasks, considering all participating teams. The error distributions and the types of the most common errors in regards to the labels to be predicted are detailed in the following subsections. We considered memes misclassified by at least 25%, 50% and 75% of the teams, distinguishing False Positive (FP) and False Negative (FN), according to the labels available in each sub-task. For the memes misclassified by at least 75% of the teams, we reported the most frequent types of errors by analysing the visual and textual content of the memes.

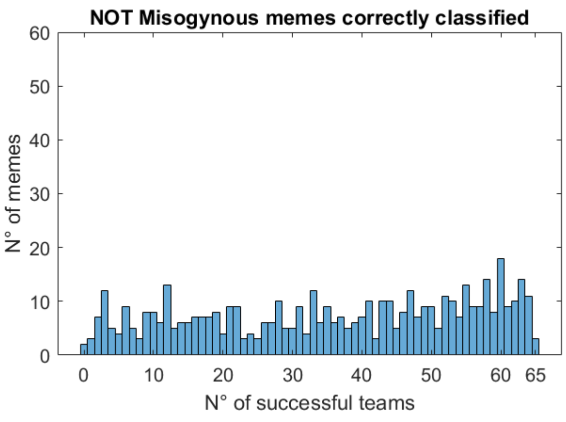
5.1 Sub-task A

In Figure 3, the distribution of correct classifications with respect to the number of successful teams is reported for misogynous and not misogynous memes. The distribution of correctly classified misogynous memes (Figure 3(a)) is uni-modal and peaked towards higher values, implying that most memes have been correctly classified by most teams. On the other hand, considering the not misogynous ones, the distribution is more uniform (Figure 3(b)), denoting that in general the models are more recall than precision oriented. There are 14 memes out of 500 (2.8%) correctly classified as misogynous by all the teams (Figure 3(a), last bin), while no one is misclassified by all the teams. In the worst case, only one misogynous meme was misclassified by 63 out of 65 teams.

In Table 6 the error distribution of Sub-task A is reported, considering the misclassification of misogynous memes and not misogynous ones sepa-



(a)



(b)

Figure 3: Distributions of correct classifications with respect to the number of successful teams for misogynous (a) and not misogynous (b) memes.

Teams	Misogynous memes predicted as NOT Misogynous (FN)		NOT Misogynous memes predicted as Misogynous (FP)		Overall misclassified memes (FP+FN)	
25% (16 teams)	128	25.60%	340	68.00%	468	46.80%
50% (33 teams)	46	9.20%	220	44.00%	266	26.60%
75% (49 teams)	12	2.40%	109	21.80%	121	12.10%

Table 6: Error distribution on Sub-task A.

rately, and finally the overall errors. In general, the percentage of classification errors of non misogynous memes are higher than misogynous ones, confirming that the methods are more precision than recall oriented. This suggests that most of the systems tends to be biased towards the misogyny category due to the presence of text or images that mislead the systems. Focusing on the memes misclassified by at least 75% of the teams, the most frequent types of errors can be summarised in the following paragraphs.

Misogynous memes predicted as NOT misogynous (FN). Twelve memes belong to this set. Five of them involve sexual objectification, that requires correlation of textual and visual content to classify. In particular, the meme depicted in Figure 4 is characterised by a neutral text and depicts a neutral object. In this case, the shape of the object together with the text needs to be correlated to grasp the sexual meaning. This meme was correctly classified by only 6 teams out of 65. Another group of misclassified memes, corresponding to one third of this set, is related to violence, both physical (visually represented), and sexual, which is less explicitly evoked.

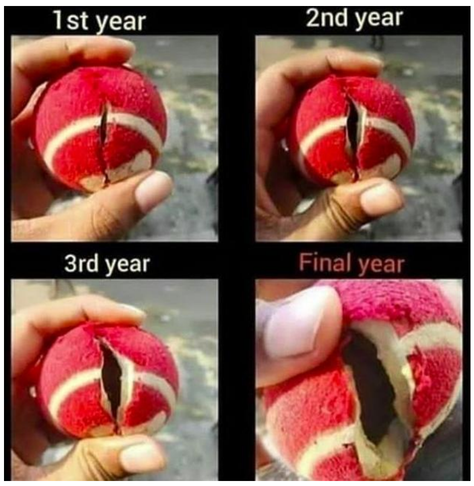


Figure 4: A misogynous meme classified as a non misogynous one (Raw image: 17013.jpg).

NOT Misogynous memes predicted as misogynous (FP). 109 NOT misogynous memes were incorrectly predicted by at least 75% of the teams. The majority of the misclassified memes contain textual or visual content that are often contained in misogynous memes. For example, 38% of the memes contain words and phrases such as “woman, man, fat, boobs, kitchen, dishwasher, chicks, make me a sandwich, . . . ”, and 31% depict close up images of women, which often emphasise the neck-

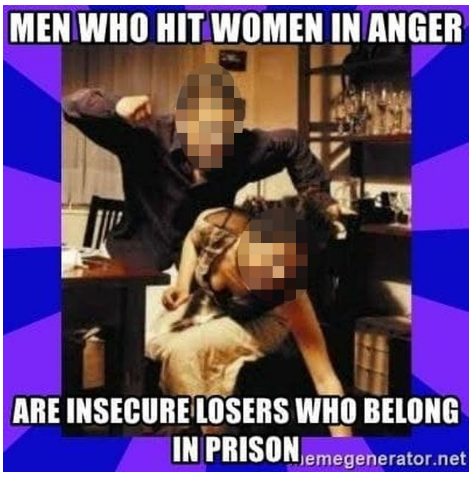

MEN WHO HIT WOMEN IN ANGER
ARE INSECURE LOSERS WHO BELONG IN PRISON
memegenerator.net

Figure 5: Example of meme with an antithetical content (Raw image: 15138.jpg).

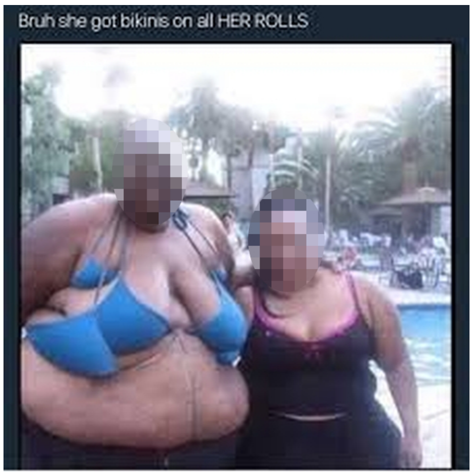


Figure 6: Most common example of Shaming meme misclassified as NOT Shaming (Raw image: 15559.jpg)

line, or depict faces with evident makeup. An interesting group of misclassified memes (7 out of 109) shows antithetical content. In general, most of the visual and textual information recall typical misogynous memes (with viral phrases such as "back to the kitchen" or depicting misogynous scenes such as physical violence), however additional information both visual and textual, with an opposite meaning, changes the overall message conveyed, as depicted in the example in Figure 5.

Memes featuring famous characters or actors who are often depicted associated to messages of all kinds, such as Ryan Gosling with the "hey girl" memes, Dwight Schrute or Willy Wonka, are also frequently misclassified (about 10%). Finally it is worth noting that other misclassified memes are those that convey feminist ideals and content.

5.2 Sub-task B

We report in Table 7, the error distribution of Subtask B, accordingly to the labels predicted (i.e., Stereotype, Violence, Shaming and Objectification). The first interesting insights involve the misogyny categories that are misclassified by at least 75% of the teams, in a ranked order: Objectification (14.60% of memes are wrongly classified by at least 31 teams in the over 41 participating teams), Stereotype (13.10%), Violence (3.30%) and Shaming (3.2%). A further interesting insight relates to the ability of the models with respect to the False Negative (FN) and the False Positive (FP) of each class. While for Shaming and Violence the percentage of FP (0.82% and 0.12% respectively) is much lower than the percentage of FN (17.2% and 20.92%), for Stereotype and Objectification the opposite is true, where FP (27.71% and 35.92% respectively) rates are much higher then FN (5.23% and 3.22%). We analysed the most predominant errors, with respect to each misogyny category.

Shaming. Regarding the first misogyny category, the most frequent error by at least 75% of the teams relates to the classification of Shaming memes as NOT Shaming (17.12%). The majority of the memes wrongly classified relates to the concept of *fat shaming* where overweight women are compared, implicitly or explicitly, to a narrow standard. An example of such errors is reported in Figure 6.

Violence. With the Violence category, the most frequent error by at least 75% of the teams relates to the classification of Violence memes as NOT Violence ones (20.92%). In this case, the majority of the memes wrongly classified as NOT Violence relates to the concept of *physical assault* typically depicted with a violent image (e.g., woman with bruises) but with neutral text (e.g., "don't tell her twice") or by a neutral image (e.g., standing men) coupled with a violent text (e.g., "women need a good beating once in a while"). An example of a misclassified violent meme is shown in Figure 7.

Stereotype. In the Stereotype category, the most frequent error by at least 75% of the teams relates to the classification of NOT Stereotype memes as Stereotype ones (27.71%). In this case, the most frequent misclassification concerns memes that are related to the concept of *men in the kitchen*, where the image typically represents men and the text is related to the stereotype of woman in kitchen ("cooking"). An example of such errors is re-

Teams	Shaming predicted as NOT Shaming (FN)		NOT Shaming predicted as Shaming (FP)		Overall misclassified Shaming memes (FP+FN)	
25% (11 teams)	92	63.01%	143	16.74%	235	23.50%
50% (21 teams)	59	40.41%	44	5.15%	103	10.30%
75% (31 teams)	25	17.12%	7	0.82%	32	3.20%
	Violence predicted as NOT Violence (FN)		**NOT Violence predicted as Violence (FP)**		**Overall misclassified Violence memes (FP+FN)**	
25% (11 teams)	90	58.82%	32	3.78%	122	12.20%
50% (21 teams)	65	42.48%	6	0.71%	71	7.10%
75% (31 teams)	32	20.92%	1	0.12%	33	3.30%
	Stereotype predicted as NOT Stereotype (FN)		**NOT Stereotype predicted as Stereotype (FP)**		**Overall misclassified Steretype memes (FP+FN)**	
25% (11 teams)	236	36.31%	278	79.43%	514	51.40%
50% (21 teams)	94	14.46%	190	54.29%	284	28.40%
75% (31 teams)	34	5.23%	97	27.71%	131	13.10%
	Objectification predicted as NOT Objectification (FN)		**NOT Objectification predicted as Objectification (FP)**		**Overall misclassified Objectification memes (FP+FN)**	
25% (11 teams)	151	23.16%	260	74.71%	411	41.10%
50% (21 teams)	65	9.97%	205	58.91%	270	27.00%
75% (31 teams)	21	3.22%	125	35.92%	146	14.60%

Table 7: Error distribution on Sub-task B.

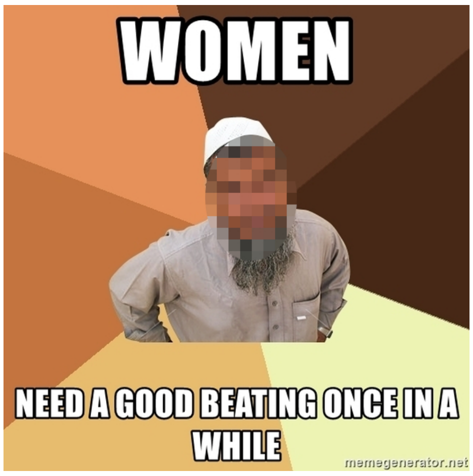


Figure 7: Most common example of Violence meme misclassified as NOT Violent (Raw image: 16067.jpg)

ported in Figure 8. The analysis of the errors in the stereotyped category is controversial and interesting. Some of the memes that our annotators have labelled as non-stereotypical could be considered expressions of benevolent sexism (Glick and Fiske, 1996). Benevolent sexism is a subtle form of prejudice, which apparently values women more than men but does it connecting this positive evaluation to their traditional roles. This is a manifestation of sexism that is difficult to detect and it is still not consensual in society. In fact, these memes were considered by our annotators not to be an expression of stereotype. The task team decided to keep the annotators' view that reflects the majority thinking in society today, however, the models seem to have detected benevolent sexism and the errors go in that direction. If models are only detecting the kitchen scenario or a more subtle form of prejudice is an intriguing question for future research.

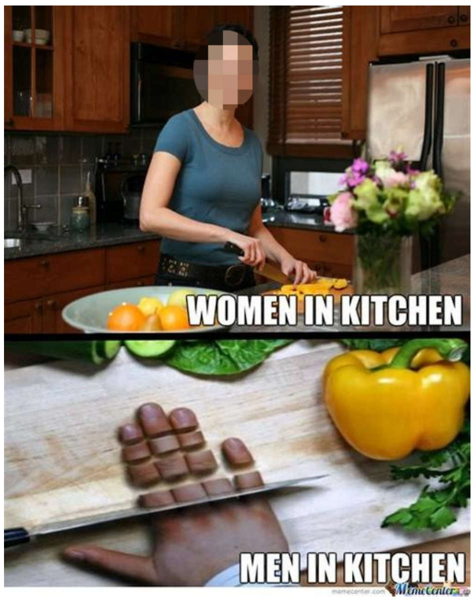


Figure 8: Most common example of NOT misogynous and NOT Stereotype meme misclassified as Stereotype (Raw image: 15137.jpg)

Objectification. In the Objectification category, the most frequent error by at least 75% of the teams relates to the classification of NOT Objectification memes as Objectification (35.92%). In this case, there is not a predominant archetype over the others that confounds the majority of the models.

6 Conclusions

The high number of participating teams at the MAMI challenge at SemEval-2022 confirms the growing interest of the research community not only in detecting abusive language but also pictorial content as sources of information. Overall, results and error analysis confirm that the detection of misogynous memes is challenging, with many open issues that need to be addressed. First of all, the fact that the most predominant error in misogyny recognition relates to the misclassification of NOT misogynous memes as misogynous ones suggests that some potential issues could be related to biased models. The research community is therefore encouraged to pay attention not only to accuracy metrics, but also to ensure models are unbiased before applying them in a real context. Another open issue relates to the capability of the systems to model the dynamics of the memes. Every day different memes, with different images and different text are generated on the web and shared online.

References

Samyak Agrawal and Radhika Mamidi. 2022. Lastresort at semeval-2022 task 5: Towards misogyny identification using visual linguistic model ensembles and task-specific pretraining. In *The 16th International Workshop on Semantic Evaluation*.

Maja Brandt Andreasen. 2021. 'rapeable'and 'unrapeable'women: the portrayal of sexual violence in internet memes about# metoo. *Journal of Gender Studies*, 30(1):102–113.

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Ayme Arango, Jesus Perez-Martin, and Arniel Labrada. 2022. Hateu at semeval-2022 task 5: Multimedia automatic misogyny identification. In *The 16th International Workshop on Semantic Evaluation*.

Giuseppe Attanasio, Debora Nozza, and Federico Bianchi. 2022. MilaNLP at semeval-2022 task 5: Using perceiver IO for detecting misogynous memes with text and image modalities. In *The 16th International Workshop on Semantic Evaluation*.

Shubham Kumar Barnwal, Ritesh Kumar, and Rajendra Pamula. 2022. IIT DHANBAD CODECHAMPS at semeval-2022 task 5: MAMI - multimedia automatic misogyny identification. In *The 16th International Workshop on Semantic Evaluation*.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Mitra Behzadi, Ali Derakhshan, and Ian Harris. 2022. Mitra behzadi at semeval-2022 task 5 : Multimedia automatic misogyny identification method based on CLIP. In *The 16th International Workshop on Semantic Evaluation*.

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

Lei Chen and Hou Wei Chou. 2022. RIT boston at semeval-2022 task 5: Multimedia misogyny detection by using coherent visual and language features from CLIP model and data-centric AI principle. In *The 16th International Workshop on Semantic Evaluation*.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Pablo Cordon, Pablo Gonzalez Diaz, Jacinto Mata, and Victoria Pachón. 2022. I2c at semeval-2022 task 5: Identification of misogyny in internet memes. In *The 16th International Workshop on Semantic Evaluation*.

Statista Research Department. 2019. Share of u.s. adults who use social media 2019, by gender.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Alice H Eagly and Antonio Mladinic. 1989. Gender stereotypes and attitudes toward women and men. *Personality and social psychology bulletin*, 15(4):543–558.

Paolo Rosso Elisabetta Fersini, Debora Nozza. 2020. Ami @ evalita2020: Automatic misogyny identification. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*. CEUR.org.

Guy Emerson, Natalie Schluter, Gabriel Stanovsky, Ritesh Kumar, Alexis Palmer, and Nathan Schneider. 2022. *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tracie Farrell, Miriam Fernandez, Jakub Novotny, and Harith Alani. 2019. Exploring misogyny across the manosphere in reddit. In *Proceedings of the 10th ACM Conference on Web Science*, pages 87–96.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the evalita 2018 task on automatic misogyny identification (ami). In *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR.org.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. In *IberEval@ SEPLN*, pages 214–228.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Mary Anne Franks. 2011. Unwilling avatars: Idealism and discrimination in cyberspace. *Colum. J. Gender & L.*, 20:224.

Simona Frenda, Bilal Ghanem, Manuel Montes-y Gómez, and Paolo Rosso. 2019. Online hate speech against women: Automatic identification of misogyny and sexism on twitter. *Journal of Intelligent & Fuzzy Systems*, 36(5):4743–4752.

José Antonio García-Díaz, Camilo Caparros-Laiz, and Rafael Valencia-García. 2022. UMUTeam at semeval-2022 task 5: Combining image and textual embeddings for multi-modal automatic misogyny identification. In *The 16th International Workshop on Semantic Evaluation*.

Francesca Gasparini, Ilaria Erba, Elisabetta Fersini, and Silvia Corchs. 2018. Multimodal classification of sexist advertisements. In *ICETE (1)*, pages 565–572.

Peter Glick and Susan T Fiske. 1996. The ambivalent sexism inventory: Differentiating hostile and benevolent sexism. *Journal of Personality and Social Psychology*, 70(3):491–512.

Qin Gu, Nino Meisinger, and Anna-Katharina Dick. 2022a. Qinian at semeval-2022 task 5: Multi-modal misogyny detection and classification. In *The 16th International Workshop on Semantic Evaluation*.

Yimeng Gu, Ignacio Castro, and Gareth Tyson. 2022b. MMVAE at semeval-2022 task 5: A multi-modal multi-task VAE on misogynous meme detection. In *The 16th International Workshop on Semantic Evaluation*.

Mohammad Habash, yahya Daqour, Malak AG Abdullah, and Mahmoud Al-Ayyoub. 2022. YMAI at semeval-2022 task 5: Detecting misogyny in memes using visualBERT and MMBT multimodal pre-trained models. In *The 16th International Workshop on Semantic Evaluation*.

Sherzod Hakimov, Gullal Singh Cheema, and Ralph Ewerth. 2022. TIB-VA at semeval-2022 task 5: A multimodal architecture for the detection and classification of misogynous memes. In *The 16th International Workshop on Semantic Evaluation*.

Chao Han, Jin Wang, and Xuejie Zhang. 2022. YNU-HPCC at semeval-2022 task 5: Multi-modal and multi-label emotion classification based on LXMERT. In *The 16th International Workshop on Semantic Evaluation*.

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

Álvaro Huertas-García, Helena Liz, Guillermo Villar-Rodríguez, Alejandro Martín, Javier Huertas-Tato, and David Camacho. 2022. AIDA-UPM at semeval-2022 task 5: Exploring multimodal late information fusion for multimedia automatic misogyny identification. In *The 16th International Workshop on Semantic Evaluation*.

Milan Kalkenings and Thomas Mandl. 2022. University of hildesheim at semeval-2022 task 5: Combining deep text and image models for multimedia misogyny detection. In *The 16th International Workshop on Semantic Evaluation*.

Miroslav Kubat. 1999. Neural networks: a comprehensive foundation by simon haykin, macmillan, 1994, isbn 0-02-352781-7. *The Knowledge Engineering Review*, 13(4):409–412.

Da Li, Ming Yi, and Yukai He. 2022. AMS_ADRN at semeval-2022 task 5: A suitable image-text multimodal joint modeling method for multi-task misogyny identification. In *The 16th International Workshop on Semantic Evaluation*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2020a. What does BERT with vision look at? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5265–5275, Online. Association for Computational Linguistics.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020b. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

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

Gustavo Lorentz and Viviane Moreira. 2022. INF-UFRGS at semeval-2022 task 5: analyzing the performance of multimodal models. In *The 16th International Workshop on Semantic Evaluation*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 13–23.

Shankar Mahadevan, Sean Benhur, Roshan Nayak, Malliga Subramanian, Kogilavani Shanmugavadivel, Kanchana Sivanraju, and Bharathi Raja Chakravarthi. 2022. Transformers at semeval-2022 task 5: A feature extraction based approach for misogynous meme detection. In *The 16th International Workshop on Semantic Evaluation*.

Paridhi Maheshwari and Sharmila Reddy Nangi. 2022. Teamotter at semeval-2022 task 5: Detecting misogynistic content in multimodal memes. In *The 16th International Workshop on Semantic Evaluation*.

Ahmed Mahmoud Mahran, Carlo Alessandro Borella, and Konstantinos Perifanos. 2022. Codec at semeval-2022 task 5: Multi-modal multi-transformer misogynic meme classification framework. In *The 16th International Workshop on Semantic Evaluation*.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Arianna Muti, Katerina Korre, and Alberto Barrón-Cedeño. 2022. UniBO at semeval-2022 task 5: A multimodal bi-transformer approach to the binary and fine-grained identification of misogyny in memes. In *The 16th International Workshop on Semantic Evaluation*.

Marinella Paciello, Francesca D'Errico, Giorgia Saleri, and Ernestina Lamponi. 2021. Online sexist meme and its effects on moral and emotional processes in social media. *Computers in Human Behavior*, 116:106655.

Andrei Paraschiv, Mihai Dascalu, and Dumitru Clementin Cercel. 2022. UPB at semeval-2022 task 5: Enhancing UNITER with image sentiment and graph convolutional networks for multimedia automatic misogyny identification. In *The 16th International Workshop on Semantic Evaluation*.

Rebecca Passonneau. 2006. Measuring agreement on set-valued items (MASI) for semantic and pragmatic annotation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Flor-Miriam Plaza-Del-Arco, M Dolores Molina-González, L Alfonso Ureña-López, and M Teresa Martín-Valdivia. 2020. Detecting misogyny and xenophobia in spanish tweets using language technologies. *ACM Transactions on Internet Technology (TOIT)*, 20(2):1–19.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Tathagata Raha, Sagar Joshi, and Vasudeva Varma. 2022. IIITH at semeval-2022 task 5: A comparative study of deep learning models for identifying misogynous memes. In *The 16th International Workshop on Semantic Evaluation*.

Ailneni Rakshitha Rao and Arjun Rao. 2022. ASRtrans at semeval-2022 task 5: Transformer-based models for meme classification. In *The 16th International Workshop on Semantic Evaluation*.

Jason Ravagli and Lorenzo Vaiani. 2022. JRLV at semeval-2022 task 5: The importance of visual elements for misogyny identification in memes. In *The 16th International Workshop on Semantic Evaluation*.

Edgar Roman-Rangel, Jorge Fuentes-Pacheco, and Jorge Hermosillo Valadez. 2022. Vision-language approach to recognize misogynous content in memes. In *The 16th International Workshop on Semantic Evaluation*.

Gagan Sharma, Gajanan Sunil Gitte, Shlok Goyal, and Raksha Sharma. 2022a. IITR codebusters at semeval-2022 task 5: Misogyny identification using transformers. In *The 16th International Workshop on Semantic Evaluation*.

Mayukh Sharma, Ilanthenral Kandasamy, and Vasantha W B. 2022b. R2d2 at semeval-2022 task 5: Attention is only as good as its values! a multimodal system for identifying misogynist memes. In *The 16th International Workshop on Semantic Evaluation*.

Limor Shifman. 2013. Memes in a Digital World: Reconciling with a Conceptual Troublemaker. *Journal of Computer-Mediated Communication*, 18(3):362–377.

Si Si, Huan Zhang, S Sathiya Keerthi, Dhruv Mahajan, Inderjit S Dhillon, and Cho-Jui Hsieh. 2017. Gradient boosted decision trees for high dimensional sparse output. In *International conference on machine learning*, pages 3182–3190. PMLR.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Rajalakshmi Sivanaiah, Angel Deborah S, Sakaya Milton Rajendram, and Mirnalinee T T. 2022. TechSSN at semeval-2022 task 5: Multimedia automatic misogyny identification using deep learning models. In *The 16th International Workshop on Semantic Evaluation*.

Harshvardhan Srivastava. 2022. Poirot at semeval-2022 task 5: Leveraging graph network for misogynistic meme detection. In *The 16th International Workshop on Semantic Evaluation*.

Dawn M Szymanski, Lauren B Moffitt, and Erika R Carr. 2011. Sexual objectification of women: Advances to theory and research $1\psi 7$. *The Counseling Psychologist*, 39(1):6–38.

Chen Tao and Jung jae Kim. 2022. taochen at semeval-2022 task 5: Multimodal multitask learning and ensemble learning. In *The 16th International Workshop on Semantic Evaluation*.

Kathleen Van Royen, Karolien Poels, Heidi Vandebosch, and Michel Walrave. 2018. Slut-shaming 2.0. In *Sexting*, pages 81–98. Springer.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie-vil: Knowledge enhanced vision-language representations through scene graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3208–3216.

Wentao Yu, Benedikt Boenninghoff, Jonas Röhrig, and Dorothea Kolossa. 2022. RubCSG at semeval-2022 task 5: Ensemble learning for identifying misogynous MEMEs. In *The 16th International Workshop on Semantic Evaluation*.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of SemEval*.

Jing Zhang and Yujin Wang. 2022. SRCB at semeval-2022 task 5: Pretraining based image to text late sequential fusion system for multimodal misogynous meme identification. In *The 16th International Workshop on Semantic Evaluation*.

JIN MEI ZHI, Zhou Mengyuan, Mengfei Yuan, Dou Hu, Xiyang Du, Lianxin Jiang, Yang Mo, and XiaoFeng Shi. 2022. PAIC at semeval-2022 task 5: Multimodal misogynous detection in MEMES with multitask learning and multi-model fusion. In *The 16th International Workshop on Semantic Evaluation*.

Ziming Zhou, Han Zhao, Jingjing Dong, Ning Ding, Xiaolong Liu, and Kangli Zhang. 2022. DD-TIG at semeval-2022 task 5: Investigating the relationships between multimodal and unimodal information in misogynous memes detection and classification. In *The 16th International Workshop on Semantic Evaluation*.

A Annotation Guidelines

We report here the annotation guidelines provided to the annotators participating in the crowdsourcing annotation process of the collected memes. Since some memes contain sensitive content, we provided an explicit advisory message to the annotators.

A.1 Overview

The job aims at labelling English memes shared by users on the web as misogynous or not misogynous. The first step is about collecting socio-demographic information about the annotators:

- Gender: indicate your gender as female, male, unspecified
- Age: please choose your age range between 18-15, 25-35, 35-45, 45-60, over-60
- Location: please indicate your country of birth

The second step is about misogyny labelling. Annotators have to decide whether a meme is misogynous or not. If a meme is labelled as misogynous, then two other questions will be answered:

- Type of misogyny: the annotator should indicate (multiple choice) if the meme represents shaming, stereotype, objectification and/or violence.
- Misogyny rating: the annotator should provide a rating about how much the meme is misogynous using stars, i.e. *, ** or ***.

A.2 Guidelines and examples

Misogyny Labelling. Looking at a meme at a time, annotators should label it as misogynous or not according to the following definitions:

- ***Misogynous***: a meme is misogynous if it conceptually describes an offensive, sexist or hateful scene (weak or strong, implicitly or explicitly) having as target a woman or a group of women. Misogyny can be expressed in the form of shaming, stereotype, objectification and/or violence.
- ***Not Misogynous***: a meme that does not express any form of hate against women.

Remark: a meme is NOT misogynous if it is conceptually not related to women or even if it is related to women, but it does not represent an offensive, sexist or hateful concept against women.

Type of misogyny. If a meme is considered misogynous, then the annotator has to choose one or more types of misogynous categories, according to the following definitions:

- ***Shaming:*** memes aimed at insulting and offending women because of some characteristics of the body. These types of misogynous memes are related to denigrating the physical appearance of women (body shaming).
- ***Stereotype*:** memes are aimed at representing a fixed idea or set of characteristics assigned to women. These types of memes convey the image of women according to their role in the society (i.e., Role Stereotyping), to her personality traits and domestic behaviours (i.e., Gender Stereotyping) or to fixed ideological characteristics related to women's rights (i.e., Feminism Stereotype).
- ***Objectification:*** it is a practice of seeing and/or treating a woman like an object. These types of memes usually report an over-appreciation of women's physical appeal, depicting woman as an object (sexual objectification or human being without any value as a person).
- ***Violence:*** indicates a physical or verbal violence represented by textual or visual content. These types of misogynous memes are aimed at showing violence against women or at alluding to the intent of physically assert power over women.

Misogyny Rating. If a meme is considered misogynous then the annotator has to indicate, according to his/her opinion, how misogynistic it is using a 1 to 3 ratings: * indicates weak misogyny, ** means medium misogyny, *** means strong misogyny.

B Team Information

We report here the details provided by those teams that have responded to a request for team information.

Team Name	Country	Members
InfUfrgs	Brazil	1
HateU	Chile	3
AMS_ADRN	China	3
DD-TIG		1
SRC-B		6
YNU-HPCC		3
TIB-VA	Germany	2
qinian		3
Hildesheim		1
RubCSG		4
TechSSN	India	4
IITR CodeBusters		3
IIT DHANBAD CODECHAMPS		1
LastResort		1
SSN_NLP_MLRG		2
Gini_us		3
ASRtrans		1
IIITG-ADBU		1
Transformers		7
R2D2		3
Poirot		1
Tathagata Raha		1
JRLV	Italy	2
Unibo		3
Triplo7		1
YMAI	Jordan	2
UAEM-ITAM	Mexico	3
UPB	Romania	1
taochen	Singapore	1
UMUTeam	Spain	1
AIDA-UPM		6
I2C		1
NLPros	UK	5
MMVAE		1
codec		1
QMUL		1
Mitra Behzadi	USA	1
RIT Boston		2
Charicfc		1
Stanford MLab		5
TeamOtter		2

Table 8: Team characteristics.

C Leader-boards

C.1 Leader-board of Sub-task A

We report in Table 9 the leader-board for Sub-task A. Team Names marked with * have submitted team name and additional information for further analysis and discussion. For those teams that have not provided the Team Name, we maintained the user name used on Codalab for submitting their predictions.

To produce the reported leader-board, we filtered the ranking defined by the evaluated metrics to maintain only the highest achieved score per group. Afterwards, we scrolled through this ranking from top to bottom in order to create clusters based on the obtained scores and the statistical difference resulting from the application of the McNemar test (McNemar, 1947).
In particular, starting from the first entry in the ranking, we have included in the same cluster the groups that presented (1) the same score or (2) had a statistical equality in performance.
As stated before, statistical equality was computed with a pairwise analysis performed with the McNemar test: we evaluated the equality in performance of the analysed algorithm with the algorithm that obtained the highest score within the cluster, considering a value of alpha equal to 0.05. According to this criterion, in the event that the algorithm under analysis could not be included in the cluster, a new one was created; the subsequent ones would have been compared with the latter.

Notice that in the leader-board were maintained all the baseline results for comparison.

C.2 Leader-board of Sub-task B

We report in Table 10 the leader-board for Sub-task B. Team Names marked with * have submitted team name and additional information for further analysis and discussion. For those teams that have not provided the Team Name, we maintained the user name used on Codalab for submitting their predictions.

To obtain the reported leader-board, a similar approach to the one used for Sub-task A has been adopted. A McNemar test (McNemar, 1947) was adopted to evaluate the similarity in performance for the identification of every single type of misogyny. Two algorithms have been considered statistically equal in performance if there was statistical significance in all 4 tests (i.e., if there was a statistical significance for the performance related to all 4 types of misogyny). Thus, a difference in performance for the prediction of only one of the four types has been valued sufficient to consider the analysed algorithm as statistically unequal. As for Sub-task A, the grouping depends on statistical equality and on the scores obtained.

Notice that in the leader-board were maintained all the baseline results for comparison.

Leaderboard Sub-task A		
	Team Name	**Macro-average F_1-score**
1	SRCB* (Zhang and Wang, 2022)	0.834
2	DD-TIG* (Zhou et al., 2022)	0.794
	RIT Boston* (Chen and Chou, 2022)	0.778
	NLPros*	0.771
3	ASRtrans* (Rao and Rao, 2022)	0.761
	Poirot* (Srivastava, 2022)	0.759
	R2D2* (Sharma et al., 2022b)	0.757
	PAIC (ZHI et al., 2022)	0.755
	ymf924	0.755
	RubCSG* (Yu et al., 2022)	0.755
	hate-alert	0.753
	AMS_ADRN* (Li et al., 2022)	0.746
	TIB-VA* (Hakimov et al., 2022)	0.734
4	union	0.727
	Unibo* (Muti et al., 2022)	0.727
	MMVAE* (Gu et al., 2022b)	0.723
	YMAI* (Habash et al., 2022)	0.722
	Transformers* (Mahadevan et al., 2022)	0.718
	taochen* (Tao and jae Kim, 2022)	0.716
	codec* (Mahran et al., 2022)	0.715
	QMUL*	0.714
	UPB* (Paraschiv et al., 2022)	0.714
	HateU* (Arango et al., 2022)	0.712
	yuanyuanya	0.708
	Triplo7* (Attanasio et al., 2022)	0.699
	InfUfrgs* (Lorentz and Moreira, 2022)	0.698
	Mitra Behzadi* (Behzadi et al., 2022)	0.694
	Gini_us*	0.692
5	riziko	0.687
	UMUTeam* (García-Díaz et al., 2022)	0.687
	Tathagata Raha* (Raha et al., 2022)	0.687
	LastResort* (Agrawal and Mamidi, 2022)	0.686
	TeamOtter* (Maheshwari and Nangi, 2022)	0.679
	ShailyDesai	0.677
	JRLV* (Ravagli and Vaiani, 2022)	0.670
	I2C* (Cordon et al., 2022)	0.665
	qinian* (Gu et al., 2022a)	0.665
	A.111	0.662
	IITR CodeBusters* (Sharma et al., 2022a)	0.662
	YNU-HPCC* (Han et al., 2022)	0.662
	WeiLW	0.661
	SSN_NLP_MLRG*	0.658
	UNIBUC-FMI	0.657
6	IIT DHANBAD CODECHAMPS* (Barnwal et al., 2022)	0.656
	Sattiy	0.655
	lianlio	0.654
	thisisatharva	0.650
	Baseline_Hierarchical_M.	0.650

Table 9 Continued from previous page

Leaderboard Sub-task A		
	Team Name	**Macro-average F_1-score**
6	IIITG-ADBU*	0.649
	UAEM-ITAM* (Roman-Rangel et al., 2022)	0.641
	Baseline_Image	0.640
	Baseline_Text	0.639
	Yet	0.639
	RaNdom	0.638
	AIDA-UPM* (Huertas-García et al., 2022)	0.636
	vishesh_gupta	0.634
	Levante	0.634
	Aily	0.632
	Charicfc*	0.620
	Stanford MLab*	0.619
7	rhitabrat	0.609
	Will To Live	0.606
	Hildesheim* (Kalkenings and Mandl, 2022)	0.603
	SakshiSingh	0.579
8	*Baseline_Image_Text*	0.543
	areen	0.524
	TechSSN* (Sivanaiah et al., 2022)	0.522
9	UET	0.481
10	*Baseline_Flat_Multilabel*	0.437

Table 9: Leader-board of Sub-task A.

Leaderboard of Sub-task B		
	Team Name	**Weighted-average F_1-score**
1	SRCB* (Zhang and Wang, 2022)	0.731
	TIB-VA* (Hakimov et al., 2022)	0.731
	PAIC (ZHI et al., 2022)	0.731
	ymf924	0.730
2	DD-TIG* (Zhou et al., 2022)	0.728
	NLPros*	0.720
3	QMUL*	0.713
4	Unibo* (Muti et al., 2022)	0.710
5	RubCSG* (Yu et al., 2022)	0.709
5	AMS_ADRN* (Li et al., 2022)	0.708
6	taochen* (Tao and jae Kim, 2022)	0.706
7	ASRtrans* (Rao and Rao, 2022)	0.705
8	codec* (Mahran et al., 2022)	0.698
9	Transformers* (Mahadevan et al., 2022)	0.695
10	Triplo7* (Attanasio et al., 2022)	0.693
	LastResort* (Agrawal and Mamidi, 2022)	0.692
	R2D2* (Sharma et al., 2022b)	0.690
	hate-alert	0.690
11	RIT Boston* (Chen and Chou, 2022)	0.689
12	Mitra Behzadi* (Behzadi et al., 2022)	0.681

Table 10 Continued from previous page

Leaderboard Sub-task B		
	Team Name	**Weighted-average F_1-score**
13	TeamOtter*(Maheshwari and Nangi, 2022)	0.680
14	Tathagata Raha* (Raha et al., 2022)	0.679
15	UPB* (Paraschiv et al., 2022)	0.673
16	riziko	0.668
17	UMUTeam* (García-Díaz et al., 2022)	0.663
18	UAEM-ITAM* (Roman-Rangel et al., 2022)	0.646
19	RaNdom	0.643
	qinian* (Gu et al., 2022a)	0.637
	UNIBUC-FMI	0.637
20	IITR CodeBusters* (Sharma et al., 2022a)	0.635
21	MMVAE* (Gu et al., 2022b)	0.634
	Yet	0.634
22	YNU-HPCC* (Han et al., 2022)	0.633
	Poirot* (Srivastava, 2022)	0.632
23	AIDA-UPM* (Huertas-García et al., 2022)	0.629
24	*Baseline_Hierarchical_M.*	0.621
25	YMAI* (Habash et al., 2022)	0.592
26	yuanyuanya	0.584
27	Stanford MLab*	0.563
28	UET	0.499
29	TechSSN* (Sivanaiah et al., 2022)	0.467
30	*Baseline_Flat_Multilabel*	0.421

Table 10: Leader-board of Sub-task B.

Transformers at SemEval-2022 Task 5: A Feature Extraction based Approach for Misogynous Meme Detection

Shankar Mahadevan[1], **Sean Benhur**[2], **Roshan Nayak**[3], **Malliga Subramanian**[4], **Kogilavani Shanmugavadivel**[4], **Kanchana Sivanraju**[2], **Bharathi Raja Chakravarthi**[5]
[1] Thiagarajar College of Engineering, India [2]PSG College of Arts and Science, India
[3]B.M.S College of Engineering, India [4]Kongu Engineering College, India
[5]Insight SFI Research Centre for Data Analytics,National University of Ireland, Galway
shankarmahadevan12901@gmail.com[1], seanbenhur@gmail.com[2], roshannayak610@gmail.com[3]
mallinishanth72@gmail.com[4], kogilanvani.sv@gmail.com[4], kanachana@psgcas.ac.in[2],
bharathi.raja@insight-centre.org[5]

Abstract

Social media is an idea created to make the world smaller and more connected. Recently, it has become a hub of fake news and sexist memes that target women. Social Media should ensure proper women's safety and equality. Filtering such information from social media is of paramount importance to achieving this goal. In this paper, we describe the system developed by our team for SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. We propose a multimodal training methodology that achieves good performance on both the subtasks, ranking 4th for Subtask A (0.718 macro F1-score) and 9th for Subtask B (0.695 macro F1-score) while exceeding the baseline results by good margins. The code will be available here[1]

1 Introduction

With the rising usage of social media and the Internet, it is tougher to establish an inclusive and welcoming community among users. Offensive speech, hate speech, and targeted insult have been increasing among users, disturbing everyone. With the rising utilization of the Internet in a pandemic, hate speech prevalence on the Internet is also increased. Online hate speech with targeted discrimination also creates threats and crimes offline.

Among these, online misogyny or sexist comments have been increasing among women (Salter et al., 2018), which includes name-calling, sexual threats, shaming. This emphasizes the need for specialized automatic misogyny detection in online platforms. Platforms such as Twitter [2] and Facebook already have policies for banning hateful content. However, these systems are primarily through manual methods and might not scale well for large users and multimodal content. Moreover, hate speech is also prevalent in multimodal form since most social media platforms support Images, text, audio and video content. These memes have been popular among users to express their opinions since people express information through memes, GIFs, and videos. But, unfortunately, this also causes the rise of multimodal hate, and offensive content, which is disturbing to users (Suryawanshi et al., 2020). This includes misogynistic posts, which are targeted towards women.

Previous works on Misogyny detection have been primarily focusing on one modality, which is text (Pamungkas et al., 2020). Misogyny detection in text falls under an area of text classification. Text classification methods such as BERT, LSTM, Naive Bayes have been used to detect misogynistic comments. In this work, we have used the provided data, which contains both images of the memes and the extracted OCR text from the memes.

This paper describes our submission for the task; we have used multiple concatenation-based fusion models and ensembled them for the final submission.

[1]https://github.com/shankrmahadevan/semevalmami2022

[2]https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy

Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 550 - 554
July 14-15, 2022 ©2022 Association for Computational Linguistics

2 Related Work

The past works have concentrated on collecting the dataset from popular social media sites such as Facebook, Reddit, and Twitter. Recent statistics do not simply focus on hate but also on the kind of hate the meme attempts to spread. Work has also been done in detecting offensive memes using various pre-trained models. (Fersini et al., 2019) presents a novel dataset for the sexist meme classification task. Sexism could be of several forms that could be categorized based on the context of the caption and the objects on the meme. The main types of sexism against women addressed in the dataset are shaming, stereotypes, objectification and violence. The research paper largely focused on a comparison between the unimodal and multimodal classifiers. The article has attempted to answer various research concerns such as whether unimodal architectures can predict the target correctly, will merging the features of both text and picture capture the inherent complexity of the sexist memes, and which one of the two modalities dominates the other. The research discovered that unimodal classifiers have shown that textual information is an excellent indicator, whereas visual information is a poor indicator to identify sexist memes. This study between unimodal and multimodal showed that unimodal architectures performed better than multimodel architectures.

In the paper, (Zia et al., 2021) the analysis is done on the dataset that focuses beyond hateful or not hateful by annotating the hate meme dataset further by the kind of hate the meme is actually spreading. This would help in understanding the meme and the intention of the person who created the meme better rather than just labelling it as hateful or not. The paper focused on two tasks. The first task was to identify the kind of hate the meme intended to spread. The second task was to detect the type of attack the meme did on a particular group such as slurs, inferiority, and mocking. Models such as CLIP (Contrastive Language Image Pre-Training) and LASER (Language Agnostic SEntence Representations), LaBSE (Language agnostic BERT Sentence Embedding) were used to extract features from the image and text. The paper concluded that multimodel architectures outperformed unimodal architecture. The multimodal, concatenated textual features (CLIP, LASER, and LaBSE) and visual features (CLIP) was the best performing model with AUROC of 0.96 and 0.97 for task A and task B, respectively.

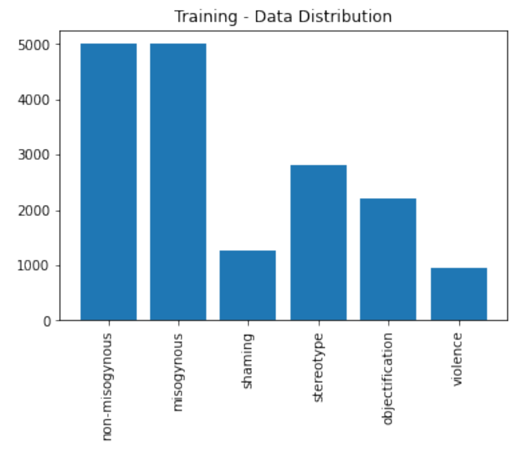


Figure 1: Training data distribution. It can be seen that the positive and negative classes are in equal proportions.

(Guest et al., 2021) introduced a taxonomical dataset of 6,383 samples from Reddit. The dataset has a three-level taxonomy which makes this dataset very different from what already exists. The first level is a binary classification between misogynistic content and non-misogynistic content. The second level corresponds to the subtypes of misogynistic and non-misogynistic content. Misogynistic content categories include misogynistic pejoratives, misogynistic treatment, misogynistic derogation and gendered personal attacks against women. Non-misogynistic content categories include counter speech against misogyny, non-misogynistic personal attacks and None of the categories. In the third level, additional flags for some of the second-level categories have been defined. BERT based models were trained on the dataset to achieve a test accuracy of 0.93.

3 Dataset

The dataset provided for the competition (Fersini et al., 2022) consisted of images of memes and OCR extracted text and labels for both subtask A and B. For Subtask A, the binary label of misogynous is given; for Subtask B, four labels were given: they are shaming, stereotype, objectification and violence, each of them containing binary values 0 or 1. The provided dataset contains 10,000 training images, 100 validation images and 1000 images for test set submission. The training dataset was randomly shuffled and split into five-folds, with each fold containing 2000 images each. The first four folds were used to train the model, and the last fold, along with the eval dataset provided, was chosen as the validation dataset to improve the model performance. The final number of samples in training, validation and test set are reported in Table 1. The Data distribution is given in Figure 1.

4 Preprocessing

Since the text was extracted from the memes using OCR tools, much noise in the text had to be cleaned manually. First, all internet links, stopwords and Twitter user handles were removed from the text. Then, the text was lemmatized using a word-net based lemmatizer. The text truncation length was set to 256. An interesting observation in the dataset was that memes that were not misogynistic in nature did belong to the other four classes. So, it was evident that a meme might not be misogynistic yet belong to any of the other subcategories.

5 Methodology

5.1 Models

Since this topic was multimodal in nature, we finetuned multiple text-based models and image-based models to handle this task. Convolutional Neural Networks (CNNs) excel in image classification challenges due to their intrinsic spatial inductive bias. CNNs have been leading the computer vision research arena for the last two decades due to their superior spatial comprehension ability. The CNN based image models chosen for this task are: InceptionV3 (Szegedy et al., 2015) and EfficientNetB7 (Tan and Le, 2020) from the TensorFlow library. The BERT (Devlin et al., 2019) model was used as the text feature extraction backbone. We also tried finetuning CLIP (Radford et al., 2021) for this task, as it was trained in a multimodal fashion. For both CLIP and other multimodal models we added a fully connected layer with softmax for classification.Another approach was to extract a set of embeddings from State-of-the-Art Text and Image models and classify the features using Support Vector Machines (SVMs). The models selected for this approach were: XLM-RoBERTa (Conneau et al., 2020), DistilBERT(Sanh et al., 2020), ResNext (Xie et al., 2017) and Data-efficient Image Transformer (Touvron et al., 2021).

5.2 Experiment Setup

We implement our multimodal training in TensorFlow using Tensor Processing Units (TPUs) offered by Google Colab for training. TPUs greatly shortened the time required to conduct numerous tests and hyper-parameter optimization. All the photos were resized to 256x256. The TensorFlow version of the BERT and CLIP models from the transformers library was used. A CUDA-accelerated implementation of SVM from the cuML library created by NVIDIA was used in the SVM training.

Split	No. of Samples
Training	8,000
Validation	2,100
Test	1,000

Table 1: Dataset Split

Image Augmentation methods Typically, CNNs are trained using millions of images to attain good accuracy. However, since the number of photos available in the dataset was less in nature, image augmentation methods were added to generate synthetic augmented images and thus boost the amount of data utilized to train the model. This ensures that the model better generalizes to the patterns present in the image modality. We employ (i) random resizing, (ii) random cropping, (iii) random horizontal flipping and (iv) random vertical flipping as the augmentation methods.

The SVMs were trained using a single K80 GPU provided by Google Colab.

5.3 Multimodal Training

The Multimodal training illustrated in this section follows the procedure shown in Figure 2. The model using InceptionV3 and BERT backbone is termed as Model A, EfficientNet B7 and BERT as Model B, CLIP Image and CLIP Text Backbone as Model C. Models A and B use Adam optimizer with a base learning rate of 1e-06 and a linear learning rate decay. Model C uses Adam optimizer with a base learning rate of 6e-05 and a linear learning rate decay. Image preprocessing is done as provided by the model authors. Text cleaning and tokenization is performed for feeding to the text model. All the models are trained for 50 epochs with an Early Stopping callback to terminate the training when the model does not learn any discriminatory features and/or overfits to the training set. A fully connected layer is added at the end to perform classification.

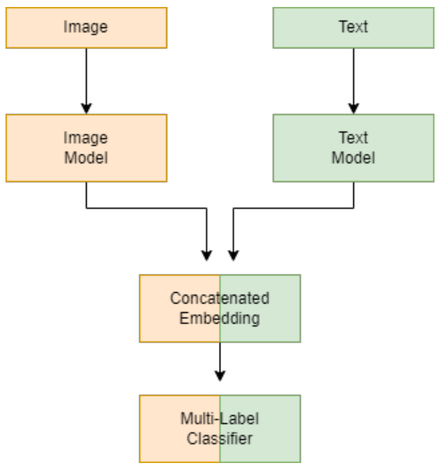


Figure 2: **An overview of the multimodal training approach.** Late fusion is adopted for effective classification.

5.4 SVM Training

In the multimodal training approach, complete models were finetuned for classification. In this approach, the pre-trained image and text models are used as feature extractors alone, and the features are supplied to SVMs for classification. Since the time complexity of training SVMs increases quadratically with respect to the available data, when the data becomes higher than tens of thousands of samples, it practically becomes impossible to train SVMs on CPUs. Since there is a significant amount of data in the training set, SVMs accelerated using CUDA from the cuML library were utilized for training the SVMs. Due to the highly parallel nature of GPU computation, the time required to train the SVM is reduced to seconds. Since SVM is a binary classifier, the multiclass classification problem is broken down into smaller binary classification problems. Thus, 5 SVMs are employed for classification.

6 Results and Discussion

The Test set results are reported in Table 2. Finetuning the CLIP model (Model C) gave results worse than the baseline findings provided by the task authors. So, Model C is not used when building the ensemble. Model A, Model B and SVM results exceed the baseline results by a good margin. This also illustrates that finetuning the models on a downstream task helped boost the accuracy, unlike the case of SVM where it was trained to classify using features extracted by a pre-trained network.

Model	Task A	Task B
Baseline	0.6500	0.6210
Model A	0.6893	0.6774
Model B	0.7005	0.6823
Model C	0.6537	0.5937
SVM	0.6760	0.6447
Ensemble	**0.7182**	**0.6949**

Table 2: Test Set Results

7 Conclusion

Thus we illustrate the system developed by us for SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. We compare multimodal finetuning vs classification of pre-trained network feature extraction. We have also discussed various methods adopted to train such models and also the data preprocessing done. We show the potential of employing such a model in real-world use cases.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Elisabetta Fersini, Francesca Gasparini, and Silvia Corchs. 2019. Detecting sexist meme on the web: A study on textual and visual cues. In *2019 8th International Conference on Affective Computing and In-*

telligent Interaction Workshops and Demos (ACIIW), pages 226–231.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. Semeval-2022 task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ella Guest, Bertie Vidgen, Alexandros Mittos, Nishanth Sastry, Gareth Tyson, and Helen Margetts. 2021. An expert annotated dataset for the detection of online misogyny. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1336–1350, Online. Association for Computational Linguistics.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing Management*, 57(6):102360.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Michael Salter, Molly Dragiewicz, Jean Burgess, Ariadna Matamoros-Fernandez, Nic Suzor, Delanie Woodlock, and Bridget Harris. 2018. Technology facilitated coercive control: Domestic violence and the competing roles of digital media platforms. *Feminist Media Studies*, 18.

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

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (MultiOFF) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 32–41, Marseille, France. European Language Resources Association (ELRA).

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision.

Mingxing Tan and Quoc V. Le. 2020. Efficientnet: Rethinking model scaling for convolutional neural networks.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers distillation through attention.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks.

Haris Bin Zia, Ignacio Castro, and Gareth Tyson. 2021. Racist or sexist meme? classifying memes beyond hateful. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 215–219, Online. Association for Computational Linguistics.

PAIC at SemEval-2022 Task 5: Multi-Modal Misogynous Detection in MEMES with Multi-Task Learning And Multi-model Fusion

Meizhi Jin and **Mengyuan Zhou** and **Mengfei Yuan** and **Dou Hu** and **Xiyang Du** and **Lianxin Jiang** and **Yang Mo** and **XiaoFeng Shi**
Ping An Life Insurance Company of China, Ltd.
{JINMEIZHI005, ZHOUMENGYUAN425, YUANMENGFEI854, HUDOU470, DUXIYANG037, JIANGLIANXIN769, MOYANG853, SHIXIAOFENG309}
@pingan.com.cn

Abstract

This paper describes our system used in the SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification (MAMI) . Multimedia automatic misogyny recognition consists of the identification of misogynous memes, taking advantage of both text and images as sources of information. The task will be organized around two main subtasks: Task A is a binary classification task, which should be identified either as misogynous or not misogynous. Task B is a multi-label classification task, in which the types of misogyny should be identified in potential overlapping categories, such as stereotype, shaming, objectification, and violence. In this paper, we proposed a system based on multi-task learning for multi-modal misogynous detection in memes. Our system combined image features with text features to train a multi-label classification. The prediction results were obtained by the simple weighted average method of the results with different fusion models, and the results of Task A were corrected by Task B. Our system achieves a test accuracy of 0.755 on Task A (ranking 3rd on the final leaderboard) and the accuracy of 0.731 on Task B (ranking 1st on the final leaderboard).

1 Introduction

In the era of mass online communication, more and more people like to share their thoughts on social media platforms. Social media platforms provide users with the ability to express themselves freely.However, it has also led to a rise in cyber hate, such as bullying, sarcasm, and misogyny, on the internet. A study has shown that women have a strong presence online, particularly in image-based social media such as Twitter and Instagram: 78% of women use social media multiple times per day compared to 65% of men. The Web has opened up a whole new world of opportunities for women, but systematic inequality and discrimination in the real world are also found in online spaces in the form of offensive content against them. Now that we live in a world where everything is connected through social media, hate speech against women used to be limited to a specific place or time. There's no need to wait for a specific time or place. All you have to do is type a few keys on the keyboard. Because people on big social media sites make millions of posts every day, it is not possible to manually check all misogynistic posts. To help human curators, it is important to make algorithms that can tell when users post inappropriate content.

Memes are one of the most popular ways to communicate on social media platforms. A meme is essentially an image characterized by a pictorial content with an overlaying text a posteriori introduced by human, with the main goal of being funny and/or ironic. Most of them are made to be funny, but in a short time, people started to use them as a form of hate against women, which led to sexist and aggressive messages in online environments that made the sexual stereotypes and gender inequality in the real world even worse. Misogyny is a type of offensive speech directed at women. It is common on all social media platforms and is becoming more and more of a problem. Previous work on automatic misogyny detection mostly focused on text mode, and they came up with a bunch of supervised methods, like traditional machine learning methods with lexical features and deep learning methods. It can't be certain, though, if memes are misogynistic if only text mode is used to detect them. Memes are intuitively important for automatic misogyny identification tasks.

In this paper, we describe a system that we submitted for detecting misogynous memes. More specifically, we introduce an ensemble model that uses a multi-task learning mechanism based on multi-modal inputs (image information and text information). In our system, image feature representation was extracted by the pre-trained model of ConvNeXt, and text feature representation was extracted by different pre-trained models (deberta-v3-large and roberta-large model). The two features then undergo representation fusion, where they are transformed into reconstructed representation vectors and pumped into a classification layer to yield the final result. Finally, a modality fusion layer performs a weighted average on the fusion results, which results in different text and image features, and the threshold of the final results is adjusted.

2 Background

Many researches worked on identification misogynous in texts. Earlier methods extract carefully engineered discrete features from texts, including n-grams, keyword's sentiment, punctuations, emoticons, etc(Bouazizi and Ohtsuki, 2015), (Ptácek et al., 2014). Recently, researchers have used the powerful technology of deep learning to obtain a more accurate semantic representation of twitter text, including CNN (Convolutional Neural Network), LSTM (Long Short Term Memory) and pre-trained model, etc. Shushkevich and Cardiff (2018) proposed a technique based on combining several simpler classifiers into a more complex blended model. The model considers the probability of classes calculated by simpler models (Logistic Regression, Naive Bayes, and Support Vector Machines - SVM) . Another method from Liu et al. (2018) was proposed, which three classifiers trained by using SVM with a linear kernel, Random Forests (RF) and Gradient Boosted Trees (GBT). In the testing stage, the same way of text pre-processing and feature extraction is applied to test instances separately, and each pair of two out of the three trained classifiers (SVM+RF, SVM+GBT and RF+GBT) are fused by combining the probabilities for each class by averaging.

The primary driver was the renaissance of neural networks, particularly convolutional neural networks (ConvNets). The introduction of AlexNet (Krizhevsky et al., 2017) precipitated the "ImageNet moment", ushering in a new era of computer vision. Since then, this field has developed rapidly. Representative ConvNets like VGGNet (Simonyan and Zisserman, 2015), ResNe(X)t (He et al., 2016), DenseNet (Huang et al., 2017), MobileNet (Howard et al., 2017) and RegNet focused on different aspects of accuracy, efficiency and scalability, and popularized many useful design principles. However, the introduction of Visual Transformer (VIT) completely changed the landscape of network architecture design, which soon replaced ConvNets and became state-of-the-art image classification model. Although the effect of swin transformer is much better than ResNet on the premise of the same size. However, due to the shift operation, it is difficult to design different networks for different sizes of inputs and restart training. And like Detection Transformer (DETR), the convergence is too slow when training. This year, researchers from Facebook AI Research (FAIR) and UC Berkeley reexamined the design space and tested the limits that pure convnet can reach, which shows that the performance of convolutional neural network is no less than that of visual transformer. This series of pure convnet models is named ConvNeXt. It is constructed entirely from standard convnet modules and competes with transformers in terms of accuracy and scalability.

However, little has been revealed by far on how to effectively combine textual and image information to automatic misogyny identification of memes. Schifanella et al. (2016) simply concatenate manually designed features or deep learning based features of texts and images to make prediction with two modalities. Cai et al. (2019) proposed a hierarchical fusion model to deeply fuse three modalities. Different from there works, We proposed a multitask learning and multi-model.

3 System overview

Figure 1 shows the architecture of our proposed. In this work, we treat text and image classification tasks as two models. Image features are mined using a fine-tuned image classification model based on ConvNeXt. Text features are extracted using a fine-tuned sentence embedding based on the pre-trained RoBERTa model. Image features and text features are fused by direct splicing. A classification layer is connected to the fused model to obtain the category probability of memes. Additionally, we create another fused model that adopts the pre-trained model DeBERTa instead of RoBERTa for text feature exaction progress. Our final submitted labels are voted on by the results inferred by these two proposed models (ConvNeXt + RoBERTa and ConvNeXt + DeBERTa). The threshold value selected for the weighted voting method is 0.405. The classification label returns 1 when the classification probability is greater than 0.405, otherwise it returns 0.

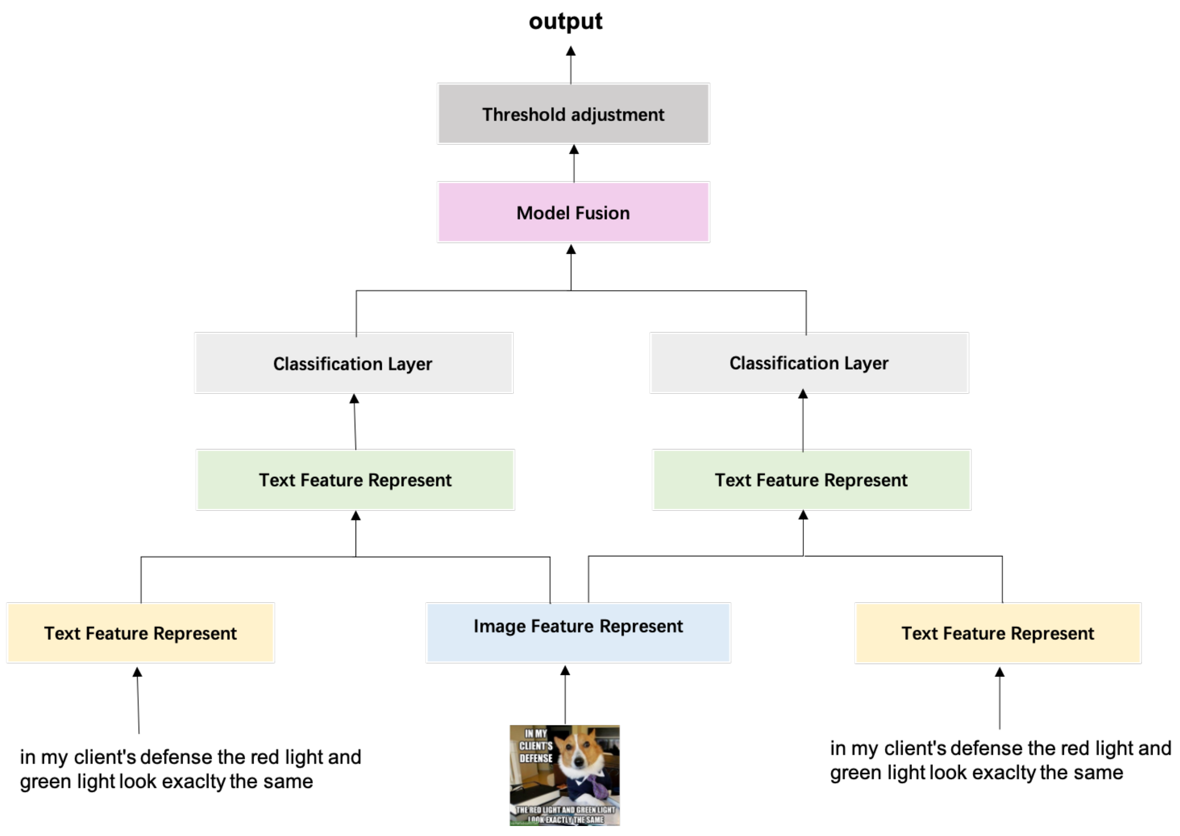


Figure 1: Overview of our proposed model

3.1 Image Feature Representation

In this work, we extracted image feature representation based on an image classification model of ConvNeXt, which trained a baseline model with the Vision Transformer training procedure. Additionally, ConvNeXt (Liu et al., 2022) applied a series of design decisions which are summarized as 1) macro design, 2) ResNeXt, 3) inverted bottleneck, 4) large kernel size, and 5) various layer-wise micro designs. In macro design, ConvNeXt adjusts the number of blocks in each stage and adopts depth-wise convolution. And in micro design, ConvNeXt used fewer activation functions, replacing ReLU with GELU and fewer normalization layers, substituting Batch Normalization (BN) with Layer Normalization (LN), etc. It is worth mentioning that ConvNeXt used data augmentation techniques such as Adamw optimizer, Mixup, Cutmix, RandAugment, Random Erasing, and regularization schemes including Stochastic Depth and Label Smoothing. This was of great help in improving the performance of the model.

3.2 Text Feature Representation

Different pre-trained modals (RoBERTa and DeBERTa) was used as the text feature representation extractor.

RoBERTa - RoBERTa(Liu et al., 2019) is essentially a BERT model with optimal parameters. Compared with BERT model, it used more training data and larger batch size to training longer time. In addition, RoBERTa was trained with dynamic masking instead of static masking, and without NSP loss.

DeBERTa - DeBERTa(He et al., 2021)is a new network architecture proposed by Microsoft. It makes the BERT and RoBERTa models better by using two new techniques. First, the disentangled attention mechanism is applied to represent each word by two vectors, which show their content and relative positions. The attention weights between the word content and position are calculated by disentangled matrices, respectively. Another thing that helps with model pre-training is an "enhanced mask decoder", which adopts the absolute position to predict the masked tokens. In addition, a new virtual adversarial training method is used for fine-tuning to improve models' generalization.

Our system used a series of training techniques. For example, since the general language information and context information obtained by the lower self-attention layer are limited, with the continuous superposition of attention layers, each layer can obtain more language information and context information when encoding. When approaching the last layer, the pre-trained model starts to adjust its embedding information to adapt to different tasks based on different fine-tuning tasks. Therefore, our system set different learning rates in the network to improve the performance of the model. Especially, we set lower learning rates for bottom layers and higher learning rates for top layers. Meanwhile, our system decayed the learning rate with a cosine annealing (Loshchilov and Hutter, 2017) for each batch to improve its overall performance when training deep neural networks.

3.3 Modality Fusion

Multi-modal feature fusion is a useful technique for improving performance in a variety of tasks. According to the order of fusion and prediction, it is divided into early fusion and late fusion. Early fusion is to fuse features before the training model, such as concatenate, add, TFN, MFN, LFN, etc. Late fusion is the fusion of results of different modal predictions, such as maximum fusion, average fusion, etc.

In this scheme, we tried to use different fusion methods for multi-modal fusion, including concatenate, TFN, etc., but finally found that the effect of directly concatenating image features and text features is the best method. Therefore, we concatenate the image features and text features into the fully-connected layer, and then get the fusion features through the normalization layer and multi-sample dropout layer.

3.4 Classification layer

In this work, our system used a two-layer fully-connected neural network as our classification layer. The activation functions for the hidden layer and the output layer are the ReLu and Sigmoid functions, respectively. The loss function is a BCE loss. In addition, we also tried to use SVM as the classification layer, but the performance was not effective.

3.5 Model Fusion

Our system used different pre-trained models to extract text features (RoBERTa and DeBERTa). After fusing image features, we trained a multi-label classification model to obtain the probability value of each category (misogyny, stereotype, shading, objective, and violence). We used weighted voting to fuse the results with two fusion models (ConvNeXt + RoBERTa, and ConvNeXt + DeBERTa). The threshold value selected for the weighted voting method is 0.405 (the classification label returns 1 when the classification probability is greater than 0.405, otherwise it returns 0). Through analysis, it is found that when "shaming", "stereotype", "objectification", and "violence" are positive samples, the "misogynous" must also be positive. In order to improve the results of Task A, the prediction results of "misogynous" are corrected by the prediction results of "shaming", "stereotype", "objectification" and "violence". Experiments show that it is a significant improvement over the result of Task A. Moreover, we also tried to use the results of "shaming", "stereotype", "objectification" and "violence" to correct the results of "misogynous", but found the effect was not satisfying.

4 Experimental setup

4.1 Dataset

The datasets for the MAMI competition(Fersini et al., 2022) are memes collected from the web and manually annotated via crowd sourcing platforms. As summarized in Table 1, the organizers provided 10,000 memes (in jpg format) and a csv file for the training set and 1,000 memes (in jpg format) and a csv file for the testing set. For both Subtask A and B, the memes are released as jpg images.

misogynous: a binary value (1/0) indicating if the meme is misogynous or not. A meme is misogynous if it conceptually describes an offensive, sexist or hateful scene (weak or strong, implicitly or explicitly) having as target a woman or a group of women.

shaming: a binary value (1/0) indicating if the meme is denoting shaming. A shaming meme aims at insulting and offending women because of some characteristics of the body.

stereotype: a binary value (1/0) indicating if the meme is denoting stereotype. A stereotyping meme aims at representing a fixed idea or set of (physically or ideologically) characteristics of women.

objectification: a binary value (1/0) indicating if the meme is denoting objectification. A meme that describes objectification represents a woman like an object through over-appreciation of physical

Task A	Training	Testing	Task B	Training	Testing
Misogynous	5000	500	stereotype	1271	146
			shaming	2810	350
			objectification	2201	348
			violence	953	153
No Misogynous	5000	500	stereotype	3	0
			shaming	0	0
			objectification	1	0
			violence	0	0
Total	10000	1000	Total	7239	997

Table 1: Dataset label distribution

appeal (sexual objectification) or depicting woman as an object (human being without any value as a person).

violence: a binary value (1/0) indicating if the meme is denoting violence. A violent meme describes physical or verbal violence represented by textual or visual content.

The label distribution related to the training and testing datasets is reported in Table1. While the distribution of labels related to the field of misogynous is balanced (for both testing and training datasets), the classes related to the other fields are quite unbalanced. Furthermore, when memes are labeled as "No Misogynous," the classes of "shaming," "stereotype," "objectification," and "violence" are found to be positive in testing datasets. However, there is some error data in training datasets where memes are identified as "No Misogynous".

4.2 Training Details

Image datasets. The public pre-trained model of ConvNeXt_base_22k_1k_384 is adopted in the proposed model. Preprocess the image data through random horizontal and vertical clipping, Random Affine, ColorJitter, Normalize, etc. It is worth mentioning that we set the image size of the training datasets to 384 x 384 and the image size of the verification and testing datasets to (384 + 32) x (384 + 32), which can improve the generalization of the model.

Text datasets. The pre-trained model of deberta-v3-large and roberta-large are adopted for analyzing the text. Models download from HuggingFace[1] are directly used in our model.

Learning rate initialization. Our system sets different learning rates for each layer of text pre-trained model. In particular, the layers of the pre-trained model are divided into three groups with distinct hyper parameters. The learning rates for layer-0 to layer-7 were set to 1e-5/2.6, while layer-8 to layer-15 were set to 1e-5, and layer-16 to layer-23 were set to 1e-5 * 2.6. Setting different learning rates for different layers can make the training more effective and improve the performance of the model.

Learning rate decay. Our system used a method of cosine annealing to decay the learning rates for each batch to avoid falling into a local optimal solution.

multi sample dropout. The four samples dropout were used in our system after fusing the image features and text features. Experiments show that the performance and effects have been significantly improved.

Optimization. Our system applied the AdamW optimizer to optimize the loss function.

Loss Function. Our system applied the BCE loss for multi-label binary classification.

Evaluation Function. Due to our mistakes, we used the incorrect evaluation criteria, which used macro-average F1-Measure instead of weighted-average F1-Measure to train the multi-label classification model. Through offline experiments, we discovered that using the correct evaluation criteria training can result in better and more stable results.

5 Results

In this section, the results of the experiments for our runs will be discussed and compared to the results published in Table 2. We try to only use image or text features for training through different pre-trained models, including Swin Transformers, ConvNeXt, RoBERTa, and DeBERTa. In addition, we also used some other training techniques, such as training a multi-label classification model with

[1]https://huggingface.co/models

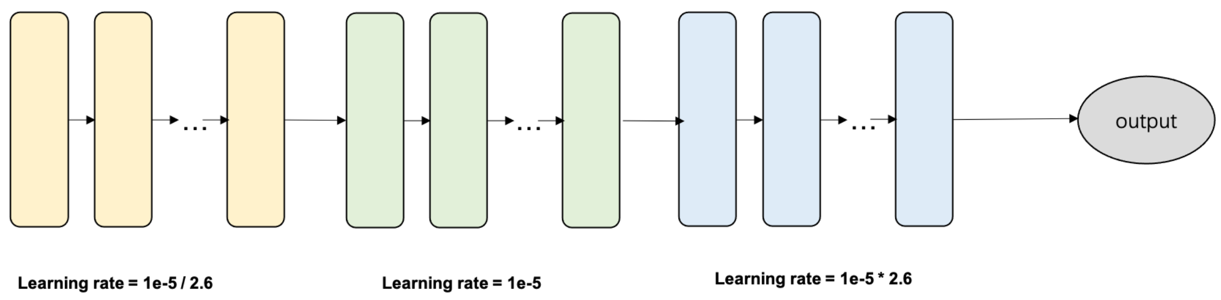


Figure 2: Learning rate initialization

five or four classes, using different classification layers (SVM or full-connected layer), using different fusion methods (concat or LFN), and adjusting the threshold value of fusion prediction results, etc. The specific result of the analysis is shown in Table 2.

V1: Training a image classification model based on a fine-tuned image classification model grounded on swin_large_patch4_window12_384_22k.

V2: Training an image classification model based on a fine-tuned image classification model grounded on ConvNeXt_base_22k_1k_384.

V3: Training a text classification model based on a pre-trained model of roberta-large.

V4: Training a text classification model based on a pre-trained model of deberta-v3-large.

V5: Training a binary classification model for Task A and a multi-label classification model for Task B using four classes (stereotype, humiliation, objectification and violence) after fusing the image features by ConvNeXt_base_22k_1k_384 and the text features by roberta-large, respectively.

V6: Training a multi-label classification model that uses four classes (stereotype, humiliation, objectification and violence) after fusing the image features by ConvNeXt_base_22k_1k_384 and text features by roberta-large (the misogynous results are obtained by the multi-label predictions).

V7: The only difference between this scheme and V6 is that LFN uses feature fusion rather than simply concatenating.

V8: This scheme is the same as the V6. The only difference is that the training labels contain misogynous content (five classes of misogynous, stereotype, humiliation, objectification, and violence).

V9: This scheme is the same as V6. The only difference is that the text features extractor was replaced by deberta-v3-large.

V10: In this scheme, the results of V8 and V9 are fused by a weighted average approach with a threshold set to 0.5.

V11: This scheme is the same as in v10. The only difference is that the threshold is adjusted to 0.4.

V12: This scheme is the same as in v10. The only difference is that the threshold is adjusted to 0.405 (Final results of the leaderboard).

V13: This scheme is the same as in v10. The only difference is that the threshold is adjusted to 0.403 (results not submitted).

Through the experiment, we found the following conclusions:

- The effect of ConvNeXt_base_22k_1k_384 is better than swin_large_patch4_window12_384_22k in our system.
- The effect of multi-modal (text and image) is better than single-modal (text or image) in our system.
- In our system, the effect of multi-task learning is better than individual training.
- According to the results of stereotype, humiliation, objectification and violence to correct misogynous results have a significant improvement effect in our system.
- Adjusting the prediction results also has an improvement for Task B in our system.
- The simple concatenate method is better than the complex feature fusion method in our system.

Version A	Model	Task A	Task B
V1	$swin\_large\_patch4\_window12\_384\_22k$	0.6135	0.5965
V2	$ConvNeXt\_base\_22k\_1k\_384$	0.6431	0.6229
V3	roberta-large	0.656	0.6389
V4	deberta-v3-large	0.6947	0.649
V5	ConvNeXt + RoBERTa + TaskA + TaskB + 4 categories	0.7053	0.7136
V6	ConvNeXt + RoBERTa + TaskB/A + 4 categories	0.7514	0.7136
V7	ConvNeXt + RoBERTa + TaskB/A + 4 categories + LFN	0.7464	0.704
V8	ConvNeXt + RoBERTa + TaskB/A + 5 categories	0.7643	0.7129
V9	ConvNeXt + DeBERTa + TaskB/A + 5 categories	0.7727	0.7240
V10	v8+v9+0.5	0.7785	0.719
V11	v8+v9+0.4	0.7554	0.728
V12	v8+v9+0.405	0.7566	0.7307
V13	v8+v9+0.403	0.7569	0.7319

Table 2: Experimental results of task a and Task B

6 Conclusion

In this paper, we proposed a system to make full use of two modes (image and text) for solving the challenging multi-mode misogynous meme detection task. We extracted the image features and text features using different pre-trained models and weighted averaged the results after fusing the features of image and text. The system performs well in identifying misogynistic memes, achieving 77.85% percent accuracy on TaskA and 73.1% percent accuracy on TaskB.

7 Acknowledgments

This research was supported by the PingAn Life Insurance. We thank the organizers of Università degli Studi di Milano Bicocca (Italy), Google Jigsaw (New York) and Universitat Politècnica de València (Spain) for their support.

References

Mondher Bouazizi and Tomoaki Ohtsuki. 2015. Sarcasm detection in twitter: "all your products are incredibly amazing!!!" - are they really? In *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, pages 1–6. IEEE.

Yitao Cai, Huiyu Cai, and Xiaojun Wan. 2019. Multi-modal sarcasm detection in twitter with hierarchical fusion model. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2506–2515. Association for Computational Linguistics.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90.

Han Liu, Fatima Chiroma, and Mihaela Cocea. 2018. Identification and classification of misogynous tweets using multi-classifier fusion. In *Proceedings of the Third Workshop on Evaluation of Human Language*

Technologies for Iberian Languages (IberEval 2018) co-located with 34th Conference of the Spanish Society for Natural Language Processing (SEPLN 2018), Sevilla, Spain, September 18th, 2018, volume 2150 of *CEUR Workshop Proceedings*, pages 268–273. CEUR-WS.org.

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

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. *CoRR*, abs/2201.03545.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tomás Ptácek, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 213–223. ACL.

Rossano Schifanella, Paloma de Juan, Joel R. Tetreault, and Liangliang Cao. 2016. Detecting sarcasm in multimodal social platforms. In *Proceedings of the 2016 ACM Conference on Multimedia Conference, MM 2016, Amsterdam, The Netherlands, October 15-19, 2016*, pages 1136–1145. ACM.

Elena Shushkevich and John Cardiff. 2018. Classifying misogynistic tweets using a blended model: The AMI shared task in IBEREVAL 2018. In *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) co-located with 34th Conference of the Spanish Society for Natural Language Processing (SEPLN 2018), Sevilla, Spain, September 18th, 2018*, volume 2150 of *CEUR Workshop Proceedings*, pages 255–259. CEUR-WS.org.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

DD-TIG at SemEval-2022 Task 5: Investigating the Relationships Between Multimodal and Unimodal Information in Misogynous Memes Detection and Classification

Ziming Zhou[2], Han Zhao[1], Jingjing Dong[2], Ning Ding[1], Xiaolong Liu[1], Kangli Zhang[1]
[1]DD-TIG
[2]Peking University
{zhaohan,yaeldingning,xlongliu,zhangkangli}@didiglobal.com
{zhouziming,djj}@stu.pku.edu.cn

Abstract

This paper describes our submission for task 5 Multimedia Automatic Misogyny Identification (MAMI) at SemEval-2022. The task is designed to detect and classify misogynous memes. To utilize both textual and visual information presented in a meme, we investigate several of the most recent visual-language transformer-based multimodal models and choose ERNIE-ViL-Large as our base model. For subtask A, with observations of models' overfitting on unimodal patterns, strategies are proposed to mitigate problems of biased words and template memes. For subtask B, we transform this multi-label problem into a multi-class one and experiment with oversampling and complementary techniques. Our approach places 2nd for subtask A and 5th for subtask B in this competition.

1 Introduction

Online misogynous speech has been a worldwide phenomenon spread widely across social media platforms where women are increasingly subjected to offensive content. It has been shown that women are twice as likely as men to encounter online sexual harassment and gender-based violence (Duggan, 2017).

The problem with misogyny detection is that it requires context and external knowledge to understand online speech, which sometimes can be very short and contain subtle meaning (Kiela et al., 2020). Since memes are getting popular as communication tools on social media platforms, misogynous memes have the potential to affect everyone in our society. Automatic multimodal internet memes identification becomes a new challenging type of misogyny detection task that can only be solved by joint reasoning and understanding of visual and textual information (Zhu, 2020).

The proposed task Multimedia Automatic Misogyny Identification (MAMI) (Fersini et al., 2022) at SemEval-2022 requires participants to identify misogynous memes (subtask A) and classify them as certain overlapping categories: stereotype, shaming, objectification, and violence (subtask B).

This paper describes the system developed by the DD-TIG team for SemEval-2022 Task 5 MAMI. This work contributes to the following: for subtask A, solutions to biased words and template memes are proposed to mitigate the effects of overfitting in unimodal information. We also utilize ensemble learning and external knowledge source like Perspective API to boost the performance of our system. For subtask B, we transform the multi-label classification problem into a multi-class classification problem and reach a better result with oversampling and complementary strategy.

2 Background

2.1 Misogynous memes dataset

MAMI task provides participants with a misogynous memes dataset that contains meme images, the transcriptions of texts on memes, and label annotations. For the training set and test set, misogynous and non-misogynous labels are balanced while misogynous category labels are imbalanced (see Table 1).

Label	Trial	Training	Test
Misogynous	44	5000	500
Non-misogynous	57	5000	500
Shaming	0	1274	146
Stereotype	34	2810	350
Objectification	2	2202	348
Violence	9	953	153

Table 1: Summary of the misogynous memes dataset

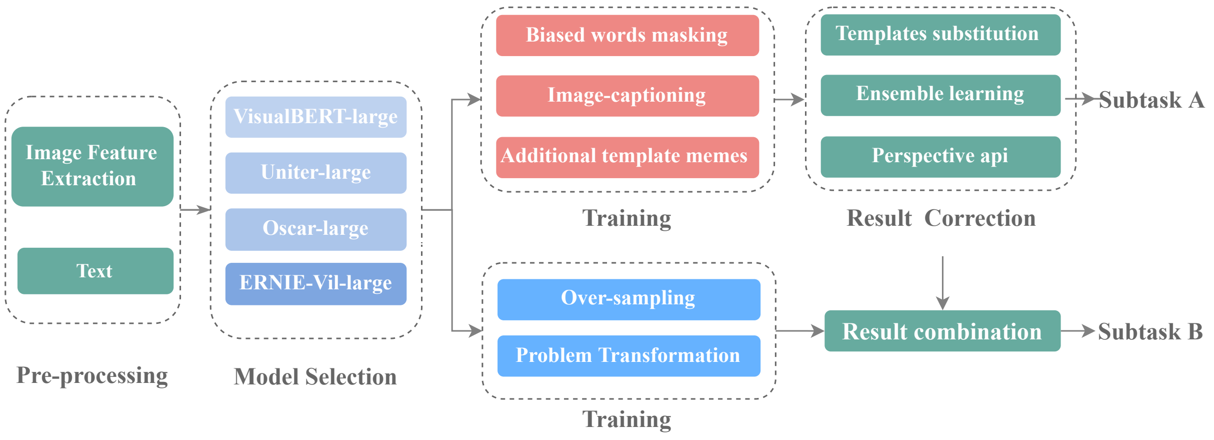


Figure 1: The overall architecture of our proposed system

2.2 Vision and language task

Multimodal misogynous memes identification is a vision and language task. Current state-of-the-art Vision-Language machine learning models are based on the transformer architecture (Vaswani et al., 2017). Among these models, there are two prevalent approaches: single-stream and dual-stream. In single-stream models, such as VisualBERT (Li et al., 2019), UNITER (Chen et al., 2020), OSCAR (Li et al., 2020), image and text features are concatenated and inputted to a standard BERT architecture, which comes under the category of early fusion. In dual-stream models, such as LXMERT (Tan and Bansal, 2019), ERNIE-ViL (Yu et al., 2020a), DeVLBERT (Zhang et al., 2020), VilBERT (Lu et al., 2019), the image and text features are first sent to two independent transformer layers and then into cross-modal transformer layers. Features are combined towards the end of the model as the category of late fusion.

2.3 Vilio: Hateful memes detection framework

The Hateful Memes Challenge (Kiela et al., 2020) is proposed by Facebook AI to leverage machine learning models to solve hateful memes detection problem. Vilio[1] (Muennighoff, 2020) is a code base of 12 different vision+language models and applied to the Hateful Memes Dataset. In our work, we conducted our baseline research on the code of Vilio.

3 System overview

3.1 Preparation

We use the detectron2[2] framework to extract Image features from memes. Detectron2 is provided by Facebook AI with state-of-the-art detection and segmentation algorithms. Specifically, 50 boxes of 2048 dimensions region-based image features are extracted for every meme by Mask-RCNN model. Together with the meme text, which has been extracted using optical character recognition (OCR) and provided in the dataset, features are then fed into the models.

3.2 Vision and language models

We first choose four different base models of VL transformer architectures, namely: VisualBERT, UNITER, OSCAR, and ERNIE-Vil.

We carry out continual pretraining on our dataset with the idea of domain adaptation to reduce the distribution gap between the pretraining dataset and our misogynous memes dataset. MLM pretraining task is taken on pretraining VisualBERT-large, UNITER-large, and OSCAR-large model. However, this does not produce significant performance improvements on our task during the finetuning stage.

Through comparison of results, we found that ERNIE-Vil-large achieves the best performance. In the following steps, we only use the results of ERNIE-Vil-large models.

[1]https://github.com/Muennighoff/vilio

[2]https://github.com/facebookresearch/detectron2

3.3 Strategy for subtask A

3.3.1 Biased words masking

Former research has shown that misogyny detection models can be affected by an unintended bias (Nozza et al., 2019). Some sensitive words, called identity terms, are associated with unreasonably high misogynous scores since they are frequently used in misogynous texts. For example, we observe that the term *kitchen* is frequently used as a stereotypical word against women in our data. Thus, our models tend to associate some non-misogynous texts containing this word with an unreasonably high misogynous score. This situation is known as **unintended bias**, in which models learn usual associations between words (commonly called **identity terms**) which causes them to classify content as misogynous just because it contains one identity word (Godoy and Tommasel, 2021).

Through error analysis on the results of models in the practice stage, we manually collect a list of biased words, including synonyms of woman, dirty words, and controversial words related to feminism. The obtained list of words has been then extended by including their plural form. Refer to Appendix A for the words list.

We propose a novel strategy of biased words masking to mitigate the effects of unintended bias, which also can be regarded as a means of data augmentation. Specifically, when training models, we first use NLTK (Loper and Bird, 2002) to tokenize texts T_i and lemmatize words $w_j \in T_i$ and get $\hat{T_i} = \{w_1, w_2, w_3, \cdots, w_l\}$. Then, for each word w_j in the input text $\hat{T_i}$, with the biased words set $A = \{w_1, w_2, w_3, \cdots, w_k\}$, if $w_j \in A$, it may be masked with a mask token $[mask]$ by a 20 percent probability. We also take strategy with dynamic masking where we generate the masking pattern every time we feed a sequence to the model.

3.3.2 Image captioning

Visual and textual information is semantically aligned in some multimodal tasks, like image-text matching, image-text retrieval, VQA (Yu et al., 2020b). However, for some misogynous memes, image and text are weakly aligned. Thus, there is a semantic gap between visual and textual information. Therefore, we take the strategy of image captioning proposed by previous studies (Das et al., 2020) to enhance model's understanding of visual components. Memes are sent into an image caption model (Xu et al., 2015), which is based on encoder-decoder architecture. This model uses the ResNet-101 as encoder and LSTM as decoder and takes the attention mechanism and beam search when decoding. As a result, this image caption model generates additional descriptions $T_a = \{w_1, w_2, w_3, \cdots, w_i\}$ for visual contents of each meme in the training set; the original text T_o and generated text T_a are concatenated with a separate token $[sep]$.

3.3.3 Templates memes

Through examination of images in the misogyny memes dataset, we notice that many memes are generated by tools of online memes websites. For example, in IMGflip [3], users can choose a meme template from thousands of meme templates and just input their text to caption this template and then get a new meme. In the following part, we refer to memes generated by templates as **template memes**.

In our training set, more than 20 percent of memes are template memes. Some misogynous memes and non-misogynous memes are generated with the same templates and different texts. This may raise a problem that our model associates a high misogynous score or low misogynous score to certain meme templates that actually serve as the medium and contain no misogynous meaning, especially when there is only a few misogynous and non-misogynous sample based on certain templates in the training set. We propose two solutions to mitigate this problem.

Additional template memes: We collect 1,800 memes from memes website. These memes are examples of meme templates and contain no misogynous meaning. Therefore, these memes can be used as the negative sample in our dataset. Without directly adding these memes into the training set, we use our model to make inferences on these memes and only add those false samples (about 90 memes) into the training set.

Templates substitution: we can use an image retrieval model like pre-trained imageNet (Vedaldi and Lenc, 2015) to match memes and templates and find memes that are generated by templates. There are more than 200 memes in the test set are generated with templates. For those memes, original texts and different background pictures are combined to produce K new memes $I_i =$

[3] `https://imgflip.com`

$\{I_1, I_2, I_3, \cdots, I_k\}$. The probability $\hat{p}$ of a sample will be a combination of model's inferential result on the original meme Ia and new memes I_i with a weighted average w.

$$\hat{p} = \frac{p_a + w \cdot \sum_{j=1}^{k} p_j}{K+1} \tag{1}$$

3.3.4 Ensemble learning

The predicted results of our models can be varying since we take the above-mentioned strategy to train different base models. Thus, we continue to improve the whole system's generalizability and robustness with ensemble learning, where predictions of multiple base models are combined with the method of majority Voting (Velioglu and Rose, 2020). In particular, K (K=20) models are selected for ensemble learning, and predictions are collected from each of the models. The label of data is determined by the majority voted class. We hypothesize that some models show a high recall and low precision and vice versa. So a collection of models may balance out individual weaknesses to achieve better performance than any single model used in the ensemble.

$$\hat{y} = \frac{\sum_{j=1}^{K} y_j}{K} \tag{2}$$

3.3.5 Perspective API

Perspective [4] is a free API that uses machine learning to identify toxic comments, making it easier to host better conversations online. We use Perspective API to get a toxic score for the text of our test data. Labels from the previous models' output and probabilities from Perspective API's results are linearly combined with simple linear regression.

3.4 Strategy for subtask B

3.4.1 Transforming a multi-label problem into multi-class problems

A conventional way to solve a multi-label problem is to transform it into binary classification problems where one binary classifier is independently trained for each label. In machine learning implementation, each unit in the output layer uses the sigmoid activation. This will predict a probability of class membership for the label, a value between 0 and 1. Finally, the model would be fit with the binary cross-entropy loss function. However, there are two problems with this approach. On the one hand, it is troublesome to set an optimal threshold for each label. On the other hand, it does not incorporate information about the relationships between labels. For example, label y_a may only occur by itself; labels y_a and y_b may often occur together; labels y_a and y_c may never occur together.

Since the number of labels in subtask B is 4, which is relatively small, we transform this multi-label problem into multi-class problems. Every possible combination of output labels ($[0,0,0,0], [1,0,0,0], \cdots$) will be taken as a class, and the new space of the label set would be 2^4.

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

3.4.2 Over-sampling Technique

label	Positive	Negative
Shaming	12.74%	87.26%
Stereotype	28.10%	71.90%
Objectification	22.02%	77.98%
Violence	9.53%	90.47%

Table 2: Distribution of misogynous categories labels in training set

In subtask B, as shown in table 2 , the number of positive samples and negative samples in all misogynous categories is widely imbalanced. Hence, up-sampling of data is done using over-sampling on the positive sample. Thus our new loss function is defined as follows:

$$J = -\sum_{i=1}^{N} \log p_i \cdot \alpha \tag{3}$$

$$\alpha = \begin{cases} \alpha_{neg} & y_i = [0,0,0,0] \\ \alpha_{pos} & otherwise \end{cases} \tag{4}$$

where N is the size of training set; α_{pos} and α_{neg} are the weights for the misogynous and non-misogynous respectively such that $\alpha_{neg} > \alpha_{pos}$ and $\alpha_{neg} + \alpha_{pos} = 1$.

3.4.3 Combination with subtask A results

We train a binary-classification model for task A and multi-class classification for task B separately. Then, we will use the result of Model A to modify the result of Model B, which means if sample X_i a is predicted as $y_a = 0$ or non-misogynous in Model A, it would not belong to any misogynous category, and its predicted label y_b of Model B would be discarded.

$$\hat{y} = \begin{cases} [0,0,0,0] & y_a = 0 \\ y_b & y_a = 1 \end{cases} \tag{5}$$

4 Experimental setup

In our baseline approaches, VisualBERT-Large, OSCAR-Large, and UNITER-Large provided by Villo are trained for 5 epochs with a batch size of 16. ERNIE-ViL-Large models provided by the original author are trained for 5000 steps with a batch size of 8. In our work, not much time was spent on hyperparameter optimization, and since we notice that there is not much difference when training models with varying hyperparameter settings and we focus more on other strategies. The hyperparameters for finetuning ERNIE-ViL-Large are presented in Appendix B.

5 Results & Discussion

5.1 Subtask A

Table 3 presents the results of our baseline approaches on the test set, where models are evaluated using Accuracy and a macro-average F1-score, while the latter one is the official metrics for system evaluation in this competition.

Model	**Accuracy**	**F1-score**
Oscar-large	69.6	68.9
Uniter-large	69.2	68.4
VisualBERT-large	69.2	68.0
ERNIE-Vil-large	**71.5**	**70.7**

Table 3: The performance of base models on subtask A

ERNIE-Vil has been the STOA model on the multimodal task leaderboard and also achieves competitive performance on our task without any other modification. It is also worth mentioning that continual pretaining with MLM task is conducted on Oscar, Uniter, and VisualBERT, but no improvement is observed. Therefore, ERNIE-Vil is chosen as our base model for further modification with other strategies. Table 4 shows the results of biased word masking, image captioning, and adding false positive samples of template memes into the training set.

Biased word masking experiments have been conducted several times, and the effectiveness is shown by considerable improvement. It is noted that we do not raise scores by only taking the Image captioning technique, but F1-score has a slight increase when image captioning is combined with biased word masking. We hypothesize that image captioning may add noise to our training data since there is a gap between memes in misogynous datasets and the image captioning model's training data. This intuition is confirmed after examining the caption text generated by the Image captioning model, which fails to detect several objects in several images. After we add false-positive samples of template memes into the training set, the performance of our model is boosted. It shows that our model does associate certain normal memes patterns with misogynous or non-misogynous attributes, which can be regarded as biased images.

The results of ensemble learning, templates substitution, and perspective API are shown in Table 4. Ensemble learning obtains a significant improvement where we use different models produced by several times' training of random biased words masking. Since these models are trained with varying texts, we hypothesize their errors will be different, and therefore ensembling may lead to complementary effects and help improve performance. Templates substitution also shows the effectiveness, and this is explained as we find models tend to associate template memes with a high misogynous score, but a majority of them are negative. Perspective API can correct predicted results when sentences contain other malicious words and phrases, but our model does not meet the word or phrase in the training set.

5.2 Subtask B

Table 5 presents the results of our system on the test set for subtask B, where models are evaluated using a weighted-average F1-score, which is the official metric for system evaluation in this competition.

The performance of our model on subtask B is notably improved after the results are modified with the result in subtask A, which has reached a relatively high accuracy score and can be benifical to reduce the number of false positive samples.

5.3 Error analysis

A confusion matrix for subtask A (see Table 6) and a classification report for subtask B (see Table 7) are presented, which will be combined with some bad cases to have both qualitative and quantitative assessments on our system.

	Misogynous	**Non-misogynous**
Misogynous	328	46
Non-misogynous	172	454

Table 6: Confused matrix for subtask A

For subtask A, obviously, the problem with

Model	Accuracy	F1-score
ERNIE-Vil-large	71.5	70.7
ERNIE-Vil-large + WM	72.8	72.1
ERNIE-Vil-large + IC	71.0	70.6
ERNIE-Vil-large + WM + IC	72.7	72.5
ERNIE-Vil-large + WM + IC + AD	73.8	73.7
ERNIE-Vil-large + WM + IC + AD + Emsembling	76.7	76.5
ERNIE-Vil-large + WM + IC + AD + Emsembling + TS	78.1	78.0
ERNIE-Vil-large + WM + IC + AD + Emsembling + TS + PA	**79.4**	**79.3**

Table 4: The performance of our systems on subtask A (WM is biased words masking. IC is image caption. AD is addtional data of template memes. TS is template substitution. PA is the Perspective API.)

Model	F1-score
ERNIE-Vil-large	70.8
ERNIE-Vil-large + Oversampling	71.3
ERNIE-Vil-large + Oversampling + PT	71.7
ERNIE-Vil-large + Oversampling + PT + RC	**72.8**

Table 5: The performance of our systems on subtask B (PT is problems transformation into multi-class classification. RC is results combination with subtask A)

our system is that a considerable number of non-misogynous samples, about 17.2 percent of total and 34.4 percent of the negative sample, is misclassified as misogynous, as the error type false positive. The goal of strategies like biased words masking is to reduce the effects of certain patterns in texts or images and prevent overfitting. Some patterns still are regarded as crucial features of misogyny by our models, but actually, they are biased. Enlarging the size of our dataset may be beneficial to deal with this problem.

The figure in Appendix C shows an example labeled as non-misogynous in the dataset but predicted as misogynous by our model. As mentioned in the previous part, the word kitchen frequently appears in the misogynous sample since misogynists always hold the stereotype to associate women with certain gender roles. We try to mitigate the bias by biased word masking, but it still can not be solved. Moreover, a girl in this image may be associated with the text, but the image and text are not aligned in fact.

label	Precision	Recall	F1-score
Shaming	0.36	0.55	0.43
Stereotype	0.62	0.64	0.63
Objectification	0.69	0.70	0.69
Violence	0.64	0.55	0.59

Table 7: Classification report for subtask B

For subtask B, our model shows relatively poor performance on the label shaming. According to the definition of shaming provided by MAMI organizers, a shaming meme aims at insulting and offending women because of some characteristics of the body. There are two possible reasons to explain this problem. First, we notice that there are some female characters in memes generated by mocking templates, and in truth, the texts on the memes are not targeted towards the female characters in the memes. Second, the definition of shaming is vague and overlaps with other categories of misogyny.

Thus, there is the challenge of this competition: the information from the image and text modalities should not always be treated equally. Sometimes text information should be emphasized if this meme is based on some templates. In multimodal understanding and reasoning tasks, unimodal information can be imbalanced.

6 Conclusion

In this paper, we have presented our work on Multimedia Automatic Misogyny Identification (MAMI) at SemEval-2022. Mainstream vision-language models are applied on misogynous memes dataset in the baseline approach. For subtask A, to better utilize multimodel information and unimodal information, we propose solutions to mitigate the effects of biased words and templates memes. Ensemble learning and external knowledge source like perspective API are used to enhance the performance of our system. For subtask B, training with oversampling strategy, we use a multi-class model to solve this multi-label problem and gain improvement from our subtask A model. In short, this task could never be solved easily since it relies heavily on the context, external knowledge, relations between modalities.

References

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Abhishek Das, Japsimar Singh Wahi, and Siyao Li. 2020. Detecting hate speech in multi-modal memes. *arXiv preprint arXiv:2012.14891*.

Maeve Duggan. 2017. Online harassment 2017.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Daniela Godoy and Antonela Tommasel. 2021. Is my model biased? exploring unintended bias in misogyny detection tasks.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Niklas Muennighoff. 2020. Vilio: state-of-the-art visio-linguistic models applied to hateful memes. *arXiv preprint arXiv:2012.07788*.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *Ieee/wic/acm international conference on web intelligence*, pages 149–155.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Andrea Vedaldi and Karel Lenc. 2015. Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 689–692.

Riza Velioglu and Jewgeni Rose. 2020. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge. *arXiv preprint arXiv:2012.12975*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020a. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*.

Zhou Yu, Yuhao Cui, Jun Yu, Meng Wang, Dacheng Tao, and Qi Tian. 2020b. Deep multimodal neural architecture search. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3743–3752.

Shengyu Zhang, Tan Jiang, Tan Wang, Kun Kuang, Zhou Zhao, Jianke Zhu, Jin Yu, Hongxia Yang, and Fei Wu. 2020. Devlbert: Learning deconfounded visio-linguistic representations. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 4373–4382.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

A Biased words list

bitch, bitches, clean, cooking, dish, equal, female, females, feminist, feminists, fuck, fucking, gender, genders, hooker, hookers, horny, house, housewife, kitchen, mama, mom, moms, prostitute, prostitutes, sex, sexism, sexual, single, wash

B Hyperparameters setting

	ERNIE-ViL-Large
Training steps	5000
Warm steps	500
Learning rate	1e-5
Learning rate decay	0.1
Batch size	8
Fusion method	sum
Attention dropout	0.1
Dropout rate	0.5
Max seqence length	256
Optimizer	AdamW

Table 8: Hyperparameters setting for finetuning ERNIE-ViL-Large

C An example of bad cases

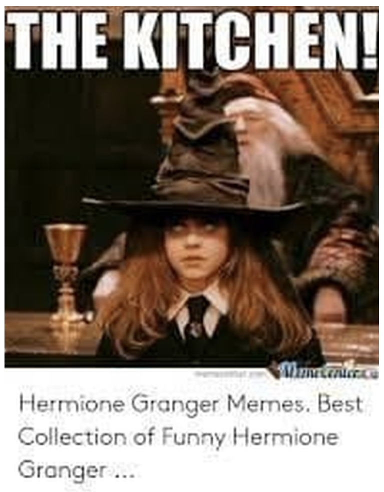

THE KITCHEN!
Hermione Granger Memes. Best Collection of Funny Hermione Granger ...

Figure 2: An example of bad cases

TechSSN at SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification using Deep Learning Models

Rajalakshmi Sivanaiah, Angel Deborah S, Sakaya Milton R, Mirnalinee T T
Department of Computer Science and Engineering
Sri Sivasubramaniya Nadar College of Engineering
Chennai - 603110, Tamil Nadu, India
rajalakshmis@ssn.edu.in, angeldeborahs@ssn.edu.in,
miltonrs@ssn.edu.in, mirnalineett@ssn.edu.in

Abstract

Research is progressing in a fast manner in the field of offensive, hate speech, abusive and sarcastic data. Tackling hate speech against women is urgent and really needed to give respect to the lady of our life. This paper describes the system used for identifying misogynous content using images and text. The system developed by the team TECHSSN uses transformer models to detect the misogynous content from text and Convolutional Neural Network model for image data. Various models like BERT, ALBERT, XLNET and CNN are explored and the combination of ALBERT and CNN as an ensemble model provides better results than the rest. This system was developed for the task 5 of the competition, SemEval 2022.

1 Introduction

In our society, women are facing lot of challenges in terms of education, employment, career and life. Eventhough women are better off now-a-days, there are not considered as equal to men in many situations. With the invent and rapid usage of social media platforms, offensive images and texts conveying several forms of hate against women are transmitted and spread online in a fast manner. Although opportunities for women have been increased on the Internet, systematic inequality and discrimination offline is continued in online in the form of offensive contents through MEMEs. A meme is essentially an image characterized by a pictorial content with an overlaying text introduced by human on it.

Most of the memes are created mainly for funniness, still it is used for ironic purpose too. The task 5 by Fersini et al. (2022) in SemEval 2022 mainly focused on identifying the misogynous memes using textual and image contents. There are two subtasks in this. Subtask A is to categorize the memes as misogynous or not misogynous. Subtask B is to classify the misogynous memes into one of the categories like stereotype, shaming, objectification and violence.

2 Related Work

The survey on various techniques used for Automatic Misogyny Identification (AMI) tasks happened in EVALITA 2018 and IBERALEVAL 2018 were discussed in detail by Shushkevich and Cardiff (2019). Machine learning (SVM, Naive Bayes, Logistic Regression) and deep learning techniques (CNN, GRU, RNN, LSTM) are used and achieved an accuracy of 90% in IBERALEVAL task and 70% in EVALITA task.

García-Díaz et al. (2021) uses sentiment analysis and social computing technologies with word embeddings and linguistic features for AMI that achieves better results than traditional machine learning algorithms. Hate speech against women is handled by Frenda et al. (2019). Pamungkas et al. (2020) created models using RNN and BERT for multilingual misogyny detection.

We have performed irony and offensive language detection in earlier SemEval workshop tasks Sivanaiah et al. (2021), Sivanaiah et al. (2020), Sivanaiah et al. (2019) and Sivanaiah et al. (2018). Various machine learning techniques like linear regression, logistic regression, naive Bayes, Random forest, Support Vector Machines and deep learning techniques like Recurrent Neural Networks, Convolutional Neural Networks, Long Short Term Memory Networks, BERT, ColBERT models are used in the above tasks. BERT and ColBERT models perfomed better than other machine learning and deep learning models.

3 System Overview

The proposed system uses the Transformer models to classify the textual content of the memes and Convolutional Neural Network (CNN) to classify

Dataset	Misogynous		Shaming		Stereotype		Objectification		Violence	
	1	0	1	0	1	0	1	0	1	0
Training (10000)	5000	5000	1271	3729	2810	2190	2201	2799	953	4047
Trial (100)	44	56	0	44	34	10	2	42	9	35
Test (1000)	500	500	146	354	350	150	348	152	153	347

Table 1: Data distribution

the visual content. The majority voting ensemble classifier is used to predict the label for the input sample using the output of textual and visual content classifiers. The data is preprocessed to remove the unwanted information before building the model. The architecture of the system is shown in Figure 1.

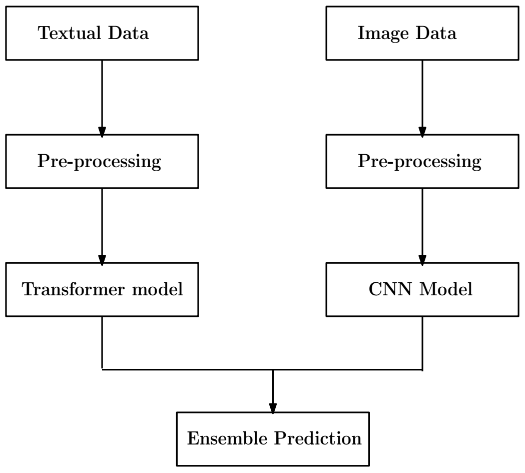


Figure 1: System Architecture

The steps can be summarized as follows:

1. Preprocess the text data and image data separately
2. Separate the image data into folders according to the label category
3. Setup the architecture of the transformer models
4. Train and evaluate the transformer models with text data
5. Use CNN model to train and evaluate the image data
6. Generate the class labels for the test data using transformer model and CNN model
7. Combine the output of both the models using majority voting ensemble classifier

3.1 Dataset

Organizers have provided the data with 10000, 100 and 1000 samples in training, trial and testing datasets. The division of data into various labels are shown in Table 1. Out of 10000 training data, 5000 belongs to misogynous class and 5000 belongs to non-misogynous class. Out of 5000 training misogynous data, 1271 belongs to shaming category and 3729 to not-shaming category. There are 2810 stereotype samples and 2190 non-stereotype samples in 5000 training misogynous data. We have 2201 objectification and 2799 non-objectification samples in misogynous type in training dataset. There are 953 violence and 4047 non-violence samples in misogynous training data.

3.2 Data Pre-processing

Data preprocessing is critical for the success of any machine learning solution to remove the irregularities in the data. Normalization is done to flatten the dimensions of data in textual form. The text data is cleaned and processed to remove URLs, annotate emojis, emoticons, convert uppercase to lowercase, remove stopwords, remove special characters, remove accented characters, lemmatize text, and remove extra whitespace. The images are categorized into their corresponding label folders for further processing. Noise in the image data is removed and converted into pixel values of size 128x128.

3.3 Classification with Transformer models

A transformer is a deep learning model that uses self-attention with weights. Transformer models are used in natural language processing and computer vision. It uses encoder-decoder mechanism to learn features from the sequential input data. Encoder learns the relevant features in the input and pass them to next layer. Decoder does the opposite by taking the output of encoder and incorporate the contextual information to generate the output sequence. Both encoder and decoder layer uses attention mechanism to perform these operations. While Recurrent Neural Networks (RNN) process the data in the sequential order, transformers does not require this. Bidirectional Encoder Representations from Transformers (BERT) is a popular model developed by Google for language modelling (Devlin et al., 2018). Variants of BERT like A Lite BERT (ALBERT) (Lan et al., 2019), XLNET (Yang et al., 2019), Robustly optimized BERT approach (ROBERTa) and COntextualized Late interaction over BERT (ColBERT) are used to perform the learning on text data. XLnet is an extension of the Transformer-XL model pre-trained using an autoregressive method to learn bidirectional contexts.

BERT model works in a better manner for sparse data representations. Instead of training the model from the base, we can take a pretrained model and tune the model to suit our need. The drawback of BERT model is that it generates many parameters for learning which makes the model complex and time consuming. ALBERT model is a lighter version of BERT that reduces the parameters size without much reduction in performance. ALBERT model uses the multi-headed, multi-layer transformer architecture. The number of epochs used to train the models is 5. The embeddings used are albert-base-v2, xlnet-base-uncased and bert-base-uncased.

3.4 Classification using CNN model

Convolutional Neural Network (CNN) (Albawi et al., 2017) is a deep learning technique that takes the image as input, assign importance in the form of learnable weights, learn the various aspects in the input and classify the data. It uses convolutional, maxpooling and dropout layers to learn the features. CNNs have wide applications in computer vision, medical image processing, NLP and recommender systems. We have used 3 convolutional layer with ReLU activation function, maxpooling layer after each convolutional layer, fully connected layer with ReLU activation function and final output layer with softmax activation. Adam optimizer is used with learning rate 0.001 for 30 epochs.

3.5 Ensemble Prediction

Performance of text model and image model are analyzed with the ensemble technique. Majority voting is done with the output of transformer models and CNN to predict the final output class label. Ensemble method will predict the output class label as misogynous if both transformer and CNN models predict the label as misogynous.

4 Results and Discussions

We have used variants of BERT models for analyzing text data and CNN for image data. The results are tabulated in Table 2. The models are executed for various epochs and the results are listed for 5 epochs.

Models	Subtask A	Subtask B
AlBERT	0.5128	-
XLNet	0.3524	-
BERT	0.4584	0.4137
CNN Model	0.4756	-
Ensemble model	0.5223	0.4673

Table 2: F1-measure Results

In subtask A, we have used AlBERT, XLNet and basic BERT techniques to classify the given sample as misogynous or non-misogynous using the text data. Albert model gives better results than other BERT variants. CNN model is used for the classification of image data. The combined ensemble model (AlBERT + CNN) gives better F1 score than using them separately. For the subtask B, we have worked on two models BERT and ensemble model. Ensemble model with multi-label classification model in transformers and CNN is used for predicting the subcategories. The ensemble model (image and text) achieved better score than the BERT model (text alone) and the performance can be increased by tweaking the hyper parameters.

5 Conclusion

User generated content in social media is rapidly increasing day by day that detecting and limiting the diffusion of sarcastic and hate speech content against women is tedious. Automatic identification and removal of these contents is the current topic of research. Many shared tasks are conducted in various conferences for Automatic Misogyny Identification (AMI).

Task 5 in SemEval 2022 focuses on Multimedia Automatic Misogyny Identification (MAMI) with the help of image and text data. Transformer models are used to identify the misogynous content from the text and CNN model is used to detect from the image content. Performance of the system can be increased by tweeking the parameters in the transformer models. The current system trains the text and visual modalities independently on the labels. However, memes portray misogynistic content though a combination of text and image cues, and training these models independently might miss out on the context provided by the other modality, which might be crucial to further improve the performance of this system.

References

Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. 2017. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Simona Frenda, Bilal Ghanem, Manuel Montes-y Gómez, and Paolo Rosso. 2019. Online hate speech against women: Automatic identification of misogyny and sexism on twitter. *Journal of Intelligent & Fuzzy Systems*, 36(5):4743–4752.

José Antonio García-Díaz, Mar Cánovas-García, Ricardo Colomo-Palacios, and Rafael Valencia-García. 2021. Detecting misogyny in spanish tweets. an approach based on linguistics features and word embeddings. *Future Generation Computer Systems*, 114:506–518.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

Elena Shushkevich and John Cardiff. 2019. Automatic misogyny detection in social media: A survey. *Computación y Sistemas*, 23(4):1159–1164.

Rajalakshmi Sivanaiah, Angel Deborah S, S Milton Rajendram, and Mirnalinee T T. 2018. SSN MLRG1 at SemEval-2018 task 3: Irony detection in English tweets using MultiLayer perceptron. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 633–637, New Orleans, Louisiana. Association for Computational Linguistics.

Rajalakshmi Sivanaiah, Angel Deborah S, S Milton Rajendram, Mirnalinee TT, Abrit Pal Singh, Aviansh Gupta, and Ayush Nanda. 2021. TECHSSN at SemEval-2021 task 7: Humor and offense detection and classification using ColBERT embeddings. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1185–1189, Online. Association for Computational Linguistics.

Rajalakshmi Sivanaiah, Angel Suseelan, Logesh B, Harshini S, Geetika B, Dyaneswaran S, S Milton Rajendram, and Mirnalinee T T. 2019. TECHSSN at SemEval-2019 task 6: Identifying and categorizing offensive language in tweets using deep neural networks. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 753–758, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Rajalakshmi Sivanaiah, Angel Suseelan, S Milton Rajendram, and Mirnalinee T.t. 2020. TECHSSN at SemEval-2020 task 12: Offensive language detection using BERT embeddings. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2190–2196, Barcelona (online). International Committee for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

LastResort at SemEval-2022 Task 5: Towards Misogyny Identification using Visual Linguistic Model Ensembles And Task-Specific Pretraining

Samyak Agrawal **Radhika Mamidi**
International Institute of Information Technology Hyderabad
samyak.agrawal@research.iiit.ac.in,
radhika.mamidi@iiit.ac.in

Abstract

In current times, memes have become one of the most popular mediums to share jokes and information with the masses over the internet. Memes can also be used as tools to spread hatred and target women through degrading content disguised as humour. The task, Multimedia Automatic Misogyny Identification (MAMI), is to detect misogyny in these memes. This task is further divided into two sub-tasks: (A) Misogynous meme identification, where a meme should be categorized either as misogynous or not misogynous and (B) Categorizing these misogynous memes into potential overlapping subcategories. In this paper, we propose models leveraging task-specific pretraining with transfer learning on Visual Linguistic models. With our best performing models, we were able to achieve rank 5 th and 10 th on sub-tasks A and B respectively.

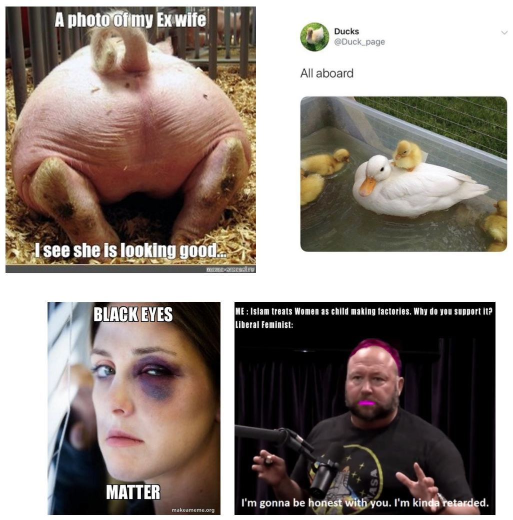

A photo of my Ex wife
I see she is looking good...
Ducks
@Duck_page
All aboard
BLACK EYES
MATTER
makeameme.org
ME : Islam treats Women as child making factories. Why do you support it?
Liberal Feminist:
I'm gonna be honest with you. I'm kinda retarded.

Figure 1: Example memes from the dataset showing the multimodal nature

1 Introduction

The term "misogyny" means hatred towards women. Misogyny can be interpreted through multiple forms such as male privilege, sexual harassment, violence against women, objectification. Memes that targeted women focus on appearance, intellect, their traditional gender roles and capabilities of women (Siddiqi et al., 2018).

For this, SemEval 2022 Task 5 (Fersini et al., 2022) focuses on identifying such behaviour in a multimodal setting (text + image). The textual cues to this task are given in the English language. The task is divided into two sub-tasks. The first subtask is modelled as a binary classification problem. The second sub-task focuses on identifying type of misogyny from a set of overlapping categories, making it a multi-label classification problem.

A meme contains text superimposed on an image. The image's aim in a meme is generally to reinforce a technique in the text, thus making its classification a multimodal problem. Both the modes of information are crucial to establishing the message conveyed by the meme, which can be very different from when the two modalities are evaluated separately.

We experiment with Visual Linguistic (VL) Models like OSCAR (Li et al., 2020) and UNITER (Chen et al., 2020) to understand the memes through both modalities. We employ transfer learning to use a model trained on another similar dataset and then finetune it on our dataset.

As task-specific pretraining has shown to improve results on several NLP tasks (Gururangan et al. (2020)), We experiment with task-specific pretraining our VL models before finetuning it and also finetune it on models task-specifically pretrained for other similar task like hateful memes detection (Kiela et al. (2021)).

We also train BERT (Devlin et al., 2019) based models on only the textual data, thus comparing the performances of multimodal setting vs unimodal settings. This comparison helps us understand how vital each modality is and how much using both together makes a difference.

We discover that even though detecting misogyny in memes can be modelled as a multi-modal task, it can, to a very good extent be done through working with just the textual cues but when it comes to detecting more subtle forms of misogyny, the visual cues play an important role as well. Our system ranked 38th and 19th for sub-taskA and sub-taskB respectively.

The paper is structured as follows: Section 2 describes the dataset along with related work. Section 3 describes our system and model architecture. Section 4 has information regarding the dataset size and splits with libraries used to implement our system. Section 5 has the discussion about the findings from our experiments and section 6 concludes our paper.

2 Background

Nowadays, the internet and various social media platforms have become an intrinsic part of more and more people's lives. With its growth, the problems associated with it have also increased exceedingly, like the increase in hate speech against certain groups including women.

Detecting misogyny and sexist slurs in general over social media can be challenging as its overall meaning can depend on its context and the user it is shown. (Fasoli et al., 2015). For this, look at the few examples in Figure 1 to exhibit the importance of visual and textual cues.

Memes can be defined as an image, video, or text, typically humorous in nature, that is copied and spread rapidly by internet users, often with slight variations. Memes in online culture have been seen to push potential instances of misogyny as a form of "joke" and "irony" while disguising itself as a harmless form of humour. (Drakett et al., 2018).

There has been previous work done to detect hate speech and misogyny. (Pamungkas et al., 2018) Employed Support Vector Machine(SVM) based architectures with a novel lexicon of abusive words to detect misogyny in English and Spanish tweets. (Gasparini et al., 2018) compared unimodal textual classifiers to multimodal classifiers trained with both visual and textual features using early fusion on a dataset of advertisements consisting of image and text marked for being sexist.

The meme classification task is primarily a Visual-linguistic(VL) task where we are trying to classify data where the image can be semantically correlated with the text. Traditional VL approaches are based on primary fusion techniques like early or late fusion, where each modality is learned separately. However, a multimodal pre-trained model might perform better at memes classification (Afridi et al., 2020).

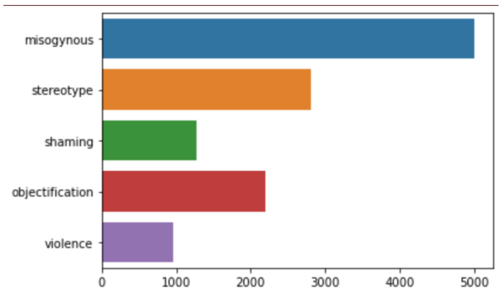


Figure 2: Data Distribution of labels in the training set.

Dataset Description

The dataset (Fersini et al., 2022) contains 10,000 memes.It is furthere divided into train and dev splits. Both tasks require the same dataset, but each task's final labels are different. Half of the 10,000 data points are marked positive and half negative. Of these half marked positive, the data is further annotated for potential overlapping categories of misogyny, namely: stereotype, shaming, objectification and violence.

3 System Overview

We use transformer based models for both the tasks with task specific modifications.

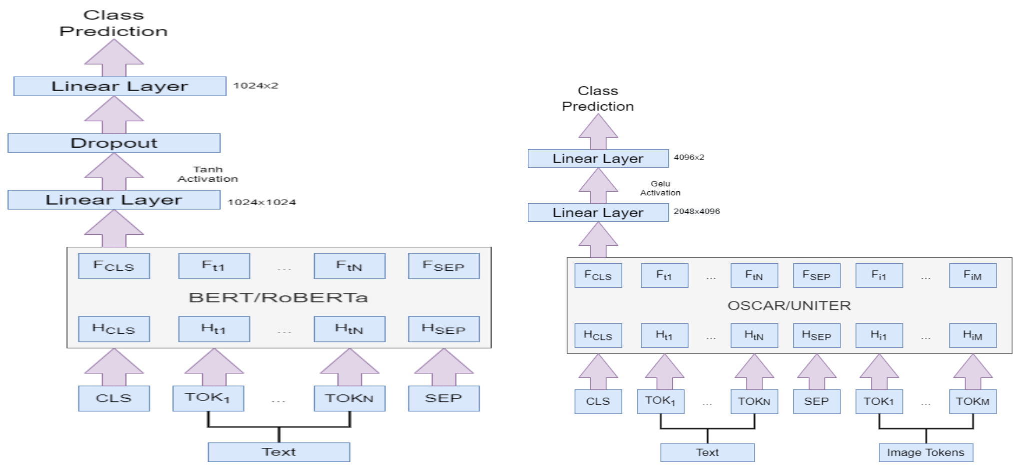
3.1 Pre Processing

Our text is tokenized into subwords to lookup the embedding. For our images, features were extracted using Faster-RCNN (Ren et al., 2016) pre-trained on the VisualGenome dataset(Krishna et al., 2017) trained with and & without object attributes (Anderson et al., 2018).We extract features with object attributes of fixed box sizes 36(OSCAR36) and 50(OSCAR50) and features without object attributes of fixed box size 50 (OSCARV50).

The final input embeddings is a concatenation of both textual and image features represented as

$$h_{[CLS]}, h_{t_1}, \cdots, h_{t_n}, h_{[SEP]}, h_{i_1}, \cdots, h_{i_m}$$

Here $h_{[CLS]}$ and $h_{[SEP]}$ are the vector representations of the special [CLS] and [SEP] tokens respectively. $h_{t_1}, \cdots, h_{t_n}$ represents the text embeddings and $h_{i_1}, \cdots, h_{i_m}$ represents the vision embeddings.



(a) Text Only Models (b) VL Models

Figure 3: Proposed architectures

3.2 Task-specific Pretraining

For both the tasks, We experiment with task-specific pretraining. Every task-specifically pre-trained models were pretrained on two pretraining objectives, namely Masked language Modelling (MLM) and Image Text Matching (ITM). We also make use of models trained on the hateful-memes dataset (Kiela et al., 2021). We use checkpoints from models that were:

1. Task-specifically pretrained on our dataset.
2. Task-specifically pretrained on hateful memes dataset.
3. Task-specifically pretrained and finetuned on hateful memes dataset.

The checkpoints for models pre-trained, fine-tuned on hateful memes dataset were taken from the vilio repository. [1]

3.3 Sub-task A

We used OSCAR as our primary VL model, we also experiemnt with another VL model named UNITER.The UNITER and OSCAR pre-trained weights are based on the BERT transformer. We used Binary Cross Entropy as our loss function to train our models. We trained 3 separate models on the 3 different visual features extracted but use the same textual features. We also experimented with ensembling these models using simple average as our ensembling technique.

We also train transformer-based models like BERT and RoBERTa (Liu et al., 2019) using just the textual cues. We use Binary Cross entropy as our loss function to train our models.

We use the CLS token embeddings from our transformer models and apply classification on top of it. The complete architecture for both text only and VL models can be seen in Figure 3.

3.4 Sub-task B

Here, instead of treating this problem as a multilabel classification problem, we treat it as a binary classification problem just like sub-taskA. We train VL models separately for each of the four labels, namely stereotype, shaming, objectification and violence, rather than training a single model for all labels. We also use an ensemble of models trained on different visual features like we did for sub-taskA.

For our textual models, we trained BERT-based multilabel classification models. We use cross-entropy loss to train our models. Since there is a significant class imbalance, we add weights to our positive data samples while calculating the loss function as done by researchers at (Gupta et al., 2021).The formula is given below:

$$\ell(\mathbf{x},\mathbf{y}) = -\frac{1}{Nd}\sum_{n=1}^{N}\sum_{k=1}^{d}\left[p^k y_n^k \log x_n^k + (1-y_n^k)\log(1-x_n^k)\right]$$
$$p^k = \frac{1}{f^k}(|K| - f^k) \tag{1}$$

Where N is the batch size, n index denotes n^{th} batch element, d is the number of classes, f stands for a vector of class absolute frequencies calculated on the train set, $\mathbf{x}$ is the output vector from the last sigmoid layer, $\mathbf{y}$ is a vector of multi-hot encoded ground truth labels and $|K|$ is the size of the train

[1] vilio-repository,https://github.com/Muennighoff/vilio/blob/master/SCORE_REPRO.md

set.

4 Experimental setup

Parameter	Text Only	VL
Dropout	0.3	-
BatchSize	8	4
Epochs	5	3
Learning Rate	1e-05	1e-05
Warmup	-	0.1
Optimizer	Adam	AdamW

Table 1: Hyperparameters

The dataset contained 10,000 images along with the corresponding texts. Half of the data is marked positive for being misogynous. 85% of the dataset was used to train the model, and the rest was used to validate the model for both subtasks.

We use the VL model implementations of OSCAR and UNITER from the library vilio and for image feature extraction. [2]. We use huggingface [3] library for our transformers trained on just text.

The information about the hyperparameters can be found in Table 1. All models were trained on a GeForce RTX 2080 Ti GPU.

4.1 Evaluation Metrics

We use f1-macro scores as our primary evaluation metrics for both the tasks. We also calculate the accuracy scores for both tasks.

Model	Accuracy	F1-Macro
RoBERTa$_{large}$	68.4	68.3
BERT$_{large}$	64.7	63.7
OSCAR$_{ens}$	68.7	67.2
OSCAR $_{pretrained\_ens}$	69.5	67.8
OSCAR $_{hm\_pretrained\_ens}$	**70**	**68.5**
OSCAR $_{hm\_finteuned\_ens}$	59.9	59.3
UNITER $_{ens}$	65.8	63.3
OSCAR + UNITER $_{ens}$	68.1	66.5
OCSAR36	69.5	67.9
OSCAR50	68.2	66.3
OSCARV50	67.1	64.7

Table 2: Results: Sub-TaskA

[2]vilio,https://github.com/Muennighoff/vilio

[3]Transformers,v4.16.2,https://huggingface.co/docs/transformers/index

5 Results And Discussion

The detailed results from all our experiments conducted can be seen in Table 2 and 3.

We here use the F1 macro scores to judge our models. For subtaskA, We see that OSCAR ensemble models, task-specifically pre-trained on hateful memes dataset perform the best. Another interesting thing to notice is the textual only RoBERTa large model performs almost as good as our best performing VL model and better than all other VL models and is significantly better than BERT large.

We also see that simple average ensemble models for OSCAR perform better than each of its constituent models, and using transfer learning methods with model fine-tuned on hateful memes dataset performed unexpectedly worse. It means that even though hateful memes detection and detecting misogyny in memes are closely related in their idea, they are still not necessarily similar to predict.

In sub-taskB, we see that the ensemble of models with task-specific pretraining on our dataset worked the best and slightly better than the ensemble with task-specific pretraining on the hateful memes dataset. We also see that our OSCAR VL models worked significantly better here than text-only models like BERT and RoBERTa, which is unexpected since the text-only models worked very well compared to VL models in sub-taskA.

Model	Accuracy	F1-Macro
BERT $_{large}$	31.9	45.8
RoBERTa $_{large}$	35.8	45.7
OSCAR $_{ens}$	41.4	**52.6**
OSCAR $_{hm\_pretrained\_ens}$	42.3	52
OSCAR $_{pretrained\_ens}$	**45.5**	31.3
RoBERTa $_{large\_misogynous\_labels}$	12.5	41

Table 3: Results: Sub-TaskB

As we observe that BERT Based models give comparable, and in the case of RoBERTa, better performance than almost all the VL models, it indicates that detecting misogyny might not be an utterly multimodal problem, and just the textual cues are enough in identifying the misogyny.

We also observe that even though text-only models performed very well on misogyny detection, they performed poorly on more fine grained classification tasks, showcasing that the visual cues mattered as well to figure out the subtleties in the classification of the type of misogyny.

We also trained both textual, and VL models on just the data points marked for misogyny as those are the only ones where at least one of the sub-categories of misogyny will be marked positively. However, in this case, the models performed much more poorly. It is because they are not trained on examples that are not misogynous in nature and thus perform poorly on them in the test dataset.

The scores according to the official metrics for our best performing unimodal and multimodal models were as follows: Sub-taskA: RoBERTa $_{\text{large}}$: 68.3; OSCAR $_{\text{hm\_pretrained\_ens}}$: 68.6; Sub-taskB: RoBERTa $_{\text{large}}$: 63.6; OSCAR $_{\text{hm\_pretrained\_ens}}$: 69.1

6 Conclusion

In this paper, our experiments indicate that although misogyny detection in memes is designed as a multi-modal setting, the textual cues also perform very well and, in some instances, better than Visual Linguistic models. We also found out that when it comes to detecting more subtle forms of misogyny, visual cues seem to help in the classification task and perform better than transformer models with just textual cues. More work can be done to improve the results. Future work like experimenting with more upcoming VL models, employing better techniques to address the class imbalance, and using more advanced ensembling techniques like Rank Averaging, Power Averaging & Simplex Optimization can improve results.

References

Tariq Habib Afridi, Aftab Alam, Muhammad Numan Khan, Jawad Khan, and Young-Koo Lee. 2020. A multimodal memes classification: A survey and open research issues.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning.

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

Jessica Drakett, Bridgette Rickett, Katy Day, and Kate Milnes. 2018. Old jokes, new media – online sexism and constructions of gender in internet memes. *Feminism & Psychology*, 28(1):109–127.

Fabio Fasoli, Andrea Carnaghi, and Maria Paola Paladino. 2015. Social acceptability of sexist derogatory and sexist objectifying slurs across contexts. *Language Sciences*, 52:98–107. Slurs.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Francesca Gasparini, Ilaria Erba, Elisabetta Fersini, and Silvia Corchs. 2018. Multimodal classification of sexist advertisements.

Kshitij Gupta, Devansh Gautam, and Radhika Mamidi. 2021. Volta at SemEval-2021 task 6: Towards detecting persuasive texts and images using textual and multimodal ensemble. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1075–1081, Online. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2021. The hateful memes challenge: Detecting hate speech in multimodal memes.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks.

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

Endang Wahyu Pamungkas, Alessandra Teresa Cignarella, Valerio Basile, Viviana Patti, et al. 2018. 14-exlab@ unito for ami at ibereval2018: Exploiting lexical knowledge for detecting misogyny in english and spanish tweets. In *3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018*, volume 2150, pages 234–241. CEUR-WS.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster r-cnn: Towards real-time object detection with region proposal networks.

Nasrina Siddiqi, Anjuman Bains, Arbaaz Mushtaq, and Sheema Aleem. 2018. Analyzing threads of sexism in new age humour: A content analysis of internet memes. *Indian Journal of Social Research*, 59(3):355–367.

HateU at SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification

Aymé Arango, Jesus Perez-Martin, Arniel Labrada
Millennium Institute for Foundational on Data, Chile
University of Chile, Chile
{aarango,jperez,alabrada}@dcc.uchile.cl

Abstract

Hate speech expressions in social media are not limited to textual messages; they can appear in videos, images, or multimodal formats like memes. Existing work towards detecting such expressions has been conducted almost exclusively over textual content, and the analysis of pictures and videos has been very scarce. This paper describes our team proposal in the Multimedia Automatic Misogyny Identification (MAMI) task at SemEval 2022. The challenge consisted of identifying misogynous memes from a dataset where images and text transcriptions were provided. We reported a 71% of F-score using a multimodal system based on the CLIP model.

1 Introduction

Expressions of hate are common in online environments, and they can appear in different types of multimedia content (Bhattacharya et al., 2020). However, the related work on hate-speech and offensive language detection is primarily focused on textual English content (Agrawal and Awekar, 2018; Hosseinmardi et al., 2015). But, even for the English language, the task is still not solved. Evidence of that is recent reports of the increasing amount of hateful content in social media[1] following the occurrence of social or political events. Recent events like the COVID pandemic have brought a new wave of hate (Vishwamitra et al., 2020), with new targets and expressions including hateful memes (Pramanick et al., 2021). Therefore, the techniques for hate speech detection need to evolve towards new types of hate, representations, and languages.

The lack of generality of existing resources along with the emergence of new nets of hate makes current systems quickly outdated[2].

Most of the available datasets contain tweets (Waseem, 2016; Basile et al., 2019), Facebook and Youtube comments (Bosco et al., 2018) and, in general, textual content. Similar to The Hateful Memes Challenge [3] hosted by Facebook in 2020, The Multimedia Automatic Misogyny Identification (MAMI) challenge (Elisabetta Fersini, 2022) is an excellent opportunity for covering the hate speech detection task beyond written expressions.

In this competition, the organizers provided a training set of 10,000 memes labeled as hate speech in two different forms: binary (misogynous, not misogynous) and multi-class (stereotype, shaming, objectification, and violence). The competition comprises two tasks: Task A, for binary identification of misogyny, and Task B, for fine-grained classification of misogynous memes. For final system evaluation, the organizers published a set of 1000 extra unlabeled memes. Each meme in training and testing sets consists of an image with an overlay text. Each object in the dataset consists of an image and a transcription of the overlay text.

This paper describes our team participation in Task A of the MAMI challenge. We encoded images and texts using a pre-trained multi-modal model based on the CLIP model (Radford et al., 2021). We combined the encoded vectors in different ways to obtain a final classification output. Our best result reported was 71% of *f-score*.

In Section 2 we describe the work that has been done on hate speech detection using multi-modal content. In Section 3 we described the training dataset provided in the competition. Then, in Section 4 we describe our system and experiments. Our conclusions can be read in Section 6.

2 Background

Most of the research in hate speech detection has been conducted over textual datasets (Davidson

[1]https://www.channel4.com/news/george-floyd-death-has-led-to-increasing-online-hate-speech-report-claims

[2]https://whatsnewinpublishing.com/the-rise-of-hate-speech-and-what-the-media-can-do-about-it/

[3]https://www.drivendata.org/competitions/64/hateful-memes/page/205/

et al., 2017; Agrawal and Awekar, 2018; Founta et al., 2018). Several strategies based on machine learning models and Natural Language Processing have been used to solve the task, though without success.

On the other hand, the identification of hate other than text formats of multimedia content has been treated only in a few works. Hosseinmardi et al. in 2015 and Singh et al. in 2017 have took advantages of the multi-modal information they could extract from *Instagram*[4] for they work on cyberbullying detection. While Perez-Martin et al. (2020) used the multi-modal representations for retrieving *Twitter* memes from textual queries.

Fortunately, in recent years the multi-modal detection of hateful content has gained popularity due to competitions like "*The Hateful Memes Challenge*" (Velioglu and Rose, 2020) hosted by *Facebook* where different models were proposed to detect hateful content on memes.

The proposed approaches encompass different visual state of the art models like VisualBert (Li et al., 2019), LXMERT (Tan and Bansal, 2019), VilBert (Lu et al., 2019) among others. The winning system combined some of these models with predefined rules (Zhong, 2020) for improving the classification accuracy of difficult samples.

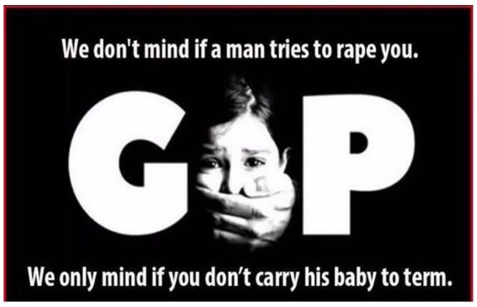
Another recent result on multimodal detection of offensive content has addressed the detection of harmful memes related to the COVID pandemic, also contributing with a new meme dataset (Pramanick et al., 2021).

There is much to do in the multimedia offensive language detection in images and video, considering the popularity of social networks like *Instagram* and *Tik Tok*[5].

3 Dataset Description

The dataset is composed of 10 000 memes, 5000 of which are labeled as *misogynous* and 5000 as *not misogynous*. For each meme is provided the corresponding image in *jpg* format and meme text transcription. All texts are in English; the most extensive text transcription found in the dataset contains 252 words, while the shortest contains one word. A characteristic of this dataset is that in some examples, only the text is enough for determining the nature of the comment (see Figure 1). We do not have evidence of an example where only the image would be necessary for identifying the nature of the meme. This characteristic may be detrimental to the multi-modal intention of the competition.

Figure 1: Meme example 17082. In this example only the texts is necessary for identifying the nature of the meme. The text transcription is: "We don't mind if a man tries to rape you. We only mind you don't carry his baby to term."


We don't mind if a man tries to rape you.
GOP
We only mind if you don't carry his baby to term.

System	F-Score
Text_Only	69.23
Image_Only	65.37
CLIP_concat	70.50
CLIP_sum	**71.20**

Table 1: The results obtained in our experimentation. The details of each system is described in Section 4.

4 Experiments and results

Though we experimented with several models for texts and images, our best result was obtained using the CLIP model as the core of our system.

CLIP model: The CLIP model proposed by Radford et al. 2021 exploits the state-of-the-art textual and visual approaches for learning about images from texts. The general idea of the CLIP training strategy is to jointly learn image and text representations and predict the most similar pairs (image, text). According to the authors, the model can competitive transfer to different vision tasks.

CLIP based systems: We use a pre-trained CLIP model[6] for learning text ($text\_clip$) and image ($text\_clip$) representations from the texts transcriptions and images provided for the competition. We combined these outputs in different ways to obtained a vector x used as input for a classification final classification.

$$output = FFN(x)$$

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

[5]https://www.tiktok.com/

[6]https://github.com/OpenAI/CLIP

The different results can be found in Table 1.

Text_Only: Based on the characteristic of the dataset spotted in Section 3, we investigated if only the text transcriptions were enough for successfully detecting misogyny using this dataset.

$$x = text\_clip$$

With this system, we obtain a 69% of f-score after three epochs. This result is very close to our best result using both types of information (71%). One of the reasons could be the percentage of memes that can be classified by only using the texts, but more need to be studied to obtain a conclusive explanation.

Image_Only: Similar to the *Text_Only* system, we investigated if only the images were enough for successfully detecting misogyny using this dataset. The f-score obtained after five epochs is 65%, a lower result than the *Text_Only* system.

$$x = image\_clip$$

CLIP_concat: This system considered both image and text representations by concatenating them into a single vector in one single vector.

$$x = concat(image\_clip, text\_clip)$$

The results improved by using both representations to a 70% of f-score.

CLIP_sum_system: In this variant of the system, we sum both image and text representation in one single vector. This sum was pondered by a trainable parameter of the model a. The idea of this combination is to give the possibility to the model of using the necessary weights for image and text.

$$x = sum(a * image\_clip, (1 - a) * text\_clip)$$

With this combination we obtained our best reported result for the competition.

5 Error Analysis

We observed the memes miss classified by our best model (*CLIP_sum*). The most common type of error was the *false negative* error, examples wrongly classified as *not misogynist*, we noticed that most of them represent male figures or inanimate objects. Only a small number of memes picture a woman as the central figure (see Figure 2).

On the other hand, the female figures in the *false positive* examples is very common (see Figure 3).

Figure 2: Meme example 15115 from the testing set. Our model *CLIP_sum* wrongly classified it as a *not misogynist* meme. The text transcription is: "YOU DON'T WORK,COOK, CLEAN OR GIVE HEAD? LMAOBRUH LMAOBRUH.com LEGALLY, MY CLIENT IS ENTITLED TO A SIDE B*TCH OR TWO"

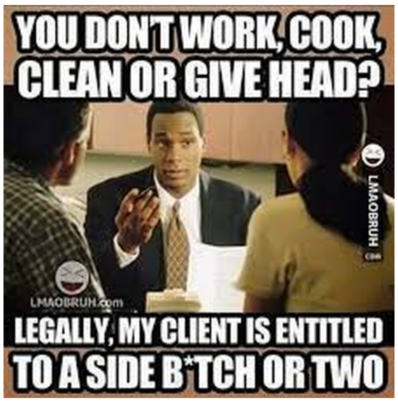
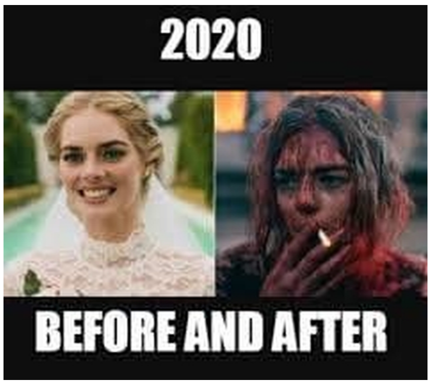

YOU DON'T WORK, COOK, CLEAN OR GIVE HEAD?
LMAOBRUH.com
LMAOBRUH.com
LEGALLY, MY CLIENT IS ENTITLED TO A SIDE B*TCH OR TWO

Figure 3: Meme example 15977 from the testing set. Our model *CLIP_sum* wrongly classified it as a *misogynist* meme. The text transcription is: "2020 BEFORE AND AFTER"



This phenomenon could be caused by a particular bias in the training set that relates misogyny memes with the images of women. But a deeper analysis has to be conducted in this regard.

6 Conclusions

This paper describes our team participation in Task 1 of the Multimedia Automatic Misogyny Identification (MAMI) at SemEval 2022. The purpose of this task was to identify memes as misogynists or not. Images and texts were provided in a training set of 10000 examples. Our team implemented a system based on the pre-trained CLIP approach and reported a 71% of f-score.

The multimodal hate speech detection has been under-addressed through the years and recently is gaining popularity, though there is still much for research in this regard. Moreover, other types of multimedia, like videos, need to be analyzed since they are popular ways of communicating on social networks.

Acknowledgements

This material was supported by the Millennium Institute for Foundational Research on Data (IMFD).

References

Sweta Agrawal and Amit Awekar. 2018. Deep learning for detecting cyberbullying across multiple social media platforms. In *Advances in Information Retrieval - 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings*, pages 141–153.

Valerio Basile, Cristina Bosco, Viviana Patti, Manuela Sanguinetti, Elisabetta Fersini, Debora Nozza, Francisco Rangel, and Paolo Rosso. 2019. Shared task on multilingual detection of hate. SemEval 2019, Task 5.

Shiladitya Bhattacharya, Siddharth Singh, Ritesh Kumar, Akanksha Bansal, Akash Bhagat, Yogesh Dawer, Bornini Lahiri, and Atul Kr Ojha. 2020. Developing a multilingual annotated corpus of misogyny and aggression. *arXiv preprint arXiv:2003.07428*.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, volume 2263, pages 1–9. CEUR.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

Giulia Rizzi Aurora Saibene Berta Chulvi Paolo Rosso Alyssa Lees Jeffrey Sorensen Elisabetta Fersini, Francesca Gasparini. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, Stanford, California, USA, June 25-28, 2018*, pages 491–500. AAAI Press.

Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. 2015. Detection of cyberbullying incidents on the instagram social network. *CoRR*, abs/1503.03909.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Jesus Perez-Martin, Benjamin Bustos, and Magdalena Saldana. 2020. Semantic search of memes on twitter. *arXiv preprint arXiv:2002.01462*.

Shraman Pramanick, Dimitar Dimitrov, Rituparna Mukherjee, Shivam Sharma, Md Akhtar, Preslav Nakov, Tanmoy Chakraborty, et al. 2021. Detecting harmful memes and their targets. *arXiv preprint arXiv:2110.00413*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Vivek K Singh, Souvick Ghosh, and Christin Jose. 2017. Toward multimodal cyberbullying detection. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 2090–2099. ACM.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Riza Velioglu and Jewgeni Rose. 2020. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge. *CoRR*, abs/2012.12975.

Nishant Vishwamitra, Ruijia Roger Hu, Feng Luo, Long Cheng, Matthew Costello, and Yin Yang. 2020. On analyzing covid-19-related hate speech using bert attention. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 669–676. IEEE.

Zeerak Waseem. 2016. Are you a racist or am I seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science, NLP+CSS@EMNLP 2016, Austin, TX, USA, November 5, 2016*, pages 138–142.

Xiayu Zhong. 2020. Classification of multimodal hate speech - the winning solution of hateful memes challenge. *CoRR*, abs/2012.01002.

SRCB at SemEval-2022 Task 5: Pretraining Based Image to Text Late Sequential Fusion System for Multimodal Misogynous Meme Identification

Yujin Wang[*], **Jing Zhang**[✉], **Bohua Peng**[*], **Xudong Zhang**[*]
Xiaoyan Qu, Yimeng Zhuang, Song Liu
Samsung Research China-Beijing (SRC-B)
{yujin1.wang, jing97.zhang, bohua.peng}@samsung.com
{xudong.zhang, xiaoyan11.qu}@samsung.com
{ym.zhuang, s0101.liu}@samsung.com

Abstract

Online misogyny meme detection is an image/text multimodal classification task, the complicated relation of image and text challenges the intelligent system's modality fusion learning capability. In this paper, we investigate the single-stream UNITER and dual-stream CLIP multimodal pretrained models on their capability to handle strong and weakly correlated image/text pairs. The XGBoost classifier with image features extracted by the CLIP model has the highest performance and being robust on domain shift. Based on this, we propose the PBR system, an ensemble system of Pretraining models, Boosting method and Rule-based adjustment, text information is fused into the system using our late sequential fusion scheme. Our system ranks 1st place on both sub-task A and sub-task B of the SemEval-2022 Task 5 Multimedia Automatic Misogyny Identification, with 0.834/0.731 macro F1 scores for sub-task A/B correspondingly.

1 Introduction

Much of the real world's information comes in multimodality, a combination of images, texts, audios and so on. Multimodal understanding aims to utilize different modal of information to improve the overall system recognition intelligence or robustness (Gadzicki et al., 2020), which plays a key foundation role in cognitive AI and embodied AI.

With transfer learning by large deep models and colossal corpus achieving remarkable success in vision and language domain, there is a rising interest in combining both sides' advances to push the multimodality understanding further (Lu et al., 2019; Tan and Bansal, 2019; Chen et al., 2019; Li et al., 2020; Yu et al., 2020; Huo et al., 2021; Kim et al., 2021; Radford et al., 2021). We will limit the discussion scope of multimodal to vision and language in this paper. There are two kinds of representative architecture of multimodal learning models, single-stream models and dual-stream models. Single-stream model fuses the image and text data at an early stage, and then feed into the model. Dual-stream models design separated structure as image encoder and text encoder, and a further module is stacked on top of the unimodel encoders for cross-modal learning objectives (Tan and Bansal, 2019; Yu et al., 2020; Radford et al., 2021; Huo et al., 2021). Usually per-unimodal objectives and multimodal objectives are designed to ensure that the model learns unimodal and crossmodal knowledge, like masked image prediction, masked token prediction, and text-image pairing (Chen et al., 2019; Kim et al., 2021). Two kinds of data distributions are explored for the large-scale pretraining, strongly paired data (Chen et al., 2019; Radford et al., 2021; Li et al., 2020; Kim et al., 2021) and weakly paired data (Huo et al., 2021). The different distributions would directly affect the correlations learned by the model, yet each pretraining corpus only falls in one pattern.

The SemEval-2022 Task 5 (Fersini et al., 2022) Multimedia Automatic Misogyny Identification (MAMI) is a multimodal classification task in English. It targets the identification of misogynous memes (characterized by a pictorial content with an overlaying text a posteriori introduced by human), using the image and text from the meme as input data. It has two sub-tasks: sub-task A: 2-fold classification, to identify whether a meme is misogynous or not; sub-task B: 4-fold fine-grained classification, to further recognize the misogynous meme among potential overlapping categories of stereotype, shaming, objectification and violence.

The relationship of the MAMI paired meme image/text data can vary from highly correlated to weakly correlated or not correlated at all. The semantic logical relationship between the meme's image and text can be: 1) align with each other, containing the same semantic, 2) independent but

[*]Contribution during Intership in Samsung Research China-Beijing.

connected to form a complete semantic, 3) irrelevant with each other, only one modality decides the meme's semantic (refer to the appendix for illustrations). In summary this task demands both understanding the image and text, as well as setting up the correct semantic logic between the image/text modalities.

In this paper, we want to investigate either the multimodal pretrained models of different design architecture are capable of handling such complex relationship between vision and language, and whether the pretrain-and-fine-tuning paradigm is advantageous over the feature-extraction-and-machine-learning-classification paradigm. We choose two strong baseline pretrained models, UNITER (Chen et al., 2019) as the single-stream model, and CLIP (Radford et al., 2021) as the dual-stream model, and fine-tuning with cross-entropy softmax is compared against the widely adopted XGBoost (Chen and Guestrin, 2016) classifier. Domain shift is discovered between the train (dev) and test dataset, and the above two paradigm is explored for both in-domain situation and domain shift situation. An adversarial discrimination loss (Tzeng et al., 2015) is added to the fine-tuning deep neural network for domain shift, while the XGBoost classifier is tuned with its hyper-parameters.

Our results show that 1) for both pretrained models, multimodal fine-tuning performs better than unimodal. The CLIP dual-stream model performs sightly better than the UNITER single-stream model on in-domain data, given the much greater pretraining corpus CLIP has than UNITER. On data with domain shift, the CLIP fine-tuning is much more stable than the UNITER model, but both models suffer from great performance degradation. 2) the performance of feature-extraction-and-machine-learning-classification by XGBoost classifier is no weaker than that of fine-tuning on top of pretrained models, the XGBoost classifier utilizing only image features from CLIP hits best performance on the test dataset among all modality combinations, plus that the XGBoost is cheaper to train. 3) for domain shift, the XGBoost classifier is more robust, the domain adversarial loss for fine-tuning brings a small rise, but still falling behind the XGBoost classifier.

Our final winning system is a combination of machine learning and deep neural network, by Pretraining models, Boosting method and Rule-based adjustment, which we name PBR, based on the XGBoost classifier of CLIP image features, and a late sequential fusion of multimodal/text information into the classifier's prediction, followed by rule-based adjustment. The details will be stated in Section 2. Our system gets 0.834 macro F1 score on sub-task A and 0.731 macro F1 score on sub-task B, ranking 1st place in both the tasks in the leaderboard.

2 System Overview

2.1 Overall Architecture

Our 3-stage ensemble system showed in Figure 1 works as following:

- stage 1, the image feature extracted by the CLIP model is learned by the XGBoost classifier, to form a image only prediction. The image/text paired data is used to fine-tune the UNITER model on the MAMI task. And external text datasets together with the MAMI text data is fed into the BERT model to train a text only model on the MAMI task.

- stage 2, the UNITER fine-tuning predictions and the BERT fine-tuning predictions are used to adjust the medium confidence zone of the XGBoost prediction, by our late sequential fusion scheme.

- stage 3, the sub-task A and sub-task B predictions are mutually adjusted, taking advantage of the two sub-tasks' logical inference relationship with each other.

2.2 Deep Pretrained Model for Image and Text Representation

The multi-head attention of transformer architecture modelling the interaction between any two tokens within a sequence by constant $\mathcal{O}(\infty)$ distance, has proved to be powerful in learning deep bidirectional interactions in language and vision (Lu et al., 2019; Dosovitskiy et al., 2020). We choose two transformer based pretrained model to get the image and text representations.

1) **single-stream model UNITER.** UNITER (Chen et al., 2019) is a large-scale pre-trained model for UNiversal Image-TExt Representation. The image and text input are fused early by concatenation, and fed into the transformer module to learn contextualized representations. The pretraining includes unimodal and multimodal tasks. The

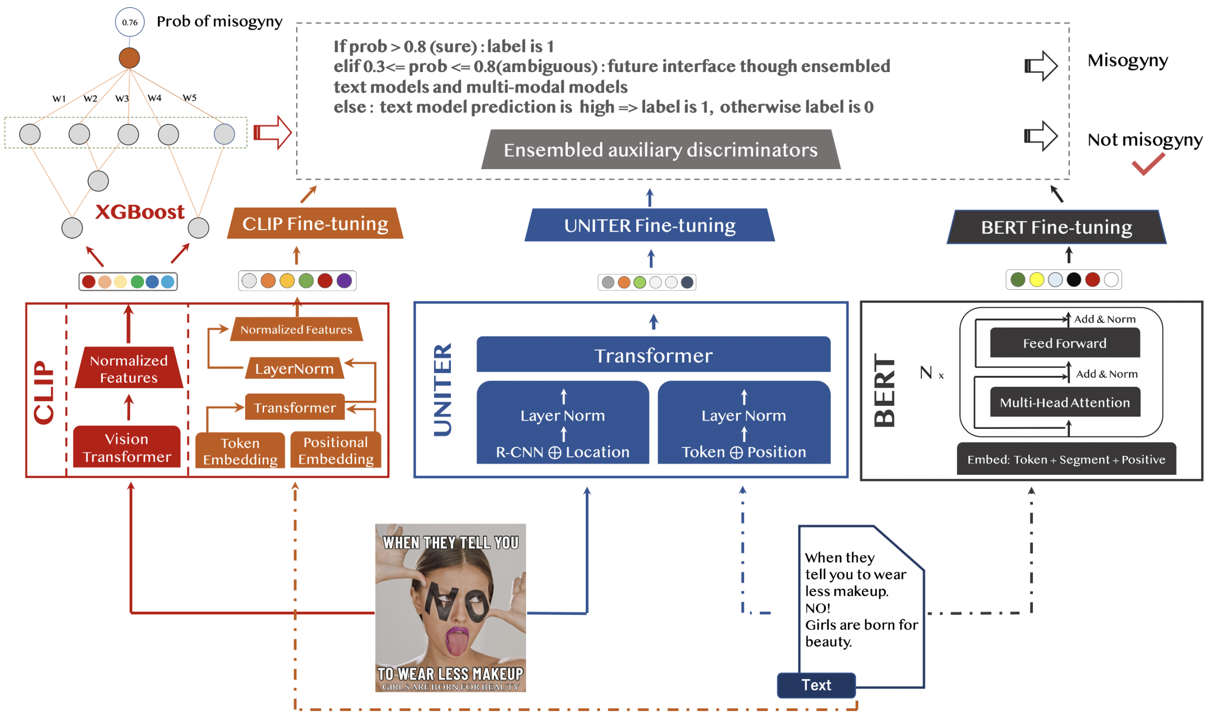


Figure 1: The overall architecture of our ensembled system.

model outputs a fused text and image representation. The pretraining dataset has about 8.4 million image/text pairs.

2) **dual-stream model CLIP.** CLIP (Radford et al., 2021), the Contrastive Language-Image Pretraining model, has separate image transformer encoder and text transformer encoder. The two are joined by a contrastive loss to learn the multi-modal embedding space. The model is pretrained on a dataset of 400 million (image, text) pairs collected from the internet. The simple pretraining task only involves multimodal alignment, predicting which text as a whole is paired with which image, unimodal learning task is not applied. whereas the natural language performs well in enabling zero-shot transfer of the model to downstream tasks when used to reference visual concepts and functioning as prompt text.

2.3 Classification on Downstream Task

2.3.1 Fine-tuning with Pretrained Models

As illustrated in Figure 2, for the UNITER model, the fine-tuning head is a feed forward neural layer (ffn) followed by the cross-entropy softmax classifier. And for the CLIP model, the encoded image and text representations are linearly transformed separately, and then concatenated to be passed forward to a ffn layer and a cross-entropy softmax classifier. The fine-tuning structure of BERT model is the same as the UNITER model. The fine-tuning is used to select the base pretrained model for MAMI, and the fine-tuned UNITER model is utilized as a multimodal voter and the fine-tuned BERT as a text unimodal voter for the model ensemble.

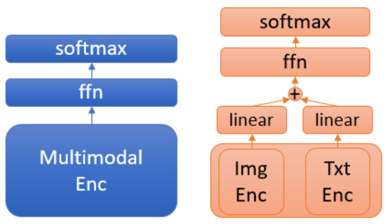


Figure 2: Fine-tuning head structure for UNITER(left) and CLIP(right).

2.3.2 XGBoost Classifier

XGBoost (Chen and Guestrin, 2016) classifier is a tree boosting ensemble model that uses additive functions to predict the output. The boosting ensemble learning algorithm combines multiple weak learners in a sequential method, iteratively improving upon observations. XGBoost borrows from random forests and supports column sampling as well as data sampling. The benefits of the XGBoost classifier is its capability to reduce bias and the low training cost.

In Eq. 1 , the f_k stands for the kth regression

Dataset	**Modality**	**# Samples**	**# Labels**	
		train/dev/test	**# misogyny**	**# non-misogyny**
MAMI	img/txt	3227/837/1000	5000	5000
searched-meme	img/txt	3447/-/-	1564	1883
misogynistic-meme	img/txt	800/-/-	400	400
sexist-detection	txt	1142/-/-	627	515
online-misogyny-eacl2021	txt	6567/-/-	699	5868

Table 1: MAMI task and augmented datasets statistics.

tree with K trees in total, $\hat{y}_i$ is the prediction of sample i formed by the sum of K regression trees. In Eq. 2, L is the training objective, l a differentiable convex loss function that measures the difference between the prediction $\hat{y}_i$ and the target y_i, and Ω is the penalty function to avoid over-fitting. The Eq. 3 describes the iterative update of object function L, y_i^t is the prediction of the i-th instance at the t-th interation. At each iteration t, a new tree f_t is added to optimize the objective, the selection of f_t is by a greedy algorithm that most improves the model according to Eq. 2.

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{1}$$

$$L(\phi) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \tag{2}$$

$$L^t(\phi) = \sum_i l(y_i, \hat{y}_i^{t-1} + f_t(x_i)) + \sum_k \Omega(f_k) \tag{3}$$

2.4 Post-adjustification

2.4.1 Late Sequential Fusion

The XGBoost classifier with image features extracted by CLIP stands out among all modality combinations by a large margin, including the multimodal fusion pattern of both CLIP and UNIER, thus it is chosen as our basis. While the information in text is non-negligible, we make use of it in a late sequential fusion fashion, with image going first and text catching up. We treat the XGBoost prediction score ranging [0,1] as the prediction confidence, denoted as $\hat{p}$, and the whole XGBoost predicted confidence on the MAMI test cases are denoted as $\hat{P}$. We rank $\hat{P}$ in decreasing order, and divide it into three intervals, the high, medium and low confidence intervals. The prediction confidence score from the BERT fine-tuning (text modality) and UNITER fine-tuning (multimodality) are denoted as $\hat{p}_b$ and $\hat{p}_u$.

$$\hat{p} = \begin{cases} \hat{p}, & \text{if } \hat{p} \in [\hat{P}_{hi}, \hat{P}_{t_1}] \\ vote(\hat{p}, \hat{p}_u, \hat{p}_b), & \text{if } \hat{p} \in (\hat{P}_{t_1}, \hat{P}_{t_2}] \\ vote(\hat{p}, \hat{p}_b), & \text{if } \hat{p} \in (\hat{P}_{t_2}, \hat{P}_{lo}] \end{cases} \tag{4}$$

The medium confidence interval reflects the model's uncertainty for classification based on image solely. In the low confidence interval, when the image is highly non-misogynous, while the text is highly misogynous, the whole semantic of the meme would be positive. Thus we fuse the text and image modality in a late sequential way by the scheme in Eq. 4. The high confidence interval take the XGBoost classifier prediction as the result directly, and the medium interval combines the text ($\hat{p}_b$) and multimodal ($\hat{p}_u$) information by voting, while the low interval takes advantage of the text information to adjust the image-only prediction. $\hat{P}_{hi}$ and $\hat{P}_{lo}$ refer to the highest and lowest probability, by experience we choose the endpoints of the medium interval t_1 to be 300, and t_2 be 700 in the descending ranked $\hat{P}$ sequence (eg. $\hat{P}_{t_1}$ equals the probability value of the 300th $\hat{P}$).

2.4.2 Mutual Adjustment of the Sub-task A/B

This step is the final adjustment towards our final results. By task definition, when the MAMI subtask A is non-misogynous, all labels in sub-task B should be 0, vice visa. If any of sub-task B is labelled as 1, sub-task A should be misogynous. Therefore, we design the following rules to increase performance.

1) If the prediction of "misogynous" in sub-task A has high confidence for label 0, while some of the four sub-classes in sub-task B are labelled as 1 , we ignore them and set all the labels to 0s due to the high confidence of the misogyny binary classification.

2) If multiple 1-labels appear in sub-task B, it suggests the sample meme probably being misogyny. Meantime if the confidence level of "misogynous" for sub-task A is in the medium interval and the text and visual combination are also very ambiguous, we set the label to be 1 in sub-task A.

2.5 Data Augmentation

Data augmentation is applied to enrich the data distribution and enhance the system's generalization capability (Perez and Wang, 2017) in the downstream task classification phase (applied both in fine-tuning and XGBoost classification).

Our data augmentation strategy includes 3 aspects: 1) collecting memes of the misogyny topic from search engines with a set of misogynous keywords and neutral keywords. 2) collecting public dataset on misogyny and related topics, to help provide more knowledge on the topic. 3) self-augmentation from the task dataset. For image self-augmentation, we used geometric-based augmentations, including flipping horizontally and vertically with cutout (DeVries and Taylor, 2017), randomly resized cropping and 30-degree rotation, as well as color-based transformation, color jittering. For text self-augmentation, back-translation is used. Details of 1) and 2) can be referred in Section 3.1

3 Experimental Setup

3.1 Training Datasets

The MAMI task dataset and augmented datasets are used for training.

MAMI. Dataset for the SemEval-2022 Task 5 (Fersini et al., 2022), the labels of sub-task A is evenly distributed (1:1 for misogyny and non-misogyny samples), and the labels for sub-task B are distributed unevenly, the shaming/stereotype/objectification/violence have 1271/2810/2201/953 labels correspondingly.

searched-meme. Memes crawled from commercial search engines. Searching by keywords in commercial search engines, and an in-house OCR tool is applied to get the paired text for each meme. The final dataset contains 3447 image-text pairs. The searching keywords is listed in the appendix.

misogynistic-meme. An expert-labeled open misogynistic dataset (Gasparini et al., 2021), it contains 800 memes with manually transcribed text, the misogynisticDE field is used as the label for misogyny.

sexist-detection. A text dataset of sexist statements at workplace (Grosz and Conde-Cespedes, 2020), the label for sexism or not is mapped to misogyny or not for the SemEval-2022 Task 5.

online-misogyny-eacl2021. A text dataset of 6567 labels for Reddit posts and comments for online misogyny detection (Guest et al., 2021).

3.2 Training Details

For the fine-tuning of UNITER and CILP, we mainly follow the original paper, detailed hyper-parameters can be referred in the appendix. The hyper-parameters of the XGBoost classifier is listed in Table 5 in the appendix. We treat the sub-task B as four independent binary classification tasks with four independent XGBoost classifiers. Dealing with the label imbalance, we adjust the XGBoost parameter "scale_pos_weight" to achieve good performance. The sub-task A is evaluated using macro-average F1-Measure, the sub-task B is evaluated using weighted-average F1-Measure (Fersini et al., 2022). A point worth noting is that when there is data imbalance of positive (label-1) and negative (label-0) samples, it is more profitable to predict the less labelled ones in the measure of macro F1. So we try to find label-1 in each category of sub-task B as much as possible by tuning the hyper-parameters since label-1 samples are much less than label-0.

4 Results and Discussion

4.1 Multimodal Pretrained Model Selection

We take the BERT text model as the baseline for text modality and the CLIP image model as the baseline for image modality (the image features of CLIP model outperforms state-of-art image pre-trained models in image classification tasks (Radford et al., 2021)), which we will denote as TB and IB below. We randomly split the 10000 training data into train/dev/inner-test data by 8:1:1, and the test data is released by the task organizer. The UNITER image model and text model are tested by putting the unimodal-only data into the model.

As shown in Table 2, the UNITER image model is well below the IB and the UNITER text model is well below the TB on dev dataset. The UNITER image-text multimodal fine-tuning dev results gains large increase compared to its unitmodal implementations, while slightly better than the TB/IB. This suggests that the unimodal pretraining objectives in UNITER is not as well learned as the unimodal

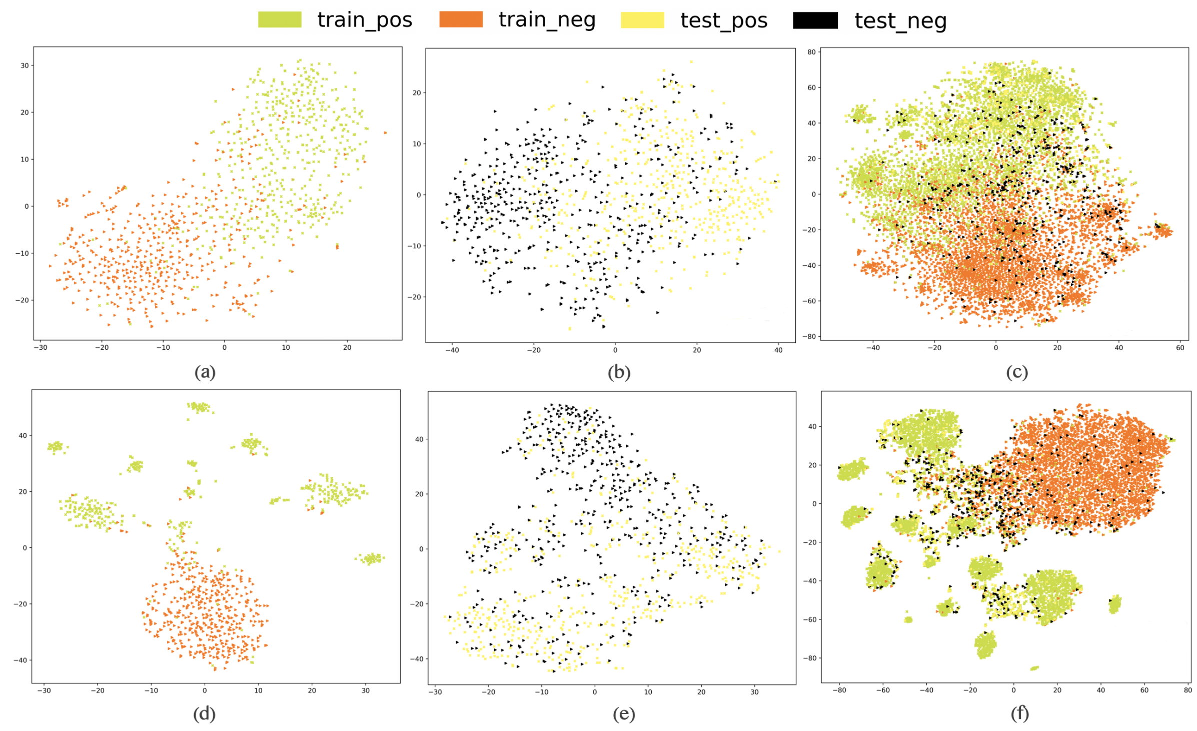


Figure 3: Visualization of t-SNE data distribution under CLIP(a,b,c) and UNITER(d,e,f) models.

Pre-trained model	**Fine-tuning**	
	dev	**test**
BERT txt (TB)	82.6	65.9
UNITER img	70.2	60.3
UNITER txt	76.8	60.7
UNITER img+txt	82.8	67.1
CLIP img (IB)	82.1	68.1
CLIP txt	82.0	66.8
CLIP img+txt	84.3	72.1

Table 2: Baseline performance of single-stream and dual-stream pre-trained models.

benchmarks, and the cross-modal learning objective is better learned through its pretraining given the improvement over the unimodality UNITER models.

The CLIP text model has a comparative performance with the TB, noting that there is no specified text pretraining objective in CLIP. With the well learned image and text unimodal semantics, the CLIP multimodal fine-tuning brings a marinal improvement, leading to 84.3 macro F1 score on dev dataset.

Overall the CLIP multi-modal fine-tuning performs better than the UNITER multimodal fine-tuning, both on dev and test dataset, with comparative unimodal performance at the same time.Thus CLIP is chosen as the pretrained model to provide image/text representations.

4.2 Domain Shift

The big performance gap between the dev and test set in Table 2 suggests domain shift between the train (dev) and test data. Domain shift can be simply expressed as Eq. 5, p_s denotes source data distribution and p_t denotes target data distribution.

$$p_s(x, y) \neq p_t(x, y) \tag{5}$$

$$p(x, y) = p(x|y)p(y) = p(y|x)p(x) \tag{6}$$

According to Bayesian joint probability distribution formula in Eq. 6, the analysis of inconsistent data distributions can be turned to the analysis of marginal probability distributions and conditional probability distributions.

1) $p_s(x) \neq p_t(x), p_s(y|x) = p_t(y|x)$
We consider the training set as the source domain and the test set as the target domain. The distribution of positive and negative samples in the training set is more separable, as shown in Figure 3-a and 3-d. However, the distribution of two classes in test set is mixed, especially the distribution of negative

samples drawn in black triangle shows a significant domain shift,the black triangle, as shown in Figure 3-b and 3-e. Therefore, we regard this as $p_s(x) \neq p_t(x)$ case.

2) $p_s(x) = p_t(x), p_s(y|x) \neq p_t(y|x)$
Another way is to observe the overall distribution in the data set clusters together, as shown in Figure 3-c and 3-f. It is indicated that the distribution of the training set and the test set have relatively low difference. $p_s(x) = p_t(x)$. Generally, the training and test sets are composed of easy samples and hard samples. A hard sample in terms of visualization is a positive sample running into the domain of a negative sample, or a negative sample distributed in the domain of a positive sample. The test set of this competition has a large number of ambiguous samples and difficult samples. This causes the deep learning model to crash, while the more interpretable XGBoost performs better.

To alleviate the problem, we exploit data augmentation and explored further with the XGBoost hyper-parameters

4.2.1 Extra Data for Better Generalization

The external searched-meme dataset and the misogynistic-meme dataset is added for training and the macro F1 score was improved by 2.1 points. This improvement is shown in Figure 4.

4.2.2 XGBoost on Domain Shift

XGBoost has many design parameters to prevent overfitting. These include the number of trees, tree depth, subsampling and colsampling, etc. In addition, there are two penalty terms in XGBoost. $\Omega(f)$ corresponds to Ω in Eq. 2. γ and λ denote the penalty factor, T is the number of leaf nodes. $||w||^2$ is equivalent to the L2 norm in Eq. 7 .

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2 \tag{7}$$

As shown in Table 3, the XGBoost classifier shows advantage over fine-tuning on the CLIP model. Different from the fine-tuning paradigm, the image-text CLIP feature performs worst on the XGBoost classifier, with 82.4 macro F1 score. The CLIP text model with XGBoost classifier achieves the best results on dev data, 90.1 macro F1 score, and CLIP image model with XGBoost achieves 85.2 on dev data, but achieves the highest macro F1 score on test data, both of them outperforms the CLIP multimodal fine-tuning. The CLIP image model with XGBoost classifier is chosen as our basis.

Pre-trained	**Fine-tuning**		**XGBoost**	
model	**dev**	**test**	**dev**	**test**
CLIP img	82.1	68.1	85.2	**77.6**
CLIP txt	82.0	66.8	90.1	65.4
CLIP img+txt	84.3	**72.1**	82.4	75.1

Table 3: XGBoost Performance on dev and test data compared with pretrained model fine-tuning.

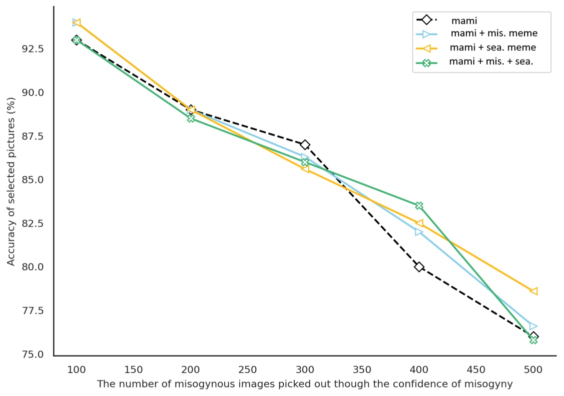


Figure 4: Accuracy of XGBoost classifier trained by different datasets. mis. meme is the misogynistic-meme datase and sea. meme is the searched-meme dataset.

4.2.3 Fine-tuning with Domain Adaptation Compared to the XGBoost Classifier

We apply domain adaption to the CLIP fine-tuning model by an adversarial loss between the source and the target domain following (Tzeng et al., 2015). The core concept is to fuse the distributions of source and target data by a domain classifier together with a domain confusion loss. Besides the standard cross-entropy loss for misogyny classification, the domain classifier (with loss L_{dm}) is on top of CLIP pretrained model to discriminate the source and target data, and the domain confusion loss L_{conf} forces the output of the domain classifier to be a uniform distribution, thus achieving the goal of fusing the source and target domain. By minimising two adversarial losses L_{dm} and L_{conf}, the performance of CLIP model improves by 1.5 on the macro F1 score of testing dataset (72.1->73.6) while it still falls behind the XGBoost.

4.3 Late Sequential Fusion and Mutual Adjustment

The confidence statistics of the XGBoost image (CLIP feature) and BERT text models are shown in Figure 5. It illustrates our intuition of the late sequential fusion. The upper left and lower right

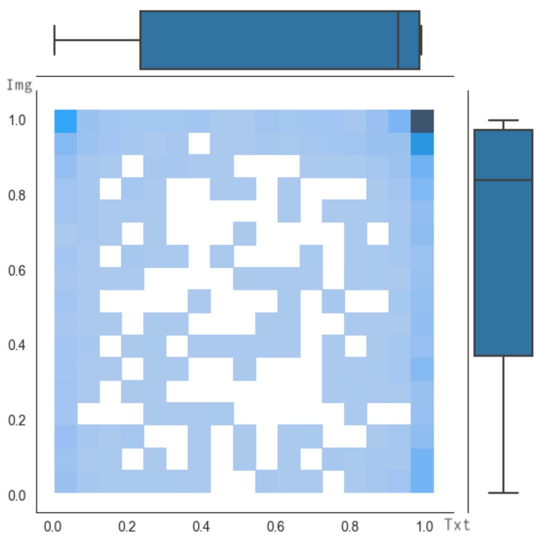


Figure 5: The joint density plot of visual and text modalities on test data.

Task	XG-Boost	Aug. data	Seq. fus.	Mut. adj.
sub-task A	77.6	79.5	81.9	83.4
sub-task B	71.1	-	71.9	73.1

Table 4: Ensemble performance of the system.

corner in the figure shows the disagreement of the image model and the text model. In the medium confidence interval of the XGBoost image prediction, the BERT text prediction can sometimes provide high confidence positive prediction. In addition, the box plot shows the positive skewness distribution in the BERT text model is more obvious than the CLIP-image XGBoost classifier, which means more negative samples are misjudged as positive by the BERT text model. The late sequential fusion boosts the macro F1 score of sub-task A (79.5->81.9) and sub-task B (71.1->71.9). And we get the leading score of 83.4 in sub-task A and 73.1 in sub-task B by the mutual adjustment of the two sub-tasks.

5 Conclusion

In this paper, we investigated the single-steam model UNITER and dual-stream model CLIP's performance on the downstream multimodal classification task, and compared the pretrain-and-fine-tuning paradigm over the feature-extraction-and-machine-learning-classification paradigm.

The experiment results show that the CLIP performs better than UNITER on the MAMI task, and is more robust on domain shift. The UNITER unimodal fine-tuning results are significantly worse than the unimodal pretrain model benchmark, suggesting its weakness in handling the complicated semantic logical relationship in the MAMI task. Wheras the structure of CLIP image feature extraction and XGBoost classificatin achieves the highest baseline performance.

We proposed the late sequential fusion scheme to fuse text information into our system PBR, and exploited extra data and mutual adjustment of the two sub-tasks to further improve the system performance. Our system ranks 1st place in both the sub-tasks in the leaderboard of the SemEval-2022 Task 5 MAMI.

References

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.

Terrance DeVries and Graham W. Taylor. 2017. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. Semeval-2022 task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Konrad Gadzicki, Razieh Khamsehashari, and Christoph Zetzsche. 2020. Early vs late fusion in multimodal convolutional neural networks. In *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, pages 1–6. IEEE.

Francesca Gasparini, Giulia Rizzi, Aurora Saibene, and Elisabetta Fersini. 2021. Benchmark dataset of memes with text transcriptions for automatic detection of multi-modal misogynistic content. *arXiv preprint arXiv:2106.08409*.

Dylan Grosz and Patricia Conde-Cespedes. 2020. Automatic detection of sexist statements commonly used at the workplace. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 104–115. Springer.

Ella Guest, Bertie Vidgen, Alexandros Mittos, Nishanth Sastry, Gareth Tyson, and Helen Margetts. 2021. An expert annotated dataset for the detection of online misogyny. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1336–1350.

Yuqi Huo, Manli Zhang, Guangzhen Liu, Haoyu Lu, Yizhao Gao, Guoxing Yang, Jingyuan Wen, Heng Zhang, Baogui Xu, Weihao Zheng, et al. 2021. Wenlan: Bridging vision and language by large-scale multi-modal pre-training. *arXiv preprint arXiv:2103.06561*.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Luis Perez and Jason Wang. 2017. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. 2015. Simultaneous deep transfer across domains and tasks. *arXiv preprint arXiv:1510.02192*.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*.

A Task Data Analysis

The 3 kind of semantic logical relationships between the meme image and meme text in the Introduction Section is illustrated in Figure 9. The subfigures a-c are the cases only text decide the meme's semantic, and d-f are cases that only image decide the semantic, lastly e-g are cases the image and text together form the complete semantic of the meme. Figure 8 shows the high frequency uni- and bigram text distribution. We analyse the top 30 frequent unigram and bigram features of the text input over training and testing distribution (with stop words filtering and ubiquitous words filtering such as "come, "makeameme", "org" which indicate sources of memes). These plots show significant bias, in terms of content and frequency, between train and test distributions.

B Hyper-parameter settings

XGBoost classifier hyper-parameters is shown in Table 5, the BERT fine-tuning hyper-parameters is in Table 6, and the UNITER fine-tuning hyper-parameters shown in Table 7.

Hyper-parameters	Value
objective	binary:logistic
n_estimator	800
learning_rate	0.03
subsample	0.90
max_depth	7
lambda	10
colsample_bytree	0.85
reg_alpha	10
reg_lamba	10
scale_pos_weight	15

Table 5: Hyper-parameters of XGBoost classifier

C keywords for crawling meme data from search engines

The keywords for misogynous memes are {'meme misogyny','meme anti-feminist', 'meme chauvinism woman', 'meme shaming/objectification/stereotype/violence/insult women/woman/girl/female/feminine', 'meme sexist' , 'woman/women/female/feminine hater' }. The keywords for non-misogynous memes are randomly selected neural words like {'meme happy girl', 'meme plants', 'meme school', 'meme actress'} etc.

Hyper-parameter	Value
learning rate	1e-5
learning rate decay	linear
warmup fraction	0.1
Adam ϵ	1e-6
Adam $beta_1$	0.9
Adam $beta_2$	0.98
gradient clip norm	1.0
Weight Decay	0.01
Dropout	0.1
Batch Size	32
Train Epochs	10

Table 6: Hyper-parameters for BERT fine-tuning

Hyper-parameter	Value
learning rate	1e-5
learning rate decay	linear
warmup fraction	0.1
Adam ϵ	1e-6
Adam $beta_1$	0.9
Adam $beta_2$	0.98
gradient clip norm	2.0
Weight Decay	0.01
Dropout	0.1
Batch Size (Token Batch)	5120
Train Epochs	10 for fine-tuning
max txt len	60

Table 7: Hyper-parameters for UNITER fine-tuning

D Experimental results for the late sequential fusion

Figure 6 shows the intial CLIP-image XGBoost classifier's tendency to misclassify the negative sample as positive samples. Figure 7 shows the intermediate ensemble performance.

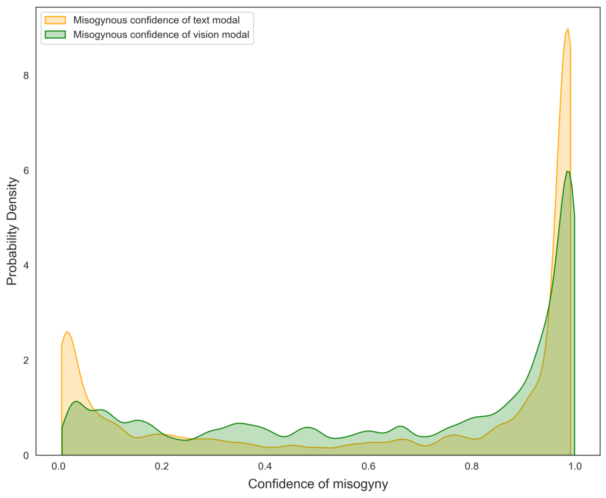


Figure 6: Confidence probability density distribution of textual and visual modalities in "CLIP + XGBoost". Aligned with the TSNE visualization, many negative examples are incorrectly identified as positive examples.

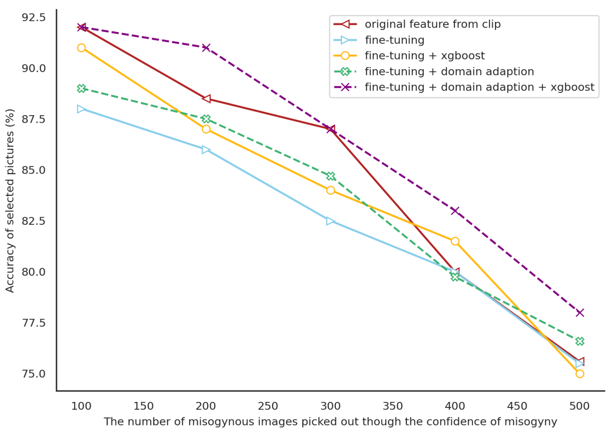
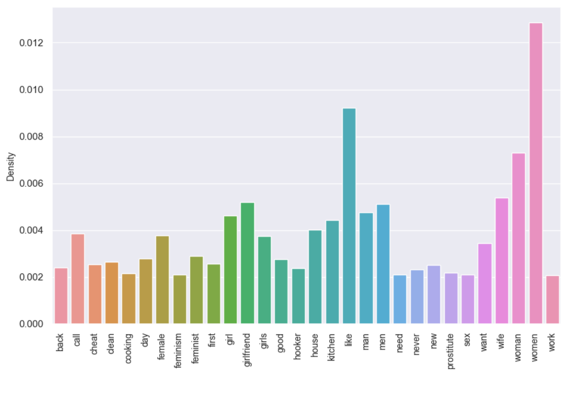
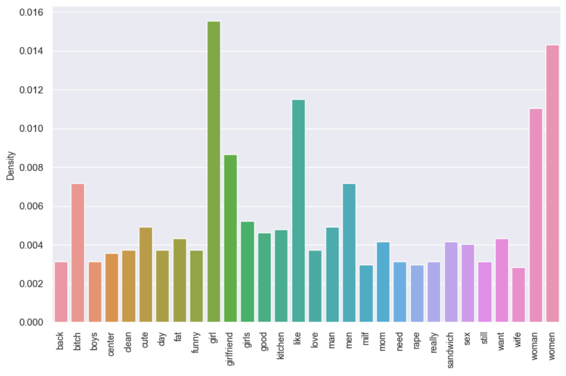


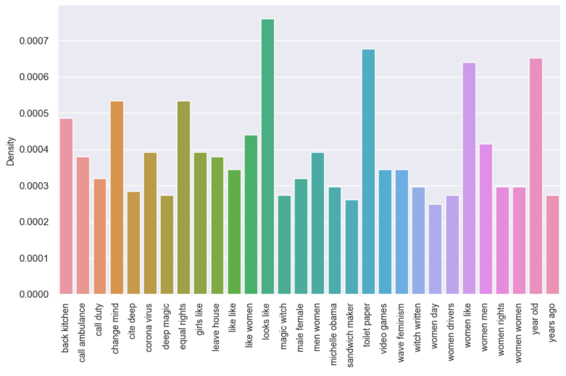
Figure 7: Some extra experimental for the combination of fine-tuning and XGBoost. Normal fine-tuning makes the model learn more towards the training data and performs relatively poorly in the test set. Fine-tuning with domain adaptation can improve the generalization ability of the model. Also according to the dashed line, it can be seen that XGBoot still has a large improvement in the results after fine-tuning.



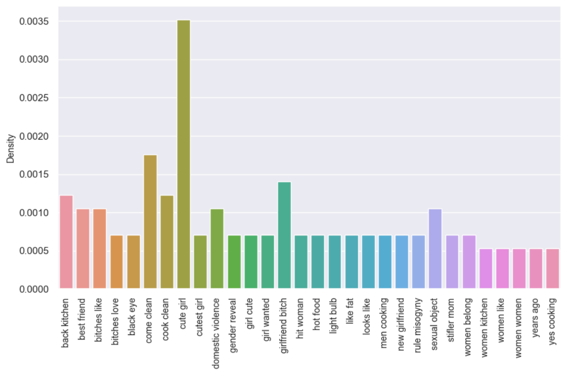
(a)



(b)

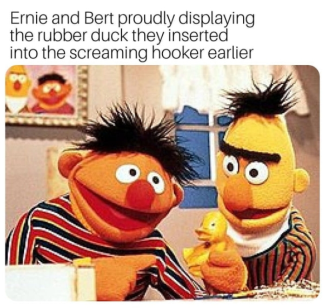


(c)

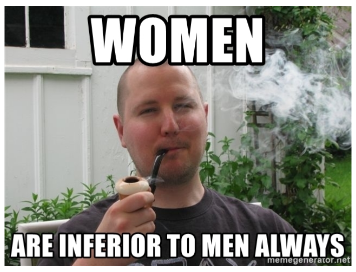


(d)

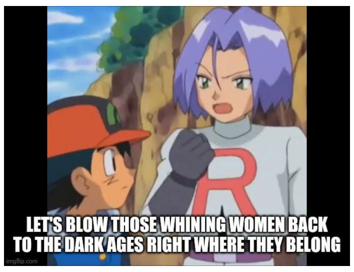
Figure 8: Text analysis of train dataset and test dataset. (a) and (b) corresponds to unigram features. (c) and (d) corresponds to bigram features of train and test dataset.


Ernie and Bert proudly displaying the rubber duck they inserted into the screaming hooker earlier

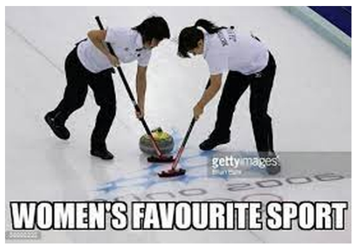
(a)



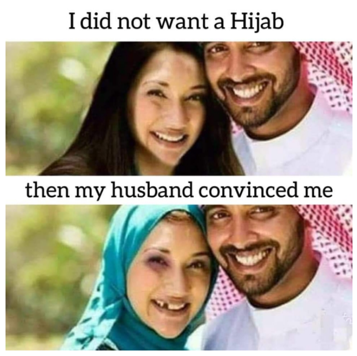
(b)


LET'S BLOW THOSE WHINING WOMEN BACK TO THE DARK AGES RIGHT WHERE THEY BELONG

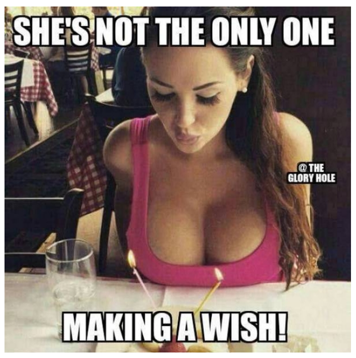
(c)



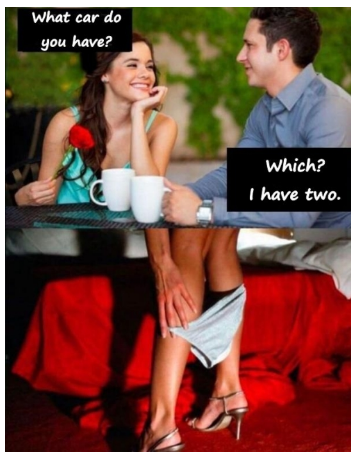
(d)



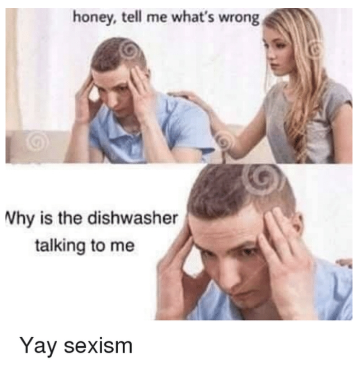
(e)



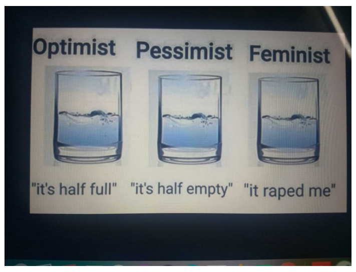
(f)



(g)


honey, tell me what's wrong
Why is the dishwasher talking to me
Yay sexism

(h)



(i)

Figure 9: Different image and text semantic relations in MAMI. a-c only text decide the meme semantic, d-f only image decide the meme semantic, g-i text and image together decide the semantic

ASRtrans at SemEval-2022 Task 5: Transformer-based Models for Meme Classification

Ailneni Rakshitha Rao[1], Arjun Rao[2]
[1]Indian Institute of Technology, Gandhinagar
[2]Chaitanya Bharathi Institute of Technology, Hyderabad
rao_ailneni@alumni.iitgn.ac.in
ugs18088_ece.arjun@cbit.ac.in

Abstract

Women are frequently targeted online with hate speech and misogyny using tweets, memes, and other forms of communication. This paper describes our system for Task 5 of SemEval-2022: Multimedia Automatic Misogyny Identification (MAMI). We participated in both the sub-tasks, where we used transformer-based architecture to combine features of images and text. We explore models with multi-modal pre-training (VisualBERT) and text-based pre-training (MMBT) while drawing comparative results. We also show how additional training with task-related external data can improve the model performance. We achieved sizable improvements over baseline models and the official evaluation ranked our system 3rd out of 83 teams on the binary classification task (Sub-task A) with an F1 score of 0.761, and 7th out of 48 teams on the multi-label classification task (Sub-task B) with an F1 score of 0.705.

1 Introduction

Despite its unique advantages, social media is considered to be one of the harmful elements of society, if not monitored properly. It has become a medium to express hatred towards particular groups, especially women. Women have a strong presence online and have become victims of systematic inequality and discrimination which is reflected from the behavior offline. Violence has increased to the point where for many girls, abuse is a day-to-day reality. A landmark survey conducted by Plan International in more than 20 countries has revealed shocking accounts of escalating online violence against girls and women, with respondents exposed to explicit messages, pornographic photos, cyberstalking, and other forms of internet abuse. The most common type of online harm includes using abusive and insulting language, followed by deliberate embarrassment, body shaming, and threats of sexual violence.

One of the most popular communication tools in social media is a *meme*. It is a combination of image and text, created typically for humor. SemEval 2022 Task 5 is the first misogynistic meme detection challenge that incorporates several categories such as stereotyping, shaming, objectification and violence.

Pre-trained language models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), etc., have emerged as the state-of-the-art models for many NLP tasks such as text classification, machine translation, sequence tagging, etc., mainly due to their rich contextual embeddings. Hence, we chose a transformer-based architecture to fuse both visual and textual features. Two types of pre-training techniques are explored in this paper: text-based and multi-modal-based. Instead of directly using a pre-trained text transformer, we further train RoBERTa on task-related data and fine-tune the model with extended visual features extracted from a image classification network (e.g. ResNet). We also use ensemble learning to combine the results of the two pre-training techniques.

We achieved significant improvement over baselines in both the sub-tasks. We were ranked (1) 3rd with an F1-macro score of 0.761 in sub-task A and (2) 7th with a weighted F1 score of 0.705 in sub-task B among 83 teams. We release the code for models and experiments via GitHub [1]

The rest of the paper is organized as follows: Section 2 describes the challenge, followed by a brief literature survey. Section 3 explains the proposed approach in detail while section 4 presents the experimental details required to reproduce the results. Results and analysis are shown in section 5. Finally, conclusions are drawn in section 7.

[1]https://github.com/rak55/ASRtrans-semeval2022

Label	No. of samples
Misogynous	5044
Shaming	1274
Objectification	2204
Violence	962
Stereotype	2844

Table 1: Distribution of training data.

2 Background

2.1 Problem Description

SemEval 2022 Task 5: MAMI: Multimedia Automatic Misogyny Identification (Fersini et al., 2022) consists of two sub-tasks: **Sub-task A** is a simple binary classification task to identify whether a meme is misogynous or not. **Sub-task B** is an advanced multi-label classification task, where memes are further classified among four categories namely *stereotype*, *shaming*, *objectification*, and *violence*.

2.2 Related Work

Multimodal data classification There are two types of approaches to multimodal data classification: late fusion and early fusion. Early works on multimodal data employed late fusion techniques such as combining major image features with bag-of-words-based text features (Tian et al., 2013). In this approach, two separate models are trained with images and text and their outputs are combined at a later stage. (Zhang and Pan, 2019) used a late fusion of CNN-based image features and RNN-based text features. The late fusion approach can be used even if one of the modalities is missing in the input. The disadvantage of this approach is, it fails to learn the interactions between different modalities. On the other hand, early fusion approaches use joint representations of images and text, thereby training a single model to learn within and across both modalities. With the development of pre-trained language models such as BERT, early fusion models like VisualBERT (Li et al., 2019), Vision and Language BERT (Lu et al., 2019), Visual-Linguistic BERT (Su et al., 2019), Multimodal Bitransformers (MMBT) (Kiela et al., 2019) and Learning Cross-Modality Encoder Representations from Transformers (Tan and Bansal, 2019) have quickly risen in popularity. However, early fusion models may perform poorly because using a single optimization strategy is sub-optimal for a model dealing with multiple modalities as explained in (Wang et al., 2019).

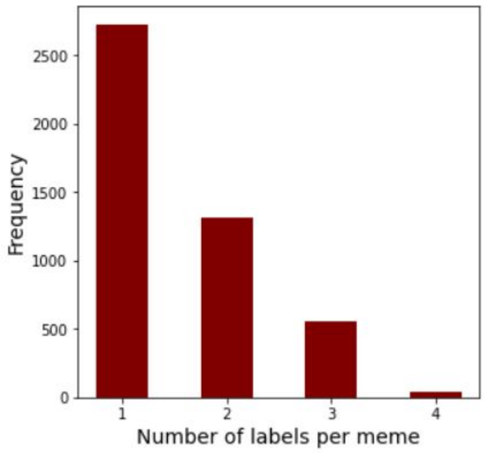


Figure 1: Frequency distribution of #labels per meme.

Multi-label classification There are mainly three different techniques to solve a multi-label classification problem (sub-task B) : Binary Relevance, Classifier Chains (Dembczyński et al., 2010) and Label Powerset method (Boutell et al., 2004). Binary Relevance treats each class independently and ignores label dependence. The Label Powerset method considers each combination of labels as a distinct class, thereby transforming a multi-label classification problem into a single-label problem. Classifier Chains connects binary classifiers in a chain such that the output of one classifier is treated as the input feature for the subsequent classifier.

3 System Overview

We explored models using both single-task and multi-task learning approaches. Since multi-task learning did not provide any significant improvement, our final model was trained using single-task learning. Consequently, we trained models using one modality (either text or image) and both modalities for the sake of comparison. We also compared the performance of models with and without multimodal pre-training.

3.1 Data

The training data provided for this task consists of 10,00 memes. The text transcription of memes along with the annotated labels for each image file is provided in a .csv file. Memes in the dataset are of random size. The distribution of different labels in the public training data is shown in Table 1. The frequency distribution of the number of labels per meme is shown in Figure 1. Each meme is labeled as either misogynous or non-misogynous

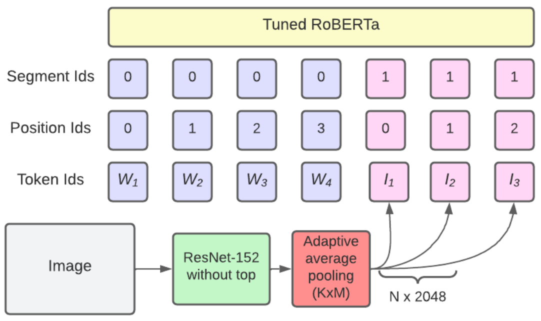


Figure 2: Architecture of the proposed model.

and in turn, misogynous memes are further classified among potential overlapping categories. 20% of data is randomly sampled from the training data for validation. The final test data provided for evaluation consists of 1000 memes.

3.2 Single modality approach

3.2.1 Text-based approach

We tested the following text-based models: i) A bidirectional LSTM model with Glove embeddings. ii) Pre-trained RoBERTa-base model fine-tuned with a classification head. iii) Ernie-2.0 (Sun et al., 2019), fine-tuned in the same way as RoBERTa-base model. RoBERTa model is further trained with external data and fine-tuned with training data for classification.

3.2.2 Leveraging External data

RoBERTa (Liu et al., 2019) has 12 layers, 12 heads, and a hidden layer dimension of 768. It is trained with masked language modeling on five datasets: BookCorpus, English Wikipedia, CC-News, OpenWebText, and Stories. We initiated it with pre-trained weights and trained on sexism-based datasets using the HuggingFace library. Sexism-based datasets are used since the language and content present is similar to our training dataset. (Grosz and CONDE-CESPEDES, 2020) introduced a labeled dataset consisting of sexist comments made in the workplace. We took 624 comments labeled 'sexist' out of 1137 comments present in this dataset. (Singh et al., 2021) presented a dataset to determine the use of sexism in English sitcoms from which, we extracted 1631 sexist text instances. From EXIST2021 dataset (Rodríguez-Sánchez et al., 2021-09), sexist text instances in English language are extracted. Altogether, we curated 5049 sexist text instances for training. Since the acquired dataset is small, we did not train the RoBERTa model from scratch. We call this model *tuned* RoBERTa (tRoBERTa) for the entirety of this paper.

3.2.3 Image-based approach

We fine-tuned several large-scale image feature extraction networks such as VGG-16 (Liu and Deng, 2015), ResNet-50, ResNet-152 (He et al., 2015) and Vision transformer (Dosovitskiy et al., 2021) for both the sub-tasks.

3.3 Multimodal input

We implemented the late fusion approach mentioned in section 3.2.1 by combining image features from VGG-16 and text features from a BiLSTM model. Besides, we also trained the state-of-the-art multimodal models such as VisualBERT and MMBT described in section 2.2.

VisualBERT is an integration of BERT and pre-trained object proposal system, Faster-RCNN. Similar to BERT, VisualBERT is pre-trained using two language model objectives: 1) Masked language modeling, where part of the input text is masked and predicted using contextual visual and text tokens. 2) Sentence-image prediction where the model determines if the text data matches the image or not. By combining image and text regions through a transformer, VisualBERT aims to learn useful alignments between them. For this purpose, it uses unordered visual embeddings extracted from an object detector, each corresponding to a bounded region in the image. In addition to BERT inputs, VisualBERT takes visual embeddings, visual token type ids, and visual attention masks as input. We fine-tuned VisualBERT for the task which is pre-trained on the VQA task with COCO dataset. Image features are extracted from a ResNeXt-based Faster RCNN, pre-trained on Visual Genome dataset.

MMBT combines representations from large language models and state-of-the-art convolutional neural networks in a straightforward way. It employs a BERT-base uncased model (12-layer 768-dim) trained on English Wikipedia. It uses N separate image embeddings extracted from the ResNet-152 network with average pooling over K x M grids (N = K x M). The dimension of each embedding is 2048. MMBT maps these image embeddings to BERT's token space using a set of randomly initialized mappings. The output of the [CLS] token in the last layer of BERT is given to a dense layer for classification. In our system, we

Model	Task A	Task B
BiLSTM + Glove emb.	0.585	0.571
RoBERTa (single-task)	0.646	0.629
RoBERTa (Multi-task)	0.648	0.631
Ernie-2.0	0.643	0.630
VGG-16	0.639	0.610
ResNet-50	0.617	0.608
ResNet-152	0.631	0.611
Vision Transformer	0.624	0.609
VGG-16 + BiLSTM	0.641	0.625
MMBT	0.725	0.669
VisualBERT	0.723	0.673
MMBT with tRoBERTa	**0.751**	**0.700**
Avg. Ensemble	0.761	0.705

Table 2: F1 score of all the major models on test dataset for both the sub-tasks. Avg. Ensemble is the weighted average of both the models.

use the tuned RoBERTa (tRoBERTa) described in section 3.2.2 instead of the BERT model used in the original implementation. The architecture of our proposed model is shown in Figure 2. This model is fine-tuned with a classification head on top by optimizing focal binary cross-entropy loss for multi-label classification.

3.4 Ensemble Learning

Multimodal models such as VisualBERT and MMBT differ in their training procedures and the datasets on which they are trained. Hence, they may focus on different aspects of the input. So, it is a good practice to combine the results of these two models to learn accurate representations. There are several ways to combine them: we can concatenate embeddings of different models and project them to a low dimensional space for prediction, but this will require high computational power. Instead, we can train these models independently and later combine their predictions. In a Weighted-Average Ensemble, results are obtained by taking a weighted average of the predictions. In this case, the weights are obtained by grid search on the validation dataset. Another way to combine the predictions is the Voting Ensemble method, where the class predicted by the majority of the models is considered as the final output. We experimented with both approaches and found that the weighted average method yields better results than the voting ensemble method.

Rank	Team	F1-macro
1	SRC-B	0.834
3	DD-TIG	0.794
5	NLPros	0.771
6	**ASRtrans**	0.761
61	Baseline_Text	0.640
62	Baseline_Image	0.639
79	Baseline_Image_Text	0.543

Table 3: Comparison of our sub-task A results with those on leaderboard.

4 Experimental Setup

We used pytorch (Paszke et al., 2019) and HuggingFace library (Wolf et al., 2019) for training and inference. All the models are trained on Google Colab. AdamW (Loshchilov and Hutter, 2019) optimizers with learning rates of 2e-5 and 5e-5 are used for training Visualbert and MMBT models respectively. Other text models such as RoBERTa and Ernie-2.0 also use AdamW optimizer with a learning rate of 2e-5. The maximum length of the text is limited to 50. We chose a batch size of 32 for training all the models. All the hyperparameters are tuned on the validation set which is 20% of the training data.

4.1 Data prepocessing

An input image is resized to 256 x 256 and then center-cropped to 224 x 224, followed by normalization before passing into MMBT. Text transcriptions of memes are cleaned to get rid of any URLs, HTML tags, and punctuation. Subsequently, they

Rank	Team	F1
1	SRC-B	0.731
7	NLPros	0.720
8	QMUL	0.713
14	**ASRtrans**	0.705
41	Baseline_Hierarchial	0.621
48	Baseline_Flat	0.421

Table 4: Comparison of our sub-task B results with those on leaderboard.

are annotated with the [CLS] token in the beginning before passing to the model. Removing stopwords showed a slight deterioration in the performance. Hence, we did not remove them.

4.2 Data Augmentation

Data augmentation is widely used to generate slightly variant larger datasets from the existing smaller ones. Since one recurring issue among all the models we trained is overfitting, three types of data augmentation techniques are tested to address this issue. We applied contextual augmentation for labeled sentences as proposed in (Kobayashi, 2018). In this method, we replace the words in a sentence with words predicted by a bi-directional language model. We also tested the back-translation approach proposed in (Sennrich et al., 2016) and easy data augmentation (EDA) techniques described in (Wei and Zou, 2019). We observed that while back-translation and contextual augmentation slightly improved the performance of the model, the use of EDA degraded it.

4.3 Loss functions

We trained our model for sub-task A with binary cross-entropy loss, whereas for sub-task B we used focal loss. Focal loss (Lin et al., 2020) is a form of cross-entropy loss, but it is dynamically scaled. It is computed as:

$$FL\left(p_t\right) = \left(1 - p_t\right)^{\gamma} \log\left(p_t\right)$$

By setting $\gamma > 0$, we are reducing the relative loss for easy, well-classified examples ($p_t > 0.5$). For prediction, we used a threshold of 0.35 as it produced better results on the validation set.

5 Results and Analysis

We have tested transformer and non-transformer-based models with features extracted from text, images, and a combination of text and images as input. We experimented with models that employ text and multimodal pre-training like MMBT and VisualBERT respectively. The results of these experiments are summarized in Table 2. We can infer from the results that pre-training transformer-based language models give better results compared to LSTM-based text models. This performance can be attributed to their rich contextual embeddings.

System	F1
MMBT with RoBERTa	0.682
+ tuned RoBERTa	0.690
+ focal bce loss	0.696
+ threshold at 0.35	0.700

Table 6: **Sub-task B (dev)**: Incremental analysis of our system.

No. of img embeds	F1
1	0.751
3	0.743
4	0.739
5	0.740

Table 5: Comparison of results of sub-task A with different number of image embeddings.

Our model (MMBT with tuned RoBERTa) performs slightly better than VisualBERT without any multimodal pre-training. This is because VisualBERT is pre-trained on Microsoft's image annotation dataset COCO and most real-world multimodal inputs are not as straightforward as a caption describing an image. For the model to adapt to the target domain, we need additional pre-training with the data of interest. In our case, instead of extra multimodal pre-training, we just trained the text model (RoBERTa) for a few epochs using external data. This is computationally more efficient and flexible. Furthermore, this approach can be easily applied to inputs with missing modalities.

The performance of our model on the development set with a different number of image embeddings is shown in Table 5. Using a single image embedding gave better results than using multiple ones. Also, an incremental analysis of our system is shown in Table 6. This shows the importance of further training RoBERTa and focal loss. Besides, combining VisualBERT and our model in an ensemble gave an additional performance boost of around 1% (F1 score) in sub-task A and 0.5% in sub-task B.

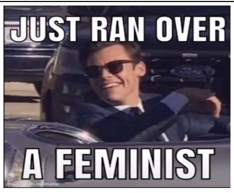
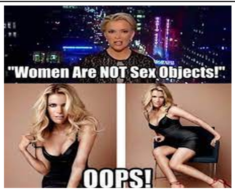
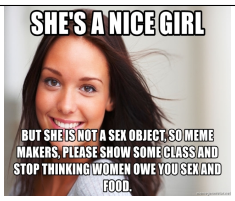
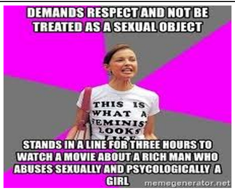
Meme			 SHE'S A NICE GIRL BUT SHE IS NOT A SEX OBJECT, SO MEME MAKERS, PLEASE SHOW SOME CLASS AND STOP THINKING WOMEN OWE YOU SEX AND FOOD.	 DEMANDS RESPECT AND NOT BE TREATED AS A SEXUAL OBJECT THIS IS WHAT A FEMINIST LOOKS LIKE STANDS IN A LINE FOR THREE HOURS TO WATCH A MOVIE ABOUT A RICH MAN WHO ABUSES SEXUALLY AND PSYCOLOGICALLY A GIRL memegenerator.net
Pred. label	stereotype, violence	stereotype, objectification	shaming, stereotype, objectification	objectification
Org. label	stereotype, violence	stereotype, objectification	non-misogynous	shaming, stereotype

Table 7: Sample test predictions of our model.

Comparison of the results of our model with the top submissions on the leaderboard for sub-task A and sub-task B are reported in Table 3 and 4 respectively. The official baselines for sub-task A are fine-tuned USE sentence embeddings (only text as input), fine-tuned VGG-16 model pre-trained on ImageNet dataset (only image as input), and concatenation of the above two models (image plus text as input). Baselines for sub-task B are: i) flat multi-label classification model with the concatenation of USE sentence embeddings and VGG-16 model features. ii) a hierarchical multi-label model based on USE sentence embeddings. Our model outperforms the best baseline model by 12% in sub-task A and 8.4% in sub-task B.

Error analysis of our model is shown in Table 7. The first two columns are examples of correctly classified memes whereas the last two columns are the incorrectly classified ones. The Model fails to classify the third meme as non-misogynous because it overlooks the word 'not'. This has been a common problem in language models such as BERT since they don't understand negation well, resulting in incomplete syntactic knowledge. In the last meme, the usage of the words *sexual objects* most likely misled the model into classifying it as *objectification* without taking into account 'not'. As this is a common error, in future, special methods should be devised to help models overcome it.

6 Conclusion and Future Work

We conducted experiments with models trained on features extracted from text, images, and both modalities combined for meme classification. To evaluate the text-based approach, we trained language models such as RoBERTa, Ernie-2.0, etc., and RNN-based models. Subsequently, we concluded that the transformer-based models produced best results. We showed that further training the RoBERTa model (tuned RoBERTa) with task-related data improves the performance. This tuned RoBERTa model combined with the features from ResNet-152 without multimodal pretraining performs slightly better than VisualBERT. Furthermore, an ensemble of these two models gave an additional performance boost. In the future, we plan to test other problem transformation approaches such as Classifier Chains and their variants for multi-label classification in order to accurately model label dependence and hierarchy. Additionally, we plan to train VisualBERT from scratch on meme datasets to see if there is an improvement in its performance.

References

Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. 2004. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771.

Krzysztof Dembczyński, Weiwei Cheng, and Eyke Hüllermeier. 2010. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 279–286, Madison, WI, USA. Omnipress.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias

Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Dylan Grosz and Patricia CONDE-CESPEDES. 2020. Automatic Detection of Sexist Statements Commonly Used at the Workplace. In *Pacific Asian Conference on Knowledge Discovery and Data Mining (PAKDD), Wokshop (Learning Data Representation for Clustering) LDRC*, Singapour, Singapore.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *CoRR*, abs/1909.02950.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *ArXiv*, abs/1805.06201.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327.

Shuying Liu and Weihong Deng. 2015. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734.

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

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

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *CoRR*, abs/1908.02265.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703.

Francisco Rodríguez-Sánchez, Jorge Carrillo-de Albornoz, Laura Plaza Morales, Julio Gonzalo Arroyo, Paolo Rosso, Miriam Comet, and Trinidad Donoso. 2021-09. Overview of exist 2021: sexism identification in social networks.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Smriti Singh, Tanvi Anand, Arijit Ghosh Chowdhury, and Zeerak Waseem. 2021. "hold on honey, men at work": A semi-supervised approach to detecting sexism in sitcoms. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 180–185, Online. Association for Computational Linguistics.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. VL-BERT: pre-training of generic visual-linguistic representations. *CoRR*, abs/1908.08530.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2019. Ernie 2.0: A continual pre-training framework for language understanding.

Hao Tan and Mohit Bansal. 2019. LXMERT: learning cross-modality encoder representations from transformers. *CoRR*, abs/1908.07490.

Lexiao Tian, Dequan Zheng, and Conghui Zhu. 2013. Image classification based on the combination of text features and visual features. *Int. J. Intell. Syst.*, 28(3):242–256.

Weiyao Wang, Du Tran, and Matt Feiszli. 2019. What makes training multi-modal networks hard? *CoRR*, abs/1905.12681.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Han Zhang and Jennifer Pan. 2019. Casm: A deep-learning approach for identifying collective action events with text and image data from social media. *Sociological Methodology*, 49(1):1–57.

UAEM-ITAM at SemEval-2022 Task 5: Vision-Language Approach to Recognize Misogynous Content in Memes

Edgar Roman-Rangel
Instituto Tecnológico Autónomo de México
Cd. de México
México
edgar.roman@itam.mx

Jorge Fuentes-Pacheco
CONACyT-CInC-UAEM
Cuernavaca, Morelos
México
jorge.fuentes@uaem.mx

Jorge Hermosillo Valadez
CInC - Universidad Autónoma del Estado de Morelos (UAEM)
Cuernavaca, Morelos
México
jhermosillo@uaem.mx

Abstract

In the context of the Multimedia Automatic Misogyny Identification (MAMI) competition 2022, we developed a framework for extracting lexical-semantic features from text and combine them with semantic descriptions of images, together with image content representation. We enriched the text modality description by incorporating word representations for each object present within the images. Images and text are then described at two levels of detail, globally and locally, using standard dimensionality reduction techniques for images in order to obtain 4 embeddings for each meme. These embeddings are finally concatenated and passed to a classifier. Our results overcome the baseline by 4%, falling behind the best performance by 12% for Sub-task B.

1 Introduction

The Multimedia Automatic Misogyny Identification (MAMI) competition (Fersini et al., 2022) consists in the identification of misogynous memes, taking advantage of both text and images available as source of information. The task was organized around two main sub-tasks. Sub-task A: a basic task about misogynous meme identification, where a meme should be categorized either as misogynous or not misogynous; Sub-task B: an advanced task, where the type of misogyny should be recognized among potential overlapping categories such as stereotype, shaming, objectification, and violence.

In this paper, we present a proposed solution for Sub-task B only, which consists of a framework for extracting lexical-semantic features from text and combine them with semantic descriptions of images, together with image content representation. We propose to use a pre-trained BERT model (Devlin et al., 2019) as lexical feature enhancer, and to use a vision-language model (Zhang et al., 2021) as visual descriptor.

The rest of this paper is organized as follows. We introduce our multimodal framework in Section 2. In Section 3 we present and discuss our results. We conclude the paper in Section 4.

2 Methods

Fig. 1 shows a diagram of the method that we propose to describe memes using both visual and text inputs. It is known that local descriptors can provide rich representations, as they can extract fine details from local areas within documents (Lowe, 2004). Therefore, we compute two types of description for each modality -visual or text-. Namely, global and local descriptors, and then concatenate the resulting four descriptors into a single vector that we use for classification.

More specifically, we relied on transfer learning coupled with fine tuning procedures, in which one pre-trained model was further adjusted for each of the four input modalities, and for the five classes problem presented by the MAMI competition.

2.1 Image descriptors

We describe images at two levels of detail. First globally using a pre-trained CNN, and also locally by detecting and describing individual objects within the images.

2.1.1 Global descriptors

In order to generate a global visual description of the image, we use the InceptionV3 (Szegedy et al., 2016) network pretrained on ImageNet (Russakovsky et al., 2015) to perform a fine tuning on the MAMI dataset. The classification head is replaced with a global average pooling, a dropout layer, and 4 dense layers with weights randomly initialized. The first three dense layers have 1024, 512, and 64 units. The last dense layer has five units with sigmoid activations.

To train the network, first the convolutional basis is frozen and the parameters of the classification head are optimized for ten epochs using a learning rate of $1e^{-3}$. Next, all parameters are unfrozen and retrained with a smaller learning rate ($1e^{-5}$). The training is stopped by using the regularization strategy of Early Stopping to avoid overfitting. Once the training process is completed, a 64-vector of floating-point values is generated for each image. This descriptor is obtained in inference mode from the output of the penultimate dense layer.

2.1.2 Local descriptors

We detect and describe each of the objects contained in the images by means of the pre-trained VinVL model (ResNeXt-152 C4 architecture) (Zhang et al., 2021). VinVL was pre-trained on four public datasets specialized in object localization, so it considers up to 1848 object categories and 524 attribute categories (nouns and adjectives respectively).

This stage generates a feature vector of length 2048 for each object within the image. More precisely, this step produces a matrix of varying length according to the number of objects detected in an image, where each component is a fixed-length vector of size 2048. We post-process this matrix using Principal Component Analysis (PCA) on both of its axes, and recovering the eighth principal components for each axis. This is, we identify features corresponding to the eighth most relevant objects in a given image, as well as those corresponding to the eighth most relevant variables describing each object representation, both of them in an orthogonal space of PCA that is independent across images.

This PCA processing results in a 64-D representation of the visual features for all object detected within an image.

2.2 Text representations

Analogous to the image processing stage, we also describe the meme's text transcriptions at two levels of granularity. First, generating an embedding for the whole sentence, and then incorporating individual word representations.

2.2.1 Contextual embeddings

We generated a global sentence embedding for each meme transcription. This was performed using a pre-trained BERT model (Devlin et al., 2019). Namely, the small uncased BERT "L-4_H-512_A-8" for English language (Turc et al., 2019). This model was used up to the layer that produces its so-called pooled output, which provides a sentence embedding vector of 512 elements. We connected such output to a classification multi-layer perceptron (MLP) for fine tuning the model.

The classification MLP, added to BERT for fine tuning, consists of two fully-connected hidden layers of 512 and 64 perceptrons, and a final 5 units layer that performs multi-class multi-label classification for the five possible labels defined for this challenge. Both hidden layers contain 'swish' activation functions, while the output layer implements 'sigmoid' non-linearities to ensure that it outputs values bounded between 0 and 1. We chose 'swish' as activation function to obtain a smooth transition between the positive and negative sides of the response space of the non-linear projection (Ramachandran et al., 2017). Dropout layers with rate equal to 0.1 were added in between fully-connected layers.

We performed fine tuning of this model on the training set of the MAMI challenge. First, warming up only on the added MLP during 8 epochs. Then, on the full model during 8 more epochs. Both training stages made use of the Adam optimizer (Kingma and Ba, 2014), the first one with initial learning rate of $3e^{-3}$ and the latter with $3e^{-5}$.

2.2.2 Enhanced vocabulary representations

We enriched the text modality description by incorporating word representations for each object present within the images. To this end, we relied on

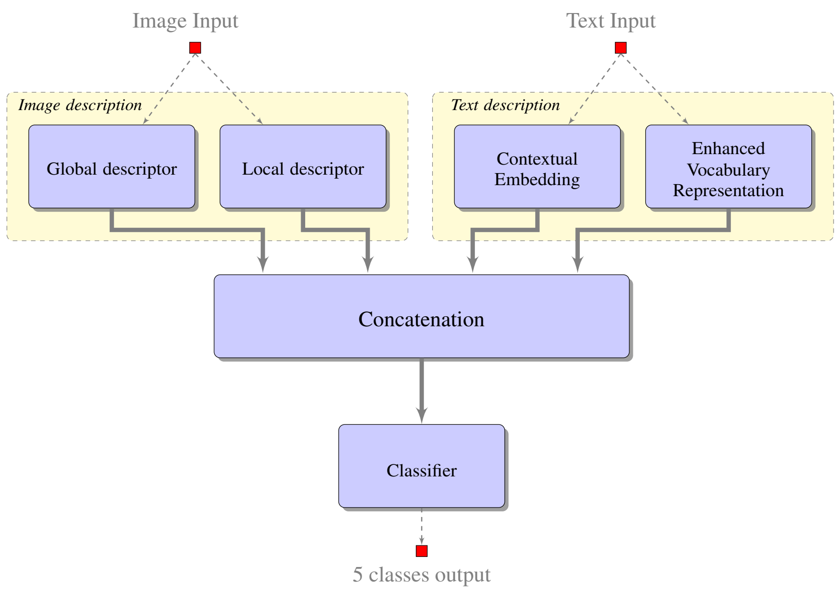


Figure 1: Architecture proposed

the pre-trained VinVL model (Zhang et al., 2021), which is used to segment objects, and provides a list of nouns and adjectives for each segmented object. Some examples of words generated in this stage are: woman, man, red, blue, thin, tall, etc.

This process produces as many lists as there are objects detected within the image. We concatenated all individual words discovered for the same image into a single vector. Then, we used this vector of nouns and adjectives to train a classification network with the same architecture and process as the one explained in sec. 2.2.1, i.e., the architecture and training process are repeated on a different set of parameters.

2.3 Classifier

After the individual training of each of the models described through sections 2.2 and 2.1, we used them in inference mode to process their corresponding inputs, and obtained their respective outputs up to their next-to-last layers. This step produces a 64-D vector for each of the four models.

By concatenating these four representations into a single vector, we produced a multi-modal feature vector of length 256. This resulting vector is used as input for a final MLP classification model, which consists of nine fully-connected layers as shown in Fig. 2.

This final model also uses ‘swish’ activation functions for all hidden layers, and the ‘sigmoid’ activation function for its output layer. As shown in Fig. 2, this model is organized in four blocks, each of which is composed by: a regularization process plus two consecutive fully-connected layers. Regularizers are either dropout or batch normalization. Dropout regularizers use a dropout rate of 0.3. Similarly to the previous individual models, this one also uses an output layer of five perceptrons corresponding to each of the five possible classes in the classification task.

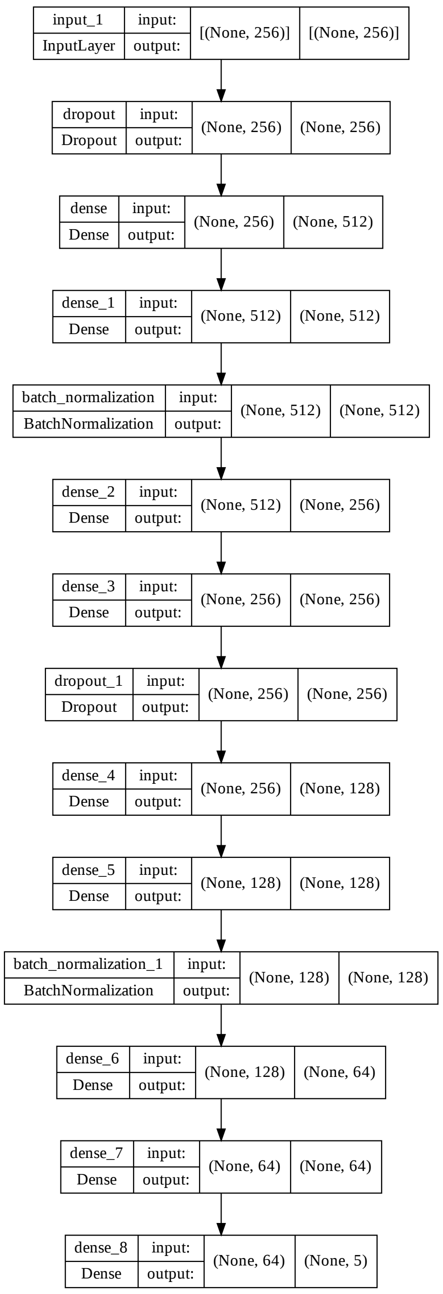
2.4 Training

The final classification model was trained using binary cross entropy as loss function, and the Adam optimizer (Kingma and Ba, 2014) with default parameters as implemented in tensorflow: learning rate $\eta = 0.001$, decay for the smoothing of first and second order moments $\beta 1 = 0.9$ and $\beta 2 = 0.999$, and minimum tolerance $\epsilon = 1e{-}7$. We trained this model during 50 epochs with batches of size 64.

We decided to stop training at 50 epoch, as we observed after several attempts that the loss function has consistently converged by then, for both training and validations sets, i.e., no overfitting was observed.

3 Results and discussion

Table 1 shows the accuracy obtained by our model during training and validation, as well as the F1 score obtained on the test set as reported by the server of the MAMI challenge. These scores are presented for Task B (multi-class - multi-label classification) and for three models: the baseline as reported by the organizers of the challenge; our



input_1 | input: | [(None, 256)] | [(None, 256)]
InputLayer | output:
dropout | input: | (None, 256) | (None, 256)
Dropout | output:
dense | input: | (None, 256) | (None, 512)
Dense | output:
dense_1 | input: | (None, 512) | (None, 512)
Dense | output:
batch_normalization | input: | (None, 512) | (None, 512)
BatchNormalization | output:
dense_2 | input: | (None, 512) | (None, 256)
Dense | output:
dense_3 | input: | (None, 256) | (None, 256)
Dense | output:
dropout_1 | input: | (None, 256) | (None, 256)
Dropout | output:
dense_4 | input: | (None, 256) | (None, 128)
Dense | output:
dense_5 | input: | (None, 128) | (None, 128)
Dense | output:
batch_normalization_1 | input: | (None, 128) | (None, 128)
BatchNormalization | output:
dense_6 | input: | (None, 128) | (None, 64)
Dense | output:
dense_7 | input: | (None, 64) | (None, 64)
Dense | output:
dense_8 | input: | (None, 64) | (None, 5)
Dense | output:

Figure 2: Classification MLP that processes the merged multi-modal feature descriptor.

proposed model; and the best model submitted to the leader board of the competition.

Model	Training	Validation	Test (F1)
Baseline	–	–	0.621
Ours	0.937	0.895	0.646
Best	–	–	0.731

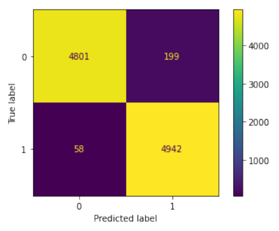
Table 1: Performance on Sub-task B from the baseline model, our proposed model, and the top model reported on the leader board. Columns Training and Validation report accuracy, while column Test reports the F1 score.

Fig. 3 shows the confusion matrices produced by our model on the instances of the MAMI challenge, computed on the joined training and validation sets. We show five confusion matrices because the categories are not mutually exclusive. Each of the matrices contains true negatives [0,0], false positives [0,1], false negatives [1,0], and true positives [1,1]. In all cases, the accuracy is greater than 0.90, with the "misogynous" class having the highest score (0.97) and the stereotyped class the lowest (0.90). Using the F1 measure, the "shaming" and "violence" (the most unbalanced) classes have the worst performance with 0.68 and 0.71 respectively. This situation is due to the small number of true positive instances in these two classes, which might bias the model towards the prediction of the negative label. Meanwhile, the "misogynous", "stereotype" and "objectification" (the most balanced) classes have a similar performance with an F1 score above 0.82. Moreover, the classes with the lowest performance, have a proportionally much lower number of training examples. This fact limits the fine tuning of the model parameters in the overall training process.

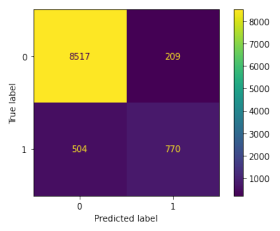
Fig. 4 visualizes the ROC curve, which represents the rate of true positives versus the rate of false positives. As in the confusion matrices, it can be observed that the best performance is obtained in the class "misogynous", while the worst occurs with the classes "shaming" and "violence".

4 Conclusions

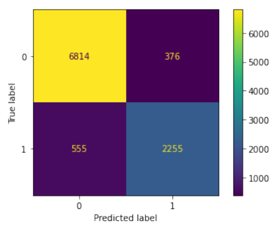
In this paper, we proposed a framework for extracting lexical-semantic features from text and combine them with semantic descriptions of images in the context of the Multimedia Automatic Misogyny Identification (MAMI) competition 2022. Our results overcame the baseline by 4%, but fell behind the best performance by 12% for Sub-task B. Our model's performance could be explained by the un-



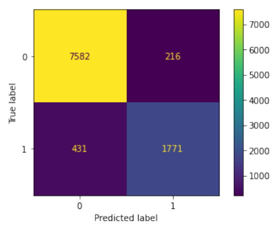
(a) Misogynous.



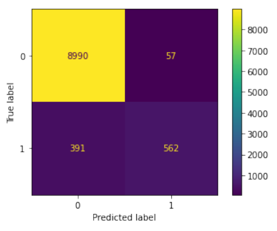
(b) Shaming.



(c) Stereotype.



(d) Objectification.



(e) Violence.

Figure 3: Confusion matrices for the 5 classes in the challenge.

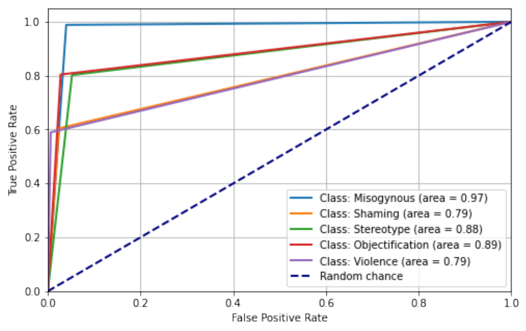


Figure 4: ROC curve for the five classes in the MAMI challenge: "misogynous", "shaming", "stereotype", "objectification", and "violence". "Shaming" and "violence" curves are overlapped.

balanced classes and low number of examples, and, therefore, is limited in this sense, achieving a performance below 0.9 and around 0.7, for accuracy and F1 measure, respectively. Still, for balanced classes we obtained a performance above 0.9 and 0.8, in terms of accuracy and F1 score respectively. As future work we propose three alternatives: 1) To fine tune the classification threshold of the sigmoid output layer of the model, independently for each class, i.e., not all classes need to have 0.5 as classification threshold; 2) Optimize the loss function directly on the F1 score; and, 3) Weight the contribution of each class in the overall loss function during backpropagation.

Aknowledgements

This research has been made possible thanks to generous support from the Consejo Nacional de Ciencia y Tecnología (CONACyT). And the support of the Asociación Mexicana de Cultura A.C.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, Minnesota.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. Semeval-2022 task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *Proc. 3rd Int. Conf. Learning Representations*.

David G. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2(60):91–110.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. 2017. Searching for activation functions. *arXiv preprint arXiv:1710.05941*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

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

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv:1908.08962v2*.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. Vinvl: Revisiting visual representations in vision-language models. In *CVPR 2021*.

JRLV at SemEval-2022 Task 5: The Importance of Visual Elements for Misogyny Identification in Memes

Jason Ravagli
Università degli Studi di Firenze
Florence, Italy
jason.ravagli@stud.unifi.it

Lorenzo Vaiani
Politecnico di Torino
Turin, Italy
lorenzo.vaiani@polito.it

Abstract

Gender discrimination is a serious and widespread problem on social media and online in general. Besides offensive messages, memes are one of the main means of dissemination for such content. With these premises, the MAMI task was proposed at the SemEval-2022, which consists of identifying memes with misogynous characteristics. In this work, we propose a solution to this problem based on Mask R-CNN and VisualBERT that leverages the multimodal nature of the task. Our study focuses on observing how the two sources of data in memes (text and image) and their possible combinations impact performances. Our best result slightly exceeds the higher baseline, but the experiments allowed us to draw important considerations regarding the importance of correctly exploiting the visual information and the relevance of the elements present in the memes images.

1 Introduction

Even though many advances and initiatives have been carried on in the last decades, misogyny and gender discrimination still represent a serious social problem. Comments and posts with hate messages or bad jokes having the female gender as the target are published every day on social networks. Memes are the most popular means of communication on the internet. A meme basically is an image with overlayed text. The combination of image and text should transmit in a clear and direct way a message, which is often funny or ironic. However, many users use them to spread hate or discriminatory messages against certain categories of people. Considering the number of contents that are published online every moment, social network administrators require automatic tools to identify and remove this type of memes. Such tools can be an effective way to fight and stop sexist and misogynous manifestations online.

For the SemEval-2022 it has been proposed the Multimedia Automatic Misogyny Identification (MAMI) task (Fersini et al., 2022), which consists of identifying misogynous memes.

In this work, we present our proposed solution for subtask A of MAMI, which is a binary classification problem where a meme must be categorized as misogynous or non-misogynous. Our solution exploits the multimodal nature of the available data (a meme is represented by a pair image - overlayed text) by using a VisualBERT model together with a Mask R-CNN. Our solution classified 35^{th} out of 69 participants, getting an F1 score 2% higher than the best baseline. Beyond metric values, our work highlights the different importance of various visual elements and how their usage can affect the performances.

2 Task Overview

The purpose of MAMI is identifying misogyny contents and messages within memes through the use of visual and textual information. Organizers set up this task as a competition, providing an annotated dataset and proposing two different sub tasks.

2.1 Dataset

The training dataset contains 10000 memes. Each sample is composed of a pair (image, text) corresponding to the visual and textual information inside the meme.All texts transcripts are already extracted from pictures and available in English language. Data samples are labeled with five different binary tags, each aimed at identifying a different type of hate content: misogyny, objectification, violence, stereotype and shaming. More than one class can characterize a single data sample.

During the final phase of the MAMI challenge, task organizers released a test dataset, made by 1000 elements, which labels were released at the end of the competition only. Therefore, we decided to reserve 25% of the labeled training samples for the validation set to perform our experiments.

In addition to general information, we compute some image-related statistics to analyze and compare all dataset splits. In particular, we use Mask R-CNN (He et al., 2018) classification and bounding-box regression branches to identify which, how many and how large are the most frequent elements within the images. We try two different Mask R-CNN pretraining: the former trained on COCO (Lin et al., 2015) (91 classes) while the latter trained on LVIS version 0.5 (Gupta et al., 2019) (1230 classes).

Computed statistics for both mask R-CNN pretrainings are reported in Table 1. Comparing the two, it is possible to observe how the COCO pretraining allows identifying elements in greater quantity and of greater dimensions than the LVIS one. This is a strange result considering the higher number of classes contained in LVIS. Looking in detail at the detected objects we noticed that the majority of elements identified by the COCO-pretrained network belong to the class "person" ($\sim 63\%$ for train and validation, $\sim 68\%$ for test). This category is not present among those of LVIS, which instead detects as most popular categories some types of clothing and jewelry, such as swim suits, dresses, necklaces and earrings (each appearing at most in $\sim 8\%$ of predicted classes). Moreover, COCO pretraining performs better also in other domains different from photos, such as drawings and cartoons, where LVIS pretraining is often not able to detect and extract any relevant patch. A last noteworthy fact is the dimensional discrepancy between the test and the other splits when using the COCO pretraining. Even though the detected object ratio is lower on the test set, the percentage of covered picture is higher, implying that elements in test images are larger but in less quantity than in the other splits.

2.2 Sub-tasks

Regarding the goals of the challenge, the MAMI task is split up into two sub-tasks:

- Sub-task A consists of a binary classification problem, where each sample/meme must be categorized as misogynous or non-misogynous. The two classes are balanced.

- Sub-task B is a multi-class and multi-label classification problem, where each meme/sample must be assigned labels belonging to 5 different classes: misogynous, stereotype, shaming, objectification, and violence. It is an advanced task since the labels identify more in-depth the type of offensive content and the classes are unbalanced.

Our team worked on a solution for sub-task A only.

3 Related Works

Hate speech detection in text data has been deeply studied in the last few years. State-of-the-art approaches usually apply transformers-based methods achieving impressive results. (Mozafari et al., 2020) for example proposed different hybrid architectures to fine-tune a BERT model (Devlin et al., 2019) for detecting offensive tweets. However, identifying hate content in multimodal data requires correlating visual and textual information and introduces an additional challenge. Multimodal transformers, such as VisualBERT (Li et al., 2019) and ViLBERT (Lu et al., 2019), are currently the state-of-the-art models for these types of problems.

In 2020 Facebook AI organized the Hateful Memes Challenge (Kiela et al., 2021). The competition was similar to MAMI and consisted of categorizing a meme as hateful or non-hateful. The dataset contained memes with various types of hate messages (e.g. targeting an ethnicity, a religion, or the sexual orientation of people). Winning solutions all used ensembles of multimodal transformers-based networks. (Zhu, 2020) won the competition using a complex and task-specific pipeline to extract additional information from the data with which to fine-tune multimodal transformers networks. (Muennighoff, 2020) and (Velioglu and Rose, 2020), who ranked respectively second and third, proposed simpler ensemble methods based on VisualBERT and VilBERT-derived architectures.

Many other works regarding meme analysis for hate speech detection can be found in the literature. Each has its own peculiarities, often related to information extraction techniques, especially from the visual content. Among them, the most used are text removal, object detection, image captioning, and entity recognition, such as in (Deshpande and Mani, 2021), (Pramanick et al., 2021) and (Lee et al., 2021).

4 Method

In this section, we present architectures and preprocessing steps used to address sub-task A of MAMI.

		COCO			LVIS		
Split	Size	Detected Objects	Objects Ratio	Selected Area	Detected Objects	Objects Ratio	Selected Area
Train	7500	21480	2.86	51.55%	13665	1.82	13.20%
Validation	2500	7174	2.87	52.31%	4460	1.78	12.79%
Test	1000	2490	2.49	60.89%	1439	1.44	12.29%

Table 1: dataset statistics related to image content. For each split and for each pretraining version of Mask R-CNN are reported the total number of detected objects, the average number of elements per image and the average percentage of pixels per image contained in all the selected boxes .

First of all, we exploit single modalities separately, through state-of-the-art deep models, to investigate their relevance and the quantity of information they carry on. Subsequently, we combine image and text embeddings to estimate the performance of a multimodal approach.

The strategies mentioned above can be considered as three baseline models in which the information from each modality is retrieved without taking into account the other one and then used both alone or together. Afterward, we adopt early fusion methods to create a more informative embedding of a whole meme, extracting information jointly from both modalities with the same model. This allows to directly obtain a final representation depending on both text and image, thus removing the need for a fusion point between unimodal embeddings.

4.1 Text Encoding

Transformer architectures (Vaswani et al., 2017) are currently the state-of-the-art models in NLP as regards the generation of textual embedding. Among them, BERT (Devlin et al., 2019) is the most renowned sentence encoder, with top-level performances in encoding sentence semantics.

Our text-based baseline model exploits BERT encoder to convert the input sentence into a 768-dimensional embedding vector, obtained with a mean pooling operation over all the output tokens. The classification is accomplished by a single fully connected layer. Both sentence encoder and classification head are trained end-to-end.

4.2 Image Encoding

Convolutional Neural Networks (LeCun et al., 1999) are the type of model most commonly employed in image analysis. CNNs consist of a stack of convolutional layers that extract visual elements from the image, assigning each an appropriate relevance. Among all CNNs, VGG16 (Simonyan and Zisserman, 2015) is one of the most known and used.

Our image-based baseline end-to-end model uses the VGG16 feature extractor along with a multi layer perceptron to obtain a 2048-dimensional embedding vector from the entire original input image. Once again, the classification is accomplished by an additional fully connected layer, as happens for the text, and the two pieces of architecture are trained end-to-end.

4.3 Multimodal Fusion

As a naive multimodal solution, we combine each modality embedding extracted with the previously described techniques. Adopting this implementation we build an end-to-end architecture with two input branches, one for each modality, and a single classification head that takes as input the concatenation of text and image representation vectors, thus setting a late fusion point in the system.

4.4 Multimodal Extraction

Baseline architectures presented in previous subsections suffer from several disadvantages. First and foremost, the choice of fusion point is a critical decision during the construction of multimodal architectures. A late fusion point does not allow information contained in one modality to affect the embedding creation of the others, limiting the influence between modalities only through the back-propagation steps along with the end-to-end architecture. On the other hand, an early fusion has proven to be the best choice for a wide range of other tasks, such as in (Barnum et al., 2020) and (Peinelt et al., 2020).

Another weakness is the usage of the entire input image. The overlayed text in memes is often characterized by a showy style. Since transcriptions are available and typography is unlikely to bring any useful information about hate content, the text should be removed from the pictures, or simply not considered, to avoid affecting the image representations with noise data.

VisualBERT (Li et al., 2019) is a novel transformer designed to address vision-and-language tasks. It reuses self-attention mechanism to align elements of the input text and regions in the input image. VisualBERT is able to address all the weaknesses highlighted above. Input tokens originate from both text and image concurrently, implicitly setting an early fusion point at the model input stage. Furthermore, image input tokens are typically extracted from the original picture as small patch representations, instead of using the whole figure, allowing to focus on important details and ignoring text, background and noise elements. This architecture has been successfully applied to other tasks involving memes (Velioglu and Rose, 2020).

The criterion adopted to select the input patches is based on Mask R-CNN. The same pretrainings mentioned in the dataset sections are used to retrieve multiple Region Of Interest (ROI) in the picture. Then, if a ROI overcomes a fixed confidence threshold, it is select as input token for VisulBERT.

Similarly to BERT, VisualBERT provides a 768-dimensional vector for each input tokens, based on both image and text information, that is finally fed as input for the fully connected classification layer.

5 Experimental Results

5.1 Experimental Design

We try several configuration of the proposed architecture. During the ROIs identification step we exploit both COCO and LVIS pretraining, feeding the model with the patches coming from both Mask R-CNN versions, both separately and simultaneously. Another setting of our architecture concerns the use of transformer output embeddings as input for the classification head: we feed the final fully connected layer with both the CLS token embedding only and the average of all input token embeddings. During our experiments, one of the major issues we faced was the small dimension of the dataset and, consequently, the risk of overfitting. Thus, we used our validation set, obtained with an hold-out splitting as described in 2.1, to monitor the model performance during training. Then, the test set is evaluated selecting the best model according to the results obtained on validation set. Task organizers designated the F1 score as the official metric for the competition, so we used it for model selection.

Algorithms' configuration. We train our models for 25 epochs to minimize BCE loss with AdamW (Loshchilov and Hutter, 2019), using a batch size of 32 and 10^{-5} as learning rate. Many of the best models were obtained within of the first 10 epochs, so there were no need for longer training sessions. For the sake of reproducibility, the code is publicly available in the project repository [1].

Hardware Settings. Experiments were perfomed on a machine equipped with AMD® Ryzen 9® 3950X CPU, Nvidia® RTX 3090 GPU, and 128 GB of RAM running Ubuntu 21.10.

5.2 Results

Table 2 shows F1 scores on validation and test set of our baselines and proposed methods. We also report the best organizers baseline result, which was released at the end of competition.

As we can see, VisualBERT methods perform slightly better than our baseline approaches. Among them, the unimodal BERT and the multimodal fusion method turn out to be our best baselines on the validation and test sets respectively, while exploiting visual information only gives unsatisfactory results. This leads us to formulate some considerations about the nature of the task. First of all, not in all memes the quantity of relevant information is equally distributed between the two data sources (image and text). Indeed, some memes can be formed by a neutral image but a very offensive text or by a neutral or ironic test and a tacky image. The former can be successfully recognized by a text-only approach, which can manage to classify also the latter if correctly trained on identifying malicious messages in ironic texts. On the contrary, image-only methods can only spot misogyny in memes with explicit offensive images. Furthermore, simple approaches that focus on whole images like our VGG16 will fail to link different objects in the image scene and to understand the context depicted in it. As a result, the network will work properly only if the meme image clearly contains the target of the hate message (e.g. a woman on the foreground).

However, in some misogynous memes the hate message is formed by the combination of an apparently innocent meme and a neutral text. VisualBERT should be capable of merging the two

[1]The GitHub project repository is available at `https://github.com/VaianiLorenzo/SemEval-2022-MAMI`

sources of information and putting them into context, identifying these subtle cases. Indeed, it significantly outperforms baseline methods in both validation and test set.

Focusing on the VisualBERT-based methods, we found that using the CLS token over the average of the output embeddings for the classification gave better results in general. We can see from Table 2 that all the three pretraining modalities for Mask R-CNN led to similar results on the validation set, with the LVIS one performing slightly better and achieving an F1 score of 0.823. However, when evaluating on the test set, VisualBERT with Mask R-CNN pretrained on the COCO dataset performed the best, with an F1 score of 0.67 and an improvement of 4% over the other two methods and 2% compared with task organizers baseline. According to the results, giving VisualBERT features from both pretrained modalities gave no improvement. The motivation behind this can be the high overlap probability between patches: combining them brings no additional information and introduce redundancy.

The performance gap between the COCO-pretrained model and the LVIS one on the test set compared to the validation can be explained by the statistics described in Section 2.1. The pretraining on COCO allows the Mask R-CNN to detect ROIs containing whole people, that is, it allows to provide to VisualBERT features regarding the potential targets of the misogynous messages. Since more people are found in the test set images, VisualBERT can exploit the visual information in a more effective way in those memes. Some examples of critical types of memes discussed in this section and their analysis can be found in the Appendix A.

Due to the limited number of examples, our models showed signs of overfitting after a few epochs. Using the whole available training set brought no improvements. Hence, we tried some solutions to help the model to generalize better on unseen data. We did a pre-train phase where we applied our VisualBERT to the Hateful Memes task before fine-tuning it on the MAMI one with a lower learning rate. Unfortunately, this did not bring any improvement. The model only converged faster on the MAMI training set, maybe due to a similarity between the data of the two competitions. Afterward, we tried to keep the VisualBERT layers frozen and gradually unfreezing them during training, but also in this case we got no improvements.

Model	**Val F1 score**	**Test F1 score**
Task Baseline	-	0.650
BERT	0.786	0.607
VGG16	0.696	0.571
BERT + VGG16	0.756	0.628
VB (COCO)	0.811	**0.670**
VB (LVIS)	<u>0.823</u>	0.631
VB (LVIS + COCO)	0.818	0.632

Table 2: F1 scores of our VisualBERT-based architectures compared with both our and organizers baselines. Best results obtained for validation and tests splits are underlined and highlighted in bold respectively.

6 Conclusions and Discussion

In this work we demonstrate how leveraging the multimodal nature of data allows to achieve a significant boost in performance when facing tasks involving memes. Moreover, we confirm that an early fusion point between modalities implemented using VisualBERT, the current state-of-the-art architecture for vision-and-language tasks, can lead to satisfactory results to the MAMI task. The reliability of this model allows us to focus on other crucial aspects of the problem, such as the data to use as input to the network. While unimodal transformers successfully identify offensive content in textual data, state-of-the-art computer vision models obtain lower results on this task when analyzing visual information only. Catching the message of a meme requires understanding the context created by the text and the various elements depicted in the image. A deep learning model will struggle to form this context if the visual data are analyzed as a whole. Therefore, we have to first extract the relevant parts from the image and use their features separately as input to our multimodal network. By doing this, VisualBERT is capable of correlating text and visual scene to create the aforementioned context. Our experiments showed that the most relevant elements in the images are people, likely the target or the offender mentioned in the text.

Despite these discoveries, the extraction of information from the pictures remains the main obstacle of this task. As a future work, we plan to thoroughly investigate the content analysis of images. In particular we would like to enrich the preprocessing stage, performing captioning and entity detection of the image, in order to transpose some visual information in textual format.

References

George Barnum, Sabera Talukder, and Yisong Yue. 2020. On the benefits of early fusion in multimodal representation learning.

Tanvi Deshpande and Nitya Mani. 2021. An interpretable approach to hateful meme detection.

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

Elisabetta Fersini, Francesca Gasparini, Aurora Saibene Giulia Rizzi, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Agrim Gupta, Piotr Dollár, and Ross Girshick. 2019. Lvis: A dataset for large vocabulary instance segmentation.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2018. Mask r-cnn.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2021. The hateful memes challenge: Detecting hate speech in multimodal memes.

Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio. 1999. Object recognition with gradient-based learning. In *Feature Grouping*. Springer.

Roy Ka-Wei Lee, Rui Cao, Ziqing Fan, Jing Jiang, and Wen-Haw Chong. 2021. Disentangling hate in online memes. *Proceedings of the 29th ACM International Conference on Multimedia*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2015. Microsoft coco: Common objects in context.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. 2020. A bert-based transfer learning approach for hate speech detection in online social media. In *Complex Networks and Their Applications VIII*, pages 928–940, Cham. Springer International Publishing.

Niklas Muennighoff. 2020. Vilio: State-of-the-art visio-linguistic models applied to hateful memes.

Nicole Peinelt, Dong Nguyen, and Maria Liakata. 2020. Better early than late: Fusing topics with word embeddings for neural question paraphrase identification.

Shraman Pramanick, Shivam Sharma, Dimitar Dimitrov, Md Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021. Momenta: A multimodal framework for detecting harmful memes and their targets.

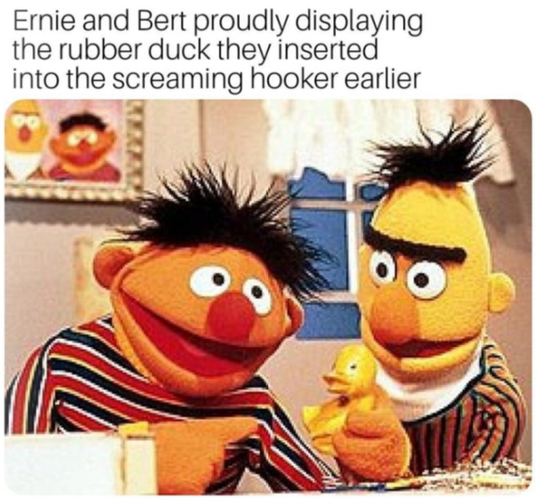
Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition.

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

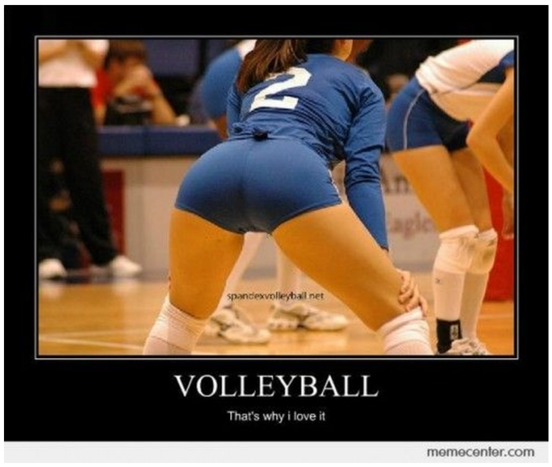
Riza Velioglu and Jewgeni Rose. 2020. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution.

A Memes Examples

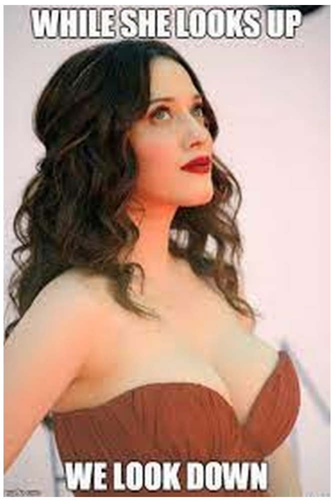

Ernie and Bert proudly displaying the rubber duck they inserted into the screaming hooker earlier

(a) Meme with a neutral image and offensive text.

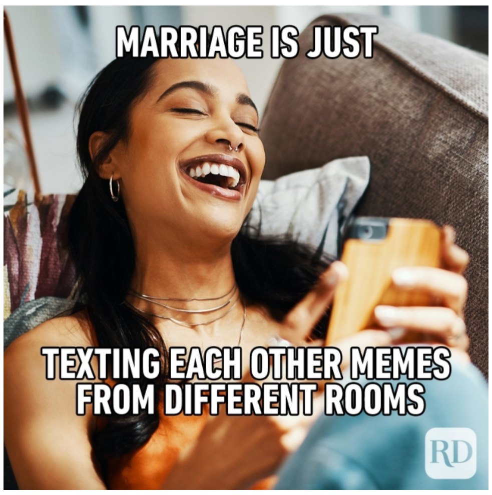


(b) Meme with neutral/ironic text and tacky image.

Figure 1: two example of misogynous memes where one of the two modalities, image in (a) and text in (b), contains neutral and /or not useful information according to the task goal. These memes highlight the importance of a multimodal approach.



(a) misogynous meme.



(b) not misogynous meme.

Figure 2: Two memes from test set depicting a woman in the foreground. The image-only baseline model classifies both memes as misogynous, probably due to the presence of a woman as main element in the picture. On the other hand, our best performing model predicts the appropriate label, demonstrating that the text is definitely needed to correctly classify these type of memes.

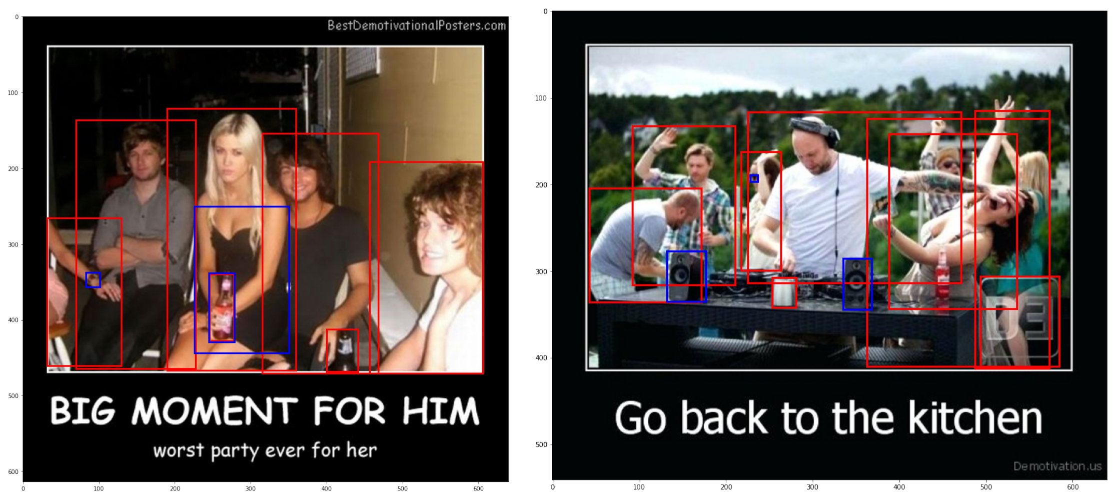


Figure 3: Two misogynous memes depicting several people. The highlighted boxes are the Mask R-CNN ROIs: red ones from the COCO pretraining and blue ones from the LVIS pretraining. As we can see, people are detected only by the former, while the latter retrieves mainly small objects. Moreover, it is possible to notice blue boxes are often contained in red ones, justifying why using COCO and LVIS pretraining jointly there is no improvement in performances.

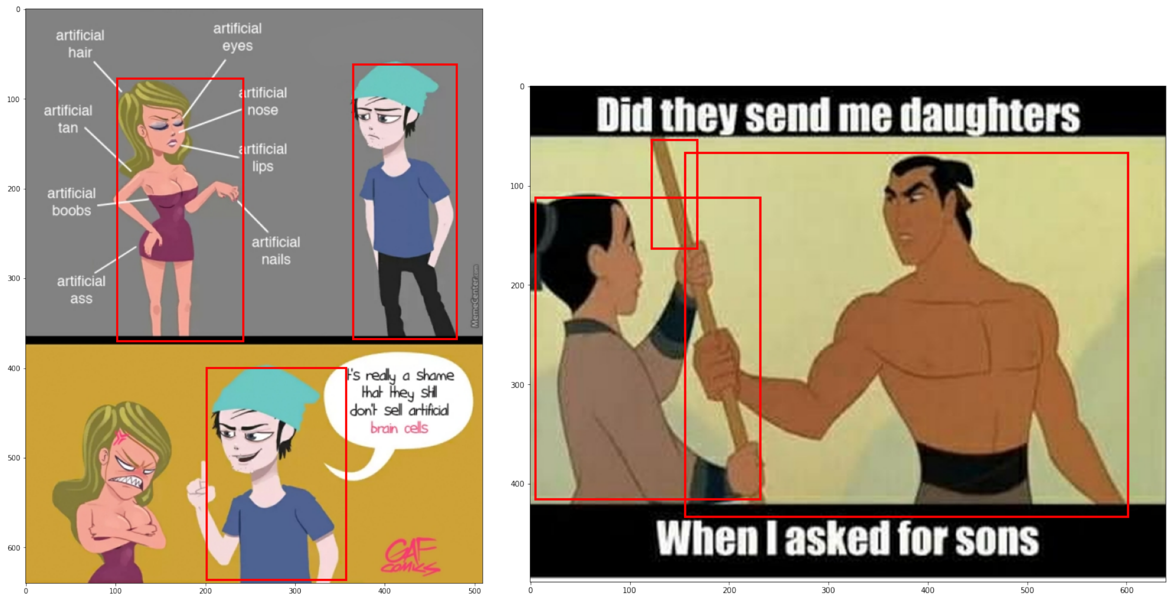


Figure 4: Two misogynous memes with cartoon style. The bounding boxes are all retrieved with Mask-R CNN pretrained on COCO, which is able to detect people also in images other than photos. On the contrary, the LVIS-pretrained model is often not able to identify any details in these kind of pictures.

UPB at SemEval-2022 Task 5: Enhancing UNITER with Image Sentiment and Graph Convolutional Networks for Multimedia Automatic Misogyny Identification

Andrei Paraschiv, Mihai Dascalu, Dumitru-Clementin Cercel
University Politehnica of Bucharest, Faculty of Automatic Control and Computers
{andrei.paraschiv74, mihai.dascalu, dumitru.cercel}@upb.ro

Abstract

In recent times, the detection of hate-speech, offensive, or abusive language in online media has become an important topic in NLP research due to the exponential growth of social media and the propagation of such messages, as well as their impact. Misogyny detection, even though it plays an important part in hate-speech detection, has not received the same attention. In this paper, we describe our classification systems submitted to the SemEval-2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification. The shared task aimed to identify misogynous content in a multi-modal setting by analysing meme images together with their textual captions. To this end, we propose two models based on the pre-trained UNITER model, one enhanced with an image sentiment classifier, whereas the second leverages a Vocabulary Graph Convolutional Network (VGCN). Additionally, we explore an ensemble using the aforementioned models. Our best model reaches an F1-score of 71.4% in Sub-task A and 67.3% for Sub-task B positioning our team in the upper third of the leaderboard. We release the code and experiments for our models on GitHub[1].

1 Introduction

The web and social network platforms, in particular, have become a significant part of our modern social lives. Sharing information, opinions, news, and jokes through these platforms with friends and family are now daily routines. One of the most prevalent forms of jokes in social networks are memes. Internet memes are small cultural units that are transformed, mixed, and shared using online platforms, often spreading in a viral manner (Milner, 2013). A prolific part are image-based memes, often available as templates that are accompanied by humorous or witty text. Unfortunately, a considerable proportion of the memes shared by Internet users are offensive or even hateful messages[2].

Detecting hate and offensive speech is a significant task for any online platform. Not only companies have this legal obligation in most countries, but also such language establishes a toxic environment that is detrimental to any online community on the long run. Hate speech can take multiple forms, but it is most frequently encountered as a disparaging message on the basis of a characteristic as race, gender, religion, and other criteria; Misogyny is one such frequent form specific to meme culture (Drakett et al., 2018; Phillips, 2012). Detecting hate speech is often a hard problem, even in an unimodal setting since the message often relies on the context, addresses current events, and incorporates cultural knowledge that cannot be easily incorporated into an automated model. The multi-modality of Internet memes increases the difficulty of the task since many memes can have a seemingly benign text that, contextualized with the associated image, becomes offensive or hateful.

Multi-modal hate speech detection had less attention in the research literature than traditional text-only methods. In the past two years, several datasets and challenges have addressed this by proposing detection tasks on meme-based data (Kiela et al., 2020; Gasparini et al., 2021; Miliani et al., 2020). Misogyny detection, as a subgroup of hate speech detection tasks, has also been more frequently encountered in research in recent years. The series of Automatic Misogyny Identification tasks proposed at IberEval 2018, EVALITA 2018, EVALITA 2020, and TRAC-2020 (Fersini et al., 2018a,b, 2020; Kumar et al., 2020; Bhattacharya et al., 2020) focused on the classification of tweets in English, Spanish, Italian, Bangla, and Hindi languages. In these tasks, researchers identified misogynous tweets and classified them as aggressive/non-aggressive or active/passive.

[1] https://github.com/readerbench/semeval-2022-task-5

[2] https://www.hmc.org.uk/blog/third-teenage-boys-admit-sending-receiving-racist-homophobic-content-online/

Semeval-2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification (Fersini et al., 2022) is a multi-modal classification task, aiming to detect misogynous memes by leveraging both image and text information. The task includes two sub-tasks: a binary identification of misogynous/non-misogynous memes (Sub-task A), and a multi-label classification distinguishing between the types of misogyny, namely: stereotype, shaming, objectification, and violence (Sub-task B).

In this paper, we present our contribution to this task by proposing two architectures based on the pre-trained multimodal UNITER (UNiversal Image-TExt Representation) model (Chen et al., 2020), as well as an ensemble from these two models. UNITER is an early fusion model pre-trained on large text-image datasets. UNITER leverages visual and location features extracted with Faster R-CNN (Ren et al., 2016), together with WordPiece encodings (Wu et al., 2016) derived from text tokens using a Transformer-based model (Vaswani et al., 2017). UNITER learns a generalizing representation for the text-image context by bringing the visual and text representations in a common embedding space. We use UNITER at the core of our two architectures: the first one enhances the pre-trained model with image sentiment features using a VGG-19 model (Vadicamo et al., 2017), while the second leverages graph convolutions on a co-occurrence graph (Kipf and Welling, 2016) built from an external dataset.

2 Background

Multi-modal tasks were traditionally associated with visual question answering (Goyal et al., 2017), image captioning (Gurari et al., 2020), audio-visual speech recognition (Paraskevopoulos et al., 2020), or cross-modal retrieval (Wang et al., 2016). With success of competitions like the Hateful Memes Challenge (Kiela et al., 2020), more research focused on multi-modal offensive classification. Pre-trained transformer models such as ViLBERT (Lu et al., 2019), VisualBERT (Li et al., 2019), LXMERT (Tan and Bansal, 2019), Oscar (Li et al., 2020), and others, dominated the competition leaderboard, either as stand-alone models or in large ensembles. While considering UNITER, Lippe et al. (2020) used an ensemble that placed them in the top 5 teams.

3 Method

We explored two ways of enhancing UNITER, first by adding a unimodal late fusion with a visual sentiment classifier, and second by using a multimodal early fusion with a modified Vocabulary Graph Convolutional Network (VGCN) (Lu et al., 2020; Paraschiv et al., 2021).

Dataset analysis and preprocessing

The training dataset contains 10,000 records, half of them misogynous. One misogynous record can have one or more of the four labels. The class distribution among the types of misogyny is as follows: shaming - 1,274, stereotype - 2,810, objectification - 2,208 and violence - 953 samples.

The text modality from the dataset was obtained, as far as we can tell, through OCR without any manual cleaning. This lead to the inclusion of date, times, mobile carrier names, Facebook user names or words on some unrelated objects in the image like *"Verizon LTE 4:41 PM Bikram Dec 11 at 12:31 AM"*. Additionally, many memes contain the watermark of the publishing website: "imgflip.com", "makeameme.org", "memez.com". Since most memes use full uppercase fonts, the letter casing was not reliable throughout the dataset and we choose to lowercase all training entries.

In our experiments, we tried two types of data cleaning techniques: one where we remove all timestamps and date mentions using the SUTime library (Chang and Manning, 2012) and a second with the supplementary step of removing all website mentions and Twitter usernames from the text.

Visual Sentiment-enhanced UNITER

Offensive texts, hate speech, and misogynous language are often correlated with negative sentiments, whereas the tone, context, and content is often highly loaded with polarized language (Ali et al., 2021; Gitari et al., 2015); as such, our intuition to enhance the UNITER model with a sentiment classifier. We focused only on image modality since large language models often already capture features required for text sentiment analysis. All images were classified using a pre-trained VGG-19 model (Simonyan and Zisserman, 2014) fine-tuned on the T4SA dataset (Vadicamo et al., 2017). The resulting 4,096 sized feature vectors were fused by concatenation with the UNITER's pooled output and classified through a fully connected layer in

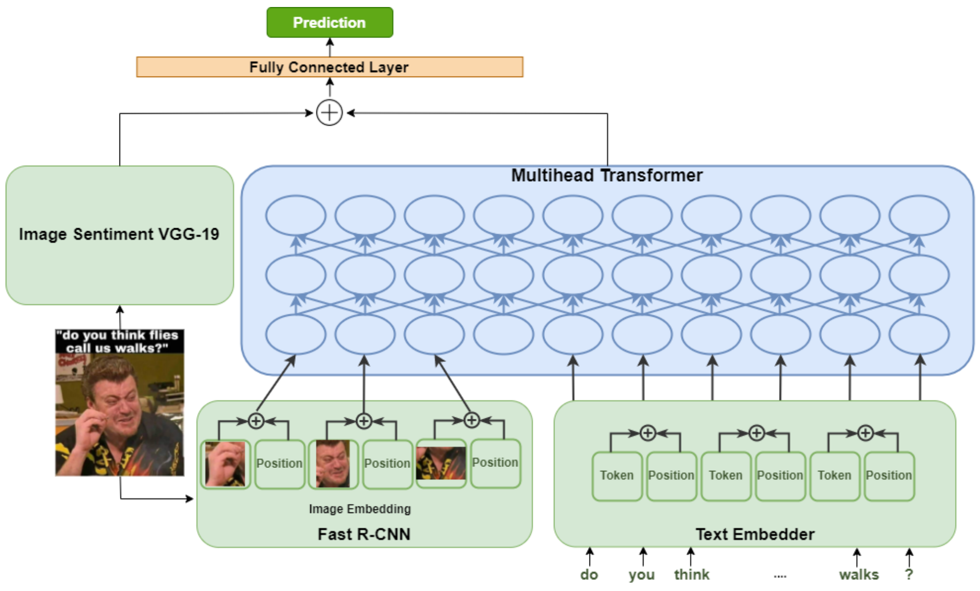


Figure 1: Illustration of the proposed UNITER-Sentiment model.

the classes for each sub-task (see Figure 1).

VGCN-enhanced UNITER

Graph enhanced BERT models have proven to be powerful for text classification (Mamani-Condori and Ochoa-Luna, 2021), even in multimodal tasks (Vlad et al., 2020). Using a GCN on a vocabulary graph (Kipf and Welling, 2016), the model can train an embedding layer to be fused with the Transformer embedding, before being processed by the multi-head attention layers. For our architecture, we employ a novel approach, namely to create a heterogeneous graph with nodes that represent text tokens and objects detected by the R-CNN layer, in the corresponding image.

A Kaggle dataset[3] containing 3,000 meme templates and their various possible text captions, totaling 533,827 text records, was used to build the aforementioned graph. The same R-CNN layer and object-token encoding as the UNITER model were considered to create a co-occurrence graph, having nodes as BERT-token-IDs and detected object-IDs, while edge values were computed using Point-wise Mutual Information (PMI). In contrast to (Lu et al., 2020), the obtained graph is independent of the training dataset and can be used for several tasks in the same domain.

In the training step, the UNITER image and text embeddings are fed through a GCN layer on top of the pre-built graph, thus generating a new embedding vector. The concatenated text, image, and graph embeddings are then processed by multi-head attention layers, while the pooled outputs are classified using a fully connected layer (see Figure 2).

[3]https://www.kaggle.com/zacchaeus/meme-project-raw

Ensemble model

In order to leverage the learning of both proposed models, we also utilize an ensemble formed from multiple versions of both models, trained on different train/dev/validation splits, that are then combined via a soft voting scheme. The best performing trained versions evaluated on our test set were picked in the ensemble. For our final submission, we used the votes from 2 UNITER-Sentiment with UNITER-base, 2 with UNITER-large, and 7 UNITER-VGCN with UNITER-large. We chose these components of the ensemble based on the results for Sub-task B on our validation set, as seen in Table 1. Details on general hyper-parameters across all models are presented in the subsequent section. For UNITER-Sentiment models, we used 150 warm-up steps with a weight decay of 0.01, in contrast to only 120 warmup steps with a weight decay of 0.1 for UNITER-VGCN models. All UNITER-VGCN models in our ensemble had a graph embedding size of 16 and the edges had a minimum normalized PMI (Bouma, 2009) of 0.3. Since we trained each of the 11 ensemble components on a different train/dev/validation partition with the same seed, each configuration converged on different weights, with different performances. The results of all 11 models were combined through

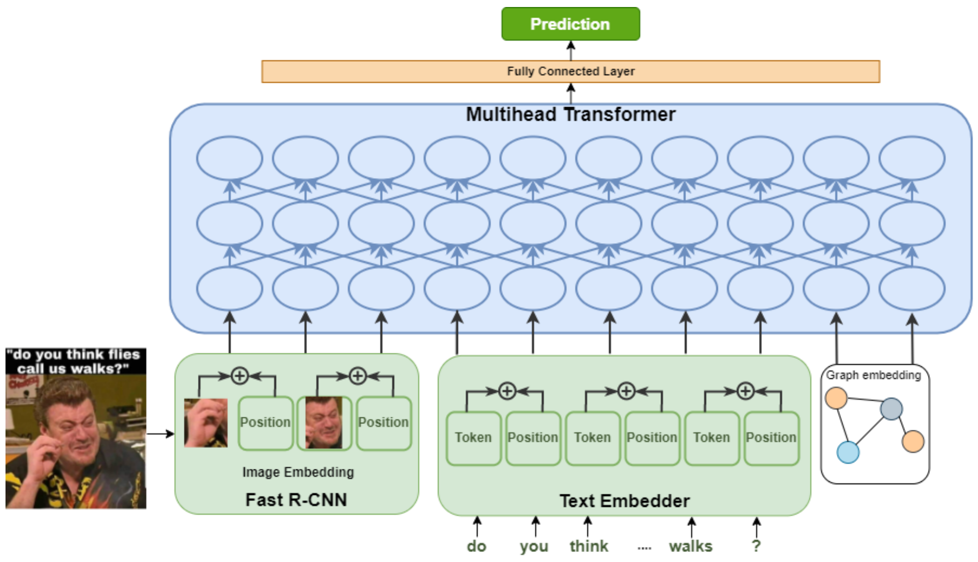


Figure 2: Illustration of the proposed UNITER-VGCN model.

a soft-voting scheme based on the average predicted probability for each label. Additional configurations were trained, but overall this ensemble had the best test performance among our submissions to SemEval-2022 Task 5.

3.1 Experimental Setup

Since the regions of interest (ROI) detection in the R-CNN layer[4] returns the probability for each rectangle, in our experiments, we use the minimum threshold for a ROI to be included in the dataset as an input hyperparameter. We experimented with both versions of UNITER: *UNITER-base* with 12 attention heads and 768 output dimensions, and *UNITER-large* with 24 attention heads and 1024 dimensions[5]. In order to mitigate class imbalance from Sub-task B, we used a weighted binary cross entropy loss using the class distribution in the training set. Other hyperparameter values were determined through grid search. Thus, the minimum confidence level for ROIs was set at 0.7, the learning rate at 1e^{-4} using an AdamW optimizer (Loshchilov and Hutter, 2017), and the maximum text length at 64 tokens. We experiment with GCN embedding sizes between 8 and 32, and selected 16 as the optimum value.

In order to further address the class imbalance between the misogyny sub-types, we experimented with oversampling from the minority classes, as an alternative to the loss re-weighting technique. Also, from our experiments, we learned that using an additional "non-misogynous" class besides the four misogyny types improved the model performance.

During training, we optimize the weighted-average F1-score (i.e., the F1-scores computed for each label, afterwards weighted by the label support) using early-stopping with a patience of 2 epochs.

[4]https://github.com/MILVLG/bottom-up-attention.pytorch

[5]https://github.com/ChenRocks/UNITER

4 Results

While considering the two proposed data preprocessing techniques, removing the websites from the textual information proved to decrease the performance of the models. Even though removing the source from the texts would provide a better generalization in a production setting, this information turned out to be a clue on misogyny content in this dataset.

Compensating for the unbalanced classes with oversampling from the minority classes proved to be less impactful than weighting the loss of the positive classes. Also, we noticed a strong tendency to overfit during our experiments. A common used mitigation technique is to augment the training data with small variations - e.g., data augmentation using similar words in the embedding space (Wang and Yang, 2015), as well as simpler replacement, swap, and deletion methods (Wei and Zou, 2019). None of the applied augmentation methods improved performance. We thus hypothesize that although these augmentations create similar meanings, innuendos get lost. For example, the phrase *"Her: Excuse me, I'm trying to put a* ***load*** *in the*

	Sub-task A			Sub-task B		
Model	**Precision**	**Recall**	**Weighted-F1**	**Precision**	**Recall**	**Weighted-F1**
UNITER-base+Sentiment$_1$	81.60%	63.95%	67.17%	68.41%	42.13%	61.34%
UNITER-base+Sentiment$_2$	83.80%	62.82%	66.16%	63.59%	44.72%	63.16%
UNITER-large+Sentiment$_1$	83.00%	63.17%	66.47%	57.67%	49.68%	64.69%
UNITER-large+Sentiment$_2$	**86.80**%	62.54%	66.13%	70.51%	45.00%	64.18%
UNITER-large+VGCN$_1$	78.80%	65.78%	68.59%	63.39%	48.87%	**65.89**%
UNITER-large+VGCN$_2$	78.40%	64.05%	66.78%	58.38%	47.06%	64.50%
UNITER-large+VGCN$_3$	78.40%	**68.89**%	**71.36%**	66.30%	46.54%	64.84%
UNITER-large+VGCN$_4$	77.60%	65.10%	67.70%	48.24%	**51.84**%	64.25%
UNITER-large+VGCN$_5$	86.60%	63.03%	66.74%	**70.91**%	42.76%	61.66%
UNITER-large+VGCN$_6$	79.20%	68.39%	71.12%	58.48%	50.62%	65.35%
UNITER-large+VGCN$_7$	81.60%	65.18%	68.50%	53.66%	50.18%	65.56%

Table 1: Results on our validation set for the Ensemble components.

Model	**Precision**	**Recall**	**Weighted-F1**
UNITER-base+Sentiment$_1$	72.60%	67.60%	68.86%
UNITER-base+VGCN	**86.20**%	61.66%	64.91%
UNITER-large+Sentiment$_1$	83.00%	63.16%	66.47%
UNITER-large+VGCN$_3$	78.40%	**68.89**%	**71.36**%
Ensemble	83.20%	66.88%	70.56%

Table 2: Results on the official test set for Sub-task A.

Model	**Precision**	**Recall**	**Weighted-F1**
UNITER-base+Sentiment$_2$	59.97%	46.43%	63.99%
UNITER-base+VGCN	63.99%	45.61%	63.39%
UNITER-large+Sentiment$_1$	57.67%	49.68%	64.68%
UNITER-large+VGCN$_1$	**66.30**%	46.54%	64.84%
Ensemble	63.19%	**50.65**%	**67.31%**

Table 3: Results on the official test set for Sub-task B.

Rank	**Team**	**Sub-task A Score**	**Team**	**Sub-task B Score**
1	SRC-B	83.4%	SRC-B	73.1%
2	DD-TIG	79.4%	TIB-VA	73.1%
3	beantown	77.8%	PAFC	73.1%
	UPB (our)	71.4%	UPB (our)	67.3%
	Baseline	65.0%	Baseline	62.1%

Table 4: Comparison for Sub-tasks A and B between the top 3 team results, our scores, and the competition baseline.

dishwasher Him: Same @gogomeme" would get an augmented counterpart *"Her: Excuse me, I'm trying to put a **burdened** in the dishwasher Him: Same @gogomeme"*. Changing the word "load", which in the context has a double meaning, loses the implicit misogynistic comparison of the woman to a dishwasher. Similarly, a text like *"my horny girlfriend on her period me"* augmented as *"my horny girlfriend on her deadline me"* would not improve the training.

Tables 2 and 3 are the best performances on the official competition test set for the models we trained. All modes were trained using the hyperparameters specified in the previous section.

Even though the performance on the binary classification from Sub-task A was comparable with

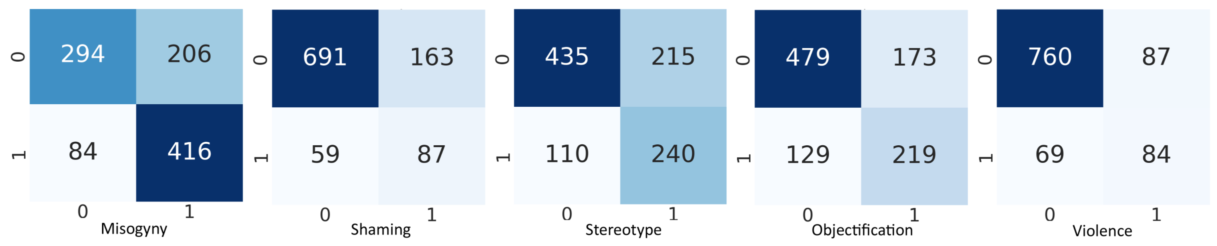


Figure 3: Per class confusion matrices for the ensemble model.

the best single model (UNITER-VGCN), it was slightly lower. As seen in Figure 3, our best system - the ensemble model - has the tendency to over-predict misogyny. Some erroneous predictions were driven by their aggressive language, for instance *"IF YOU'RE DATING MY DAUGHTER AND YOUR STUPID ENOUGH TO DO THIS I'M GOING TO KILL YOU!" depicting also that a domestic abuse victim was understandably detected as misogynous type "violence"*. Records 15566 and 15311 depict a face close up of a famous woman and were incorrectly labeled as misogynous, even if the text is replaced with a neutral or even positive one like "woman" or "best". This can be explained by the off-balance in the training data where the visual object "woman" is detected 2,262 times in the misogynous entries, and only 639 times in non-misogynous memes; similarly, "eyebrows" are four times more likely to appear in the misogynous class. However, memes like *"When you know it's a trap but you can't wait to take the bait"* that depict a woman body are not detected as misogyny since these memes require additional background knowledge in order to understand the real intent.

5 Conclusion

In this paper, we describe the architectures used in our submission at Semeval-2022, Task 5: MAMI - Multimedia Automatic Misogyny Identification. Our proposed models took the pre-trained UNITER-base and UNITER-large models and enhanced them with image sentiment features or, by using a GCN, with additional domain information from an external dataset. Our best model achieved an F1-score of 71.4 for Sub-task A and 67.3 in Sub-task B, seen in Table 4 in comparison to the top three results on each sub-task, arguing that these models can perform reasonable well in a multi-modal setting and that the generalization power of the UNITER pre-trained model was enhanced by integrating image object nodes in the co-occurrence graph. Also, we showed that our ensemble smoothed out the uneven performance caused by different train/dev data splits and improved the overall performance.

In terms of future work, we plan to continue our research into the automatic detection of abusive and hateful online content, and extend the experiments onto other pre-trained multi-modal models, as well as to attempt to improve their performance through task-adaptive pre-training (Gururangan et al., 2020). Even though Internet memes provide an interesting combination of textual message and image, they represent only one medium that can spread toxic messages. Studying other modalities like video or audio would help widen the understanding on how to detect and limit the spread of these undesired messages.

References

Muhammad Z. Ali, Ehsan-Ul-Haq, Sahar Rauf, Kashif Javed, and Sarmad Hussain. 2021. Improving hate speech detection of urdu tweets using sentiment analysis. *IEEE Access*, 9:84296–84305.

Shiladitya Bhattacharya, Siddharth Singh, Ritesh Kumar, Akanksha Bansal, Akash Bhagat, Yogesh Dawer, Bornini Lahiri, and Atul Kr. Ojha. 2020. Developing a multilingual annotated corpus of misogyny and aggression. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 158–168, Marseille, France. European Language Resources Association (ELRA).

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 30:31–40.

Angel X. Chang and Christopher Manning. 2012. SUTime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3735–3740, Istanbul, Turkey. European Language Resources Association (ELRA).

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and

Jingjing Liu. 2020. Uniter: Universal image-text representation learning. *arXiv:1909.11740 [cs]*. ArXiv: 1909.11740.

Jessica Drakett, Bridgette Rickett, Katy Day, and Kate Milnes. 2018. Old jokes, new media – online sexism and constructions of gender in internet memes. *Feminism & Psychology*, 28(1):109–127.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the evalita 2018 task on automatic misogyny identification (ami). *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:59.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. Ami@ evalita2020: Automatic misogyny identification. In *EVALITA*.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. *Ibereval@ sepln*, 2150:214–228.

Francesca Gasparini, Giulia Rizzi, Aurora Saibene, and Elisabetta Fersini. 2021. Benchmark dataset of memes with text transcriptions for automatic detection of multi-modal misogynistic content. *arXiv:2106.08409 [cs]*. ArXiv: 2106.08409.

Njagi Dennis Gitari, Zuping Zhang, Hanyurwimfura Damien, and Jun Long. 2015. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(4):215–230.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913.

Danna Gurari, Yinan Zhao, Meng Zhang, and Nilavra Bhattacharya. 2020. Captioning images taken by people who are blind. In *European Conference on Computer Vision*, pages 417–434. Springer.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2020. Evaluating aggression identification in social media. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 1–5, Marseille, France. European Language Resources Association (ELRA).

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. *arXiv:2004.06165 [cs]*. ArXiv: 2004.06165.

Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, and Helen Yannakoudakis. 2020. A multimodal framework for the detection of hateful memes. *arXiv:2012.12871 [cs]*. ArXiv: 2012.12871.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Zhibin Lu, Pan Du, and Jian-Yun Nie. 2020. Vgcn-bert: Augmenting bert with graph embedding for text classification. *arXiv:2004.05707 [cs, stat]*. ArXiv: 2004.05707.

Errol Mamani-Condori and José Ochoa-Luna. 2021. Aggressive language detection using vgcn-bert for spanish texts. In *Brazilian Conference on Intelligent Systems*, pages 359–373. Springer.

Martina Miliani, Giulia Giorgi, Ilir Rama, Guido Anselmi, and Gianluca E. Lebani. 2020. *DANKMEMES @ EVALITA 2020: The Memeing of Life: Memes, Multimodality and Politics*, page 275–283. Accademia University Press.

Ryan M. Milner. 2013. Fcj-156 hacking the social: Internet memes, identity antagonism, and the logic of lulz. *The Fibreculture Journal*, pages 61–91.

Andrei Paraschiv, George-Eduard Zaharia, Dumitru-Clementin Cercel, and Mihai Dascalu. 2021. Graph convolutional networks applied to fakenews: Corona virus and 5g conspiracy. *University Politehnica of Bucharest Scientific Bulletin Series C-Electrical Engineering and Computer Science*, pages 71–82.

Georgios Paraskevopoulos, Srinivas Parthasarathy, Aparna Khare, and Shiva Sundaram. 2020. Multiresolution and multimodal speech recognition with transformers. *arXiv preprint arXiv:2004.14840*.

Whitney Phillips. 2012. This is why we can't have nice things: The origins, evolution and cultural embeddedness of online trolling. Accepted: 2012-12-07T23:12:32Z.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv:1506.01497 [cs]*. ArXiv: 1506.01497.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Lucia Vadicamo, Fabio Carrara, Andrea Cimino, Stefano Cresci, Felice Dell'Orletta, Fabrizio Falchi, and Maurizio Tesconi. 2017. Cross-media learning for image sentiment analysis in the wild. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, page 308–317. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

George-Alexandru Vlad, George-Eduard Zaharia, Dumitru-Clementin Cercel, and Mihai Dascalu. 2020. Upb@ dankmemes: Italian memes analysis-employing visual models and graph convolutional networks for meme identification and hate speech detection. *EVALITA Evaluation of NLP and Speech Tools for Italian-December 17th, 2020*, page 288.

Liwei Wang, Yin Li, and Svetlana Lazebnik. 2016. Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5005–5013.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

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

RubCSG at SemEval-2022 Task 5: Ensemble learning for identifying misogynous MEMEs

Wentao Yu, Benedikt Boenninghoff, Jonas Roehrig, Dorothea Kolossa
Institute of Communication Acoustics, Ruhr University Bochum, Germany
{wentao.yu, benedikt.boenninghoff,
jonas.roehrig, dorothea.kolossa}@rub.de

Abstract

This work presents an ensemble system based on various uni-modal and bi-modal model architectures developed for the SemEval 2022 Task 5: MAMI-Multimedia Automatic Misogyny Identification. The challenge organizers provide an English meme dataset to develop and train systems for identifying and classifying misogynous memes. More precisely, the competition is separated into two sub-tasks: sub-task A asks for a binary decision as to whether a meme expresses misogyny, while sub-task B is to classify misogynous memes into the potentially overlapping sub-categories of stereotype, shaming, objectification, and violence. For our submission, we implement a new model fusion network and employ an ensemble learning approach for better performance. With this structure, we achieve a 0.755 macro-average F1-score (11th) in sub-task A and a 0.709 weighted-average F1-score (10th) in sub-task B. [1]

1 Introduction

Hate speech against women remains rampant despite many efforts at education, prevention and blocking. Misogyny takes place online and offline. Especially on social media platforms, misogyny appears in different forms and has serious implications (Chetty and Alathur, 2018). Currently, automated detection and filtering seem to be the most effective way to prevent hate speech online. However, over the past few years, the rising popularity of memes brought misogyny to a new multi-modal form, which may be more likely to go viral due to their often surprising combinations of text and image that may strike viewers as funny and hence, as eminently shareable.

The multi-modality of memes also makes automatic detection more challenging. Some memes express their hatred implicitly or through juxtaposition, so they may even appear harmless when considering the text or the image in isolation. SemEval-5 2022 Multimedia Automatic Misogyny Identification (MAMI) (Fersini et al., 2022) aims to identify and classify English misogynous memes.

In recent years, the Transformer model (Vaswani et al., 2017) has been widely used in natural language processing (NLP) and image processing. Transfer learning (Torrey and Shavlik, 2010) with a pre-trained Transformer model can save training resources and increase efficiency with less training data (Wang et al., 2020).

Therefore, in this work, we consider transfer learning to customize three uni-modal models based on the Transformer model: i) fine-tuning a pre-trained RoBERTa model for classification (BERTC) (Liu et al., 2019); ii) training a graph convolutional attention network (GCAN) using the pre-trained RoBERTa model for word embedding; iii) fine-tuning a pre-trained image model, the Vision Transformer (ViT) (Dosovitskiy et al., 2020). Based on these three uni-modal models, four bi-modal models are trained through our proposed model fusion network, namely BERTC-ViT, GCAN-ViT, BERTC-GCAN, and BERTC-GCAN-ViT. All models are evaluated with 10-fold cross-validation. The macro-average and weighted-average F1-scores are employed as the metrics for the sub-tasks. Ultimately, the ensemble strategy is applied on both the dataset- and the model-level (detailed in Section 3.3) for better performance.

The remainder of the paper is structured as follows: Section 2 introduces the MAMI challenge and related solutions to the task. Our ensemble model is described in Section 3, followed by the experimental setup in Section 4. Finally, our results are shown and conclusions are drawn in Sections 5 and 6.

[1]Code available at: https://github.com/rub-ksv/SemEval-Task5-MAMI.

2 Background

The MAMI dataset contains 10,000 memes as the training and 1,000 memes as the test set; all of these are given together with the text transcription as obtained through optical character recognition (OCR). The reference labels are obtained by manual annotation via a crowdsourcing platform.

The challenge is composed of two sub-tasks: Sub-task A represents a binary classification task and focuses on the identification of misogynous memes, so each meme should be classified as not misogynous (noMis) or misogynous (Mis). Sub-task B, in contrast, presents a multi-label classification task, where the misogynous memes should be grouped further, into four potentially overlapping categories. The dataset class distribution is illustrated in Table 1.

Table 1: MAMI-22 dataset class distribution. **Mis**: misogynous; **Shm**: shaming; **Ste**: stereotype; **Obj**: objectification; **Vio**: violence.

Sets	**Mis**	**Shm**	**Ste**	**Obj**	**Vio**
training set	5000	1274	2810	2202	953
test set	500	146	350	348	153

Since the provided dataset contains two modalities (namely, images and texts), an automated approach requires integrating the information from the images with the textual information. However, the OCR-based transcriptions are quite error prone, while the images are often hard to recognize for automatic systems, due, among other reasons, to overlaid text and to the popularity of further changes, such as the composition of multiple sub-images. Consequently, it is challenging to identify the pertinent information of the respective modalities, in order to merge it into a joint classification decision.

Some researchers have already worked on meme datasets. For example, (Sabat et al., 2019) created a hateful memes database, using the BERT model to extract a contextual text representation and the VGG-16 convolutional neural network (Simonyan and Zisserman, 2014) for image features. Then, text and image representations are concatenated to obtain a multi-modal representation. Facebook also organized a challenge for the identification of hateful memes in 2020 (Kiela et al., 2020). The winner of this challenge adopted an ensemble system with four different visual-linguistic transformer architectures (Zhu, 2020).

The Transformer model has shown excellent performance in many tasks, and it also shows promising results in the above studies, based on its use of the attention mechanism to extract the contextual information within a text. However, its ability to capture global information about the vocabulary of a language remains limited (Lu et al., 2020), and we hypothesize that this is even more of an issue in the task at hand, due to the very short texts in the given challenge.

For this reason, we combine a Transformer model with a graph convolutional network (GCN) (Yao et al., 2019), which may help to address this issue. GCNs can be understood as a generalization of CNNs, where the data has graph structure and locality is defined by the connectivity of the graph. As input, a GCN receives features that connect to a set of nodes. From layer to layer, the features of a node are updated as weighted combinations of its neighbors' features. In our case, the graph is defined as follows: There is a node for every word in the vocabulary and for every document. The collection of nodes is $V = \{D_1, D_2 \cdots D_{n_D}, W_1, W_2, \cdots W_{n_w}\}$, where D_i and W_i indicate the document and word nodes, respectively. n_D is the number of documents and n_W is the number of unique words in the corpus. The edges between word nodes are weighted with the word co-occurrence, the edges between document-word pairs are weighted with the term frequency-inverse document frequency (TF-IDF).

A fixed-size sliding window with step size 1 is used to gather the word co-occurrence information through the entire dataset. The point-wise mutual information (PMI) is employed to measure the relationship between the words i and j as follows:

$$\begin{aligned} \text{PMI}(i,j) &= \log\tfrac{p(i,j)}{p(i)p(j)}, \\ p(i,j) &= \tfrac{N(i,j)}{N}, \\ p(i) &= \tfrac{N(i)}{N}, \end{aligned} \tag{1}$$

where $N(i)$ counts the sliding windows in the training set that contain word i, $N(i,j)$ is the number of sliding windows that carry both words i and j, and N is the total number of sliding windows in the corpus. As described in (Yao et al., 2019), a positive PMI value indicates a high semantic correlation of words in corpus and vice versa.

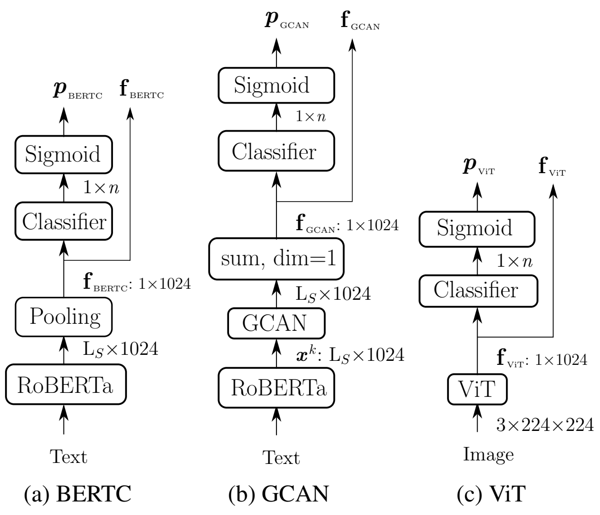
The adjacency matrix $\boldsymbol{A}$ of the graph is then computed elementwise, as follows:

$$A_{i,j} = \begin{cases} \mathrm{PMI}(i,j) & i,j \text{ are word nodes, } \mathrm{PMI}(i,j) > 0; \\ & n_D < i,j \leqslant n_D + n_W \\ \text{TF-IDF}_{i,j} & \text{document node } i \text{ and word node } j; \\ & i \leqslant n_D; n_D < j \leqslant n_D + n_W \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Since the graph is undirected, the adjacency matrix is symmetric. Finally, the adjacency matrix is normalized by $\widetilde{\boldsymbol{A}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the degree matrix of $\boldsymbol{A}$. The normalized adjacency matrix $\widetilde{\boldsymbol{A}}$ is used to weight the graph node features, cf. Section 3.1. A PyTorch implementation based on Text-GCN (Yao et al., 2019), as provided on GitHub[2], was used for the implementation.

3 System Overview

In this section, we specify our uni- and bi-modal models.



$\boldsymbol{p}_{\text{BERTC}}$
$\mathbf{f}_{\text{BERTC}}$
Sigmoid
$1\times n$
Classifier
$\mathbf{f}_{\text{BERTC}}$: 1×1024
Pooling
$\mathrm{L}_S\times1024$
RoBERTa
Text
(a) BERTC
$\boldsymbol{p}_{\text{GCAN}}$
$\mathbf{f}_{\text{GCAN}}$
Sigmoid
$1\times n$
Classifier
$\mathbf{f}_{\text{GCAN}}$: 1×1024
sum, dim=1
$\mathrm{L}_S\times1024$
GCAN
$\boldsymbol{x}^k$: $\mathrm{L}_S\times1024$
RoBERTa
Text
(b) GCAN
$\boldsymbol{p}_{\text{ViT}}$
$\mathbf{f}_{\text{ViT}}$
Sigmoid
$1\times n$
Classifier
$\mathbf{f}_{\text{ViT}}$: 1×1024
ViT
3×224×224
Image
(c) ViT

Figure 1: Uni-modal models, where L_S is the sequence length, which depends on the RoBERTa tokenizer.

Figure 1 depicts the three uni-modal models BERTC (1a), GCAN (1b), and ViT (1c), which form the basis of our further experiments. The bi-modal models are constructed based on trained uni-modal models and our proposed model fusion network, which is further detailed in Section 3.2. Finally, we apply soft and hard voting ensembles on the trained candidate models.

3.1 Uni-modal models

As illustrated in Figure 1, every uni-modal model has two outputs: the classification probabilities $\boldsymbol{p}_{\text{i}}$ and the classification features $\mathbf{f}_{\text{i}}$. All classifier blocks in our models have the following, identical structure: a fully connected layer reduces the feature dimension to half the input dimension, followed by a ReLU activation and a dropout layer. Ultimately, an output layer projects the features to the output dimension n, and a sigmoid function squashes the range of the output vector components to $(0, 1)$, allowing for an interpretation as a vector of label probabilities, with possible overlap in categories.

BERTC: We fine-tune a pre-trained large RoBERTa language model (`roberta-large`) for classification. The text input is encoded by the RoBERTa model with the embedding dimension 1024. The Pooler layer returns the first classification token `[cls]` embedding $\mathbf{f}_{\text{BERTC}}$ and feeds it into the classifier to obtain the probabilities $\boldsymbol{p}_{\text{BERTC}}$.

GCAN: Again, a pre-trained RoBERTa model extracts contextual text information. Each token is considered as a word node and each meme is a document node. Thus the word node representation is given by the corresponding RoBERTa word embedding vector. We denote the input embedding sequence of document k as $\boldsymbol{x}^k = [\boldsymbol{x}_1^k, \boldsymbol{x}_2^k, \cdots \boldsymbol{x}_{\mathrm{L}_S}^k]$, where $\boldsymbol{x}_i^k$, $i \in \{1, \ldots, \mathrm{L}_S\}$ is a 1024-dimensional embedding vector of the i-th token. As depicted in Figure 1b, $\boldsymbol{x}^k$ is an $\mathrm{L}_S\times1024$ matrix. The first classification token `[cls]` embedding represents the document classification information. Thus, we use the document-word co-occurrence information TF-IDF as the edge weights for the `[cls]` embedding. All other token embeddings are weighted with the word co-occurrence information PMI.

For each document k, we extract its specific adjacency matrix $\widetilde{\boldsymbol{A}}_k$ from the complete adjacency matrix $\widetilde{\boldsymbol{A}}$ by reducing it to rows and columns of all the document and word nodes (i and j in Equation 2) that are present in this document. The extracted document adjacency matrix $\widetilde{\boldsymbol{A}}_k$ is an $\mathrm{L}_S \times \mathrm{L}_S$ matrix.

The GCAN block in Figure 1b adopts the multi-head self-attention mechanism in 3 successive GCAN layers to embed the node representations. The queries $\mathbf{Q}$, keys $\mathbf{K}$ and values $\mathbf{V}$ are identical and set to the respective layer input. The first layer input is given by the RoBERTa word embeddings $\boldsymbol{x}^k$ of the input text. The attention of head j is obtained by

[2]https://github.com/codeKgu/Text-GCN

$$\boldsymbol{\alpha}_j = \text{softmax}\left(\frac{\left(\mathbf{W}_j^Q\mathbf{Q}^T\right)^T\left(\mathbf{W}_j^K\mathbf{K}^T\right)}{\sqrt{d_k}}\right)\left(\mathbf{W}_j^V\mathbf{V}^T\right)^T \tag{3}$$

where $\mathbf{W}_j^*$ are learned parameters for the queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$, respectively. A superscript T denotes the transpose; $d_k = \frac{d_{att}}{h}$, d_{att} is the attention dimension and h is the number of attention heads. Having computed the multi-head self-attention, each attention head output is multiplied by the document adjacency matrix $\widetilde{\boldsymbol{A}}_k$

$$\widetilde{\boldsymbol{\alpha}}_j = \widetilde{\boldsymbol{A}}_k\boldsymbol{\alpha}_j. \tag{4}$$

Equation 5 describes the output $\boldsymbol{\alpha}$ of the GCAN layer: The weighted outcomes all heads are concatenated (concat), and a fully connected layer (FC) projects the representation to the attention dimension. Inspired by (Veličković et al., 2017), instead of concatenating the weighted attention head outputs, we employ averaging (avg) to fuse these weighted outputs in the last GCAN layer. A fully connected layer again projects the final representation to the attention dimension. Thus, after the GCAN block, the text representation is still an $\text{L}_S \times 1024$ matrix. The document classification feature vector $\mathbf{f}_{\text{GCAN}}$ is obtained by summing all node representations.

$$\boldsymbol{\alpha} = \begin{cases} \text{FC}\left(\text{concat}\left(\widetilde{\boldsymbol{\alpha}}_1, \cdots \widetilde{\boldsymbol{\alpha}}_h\right)\right) & \text{not in last layer} \\ \text{FC}\left(\text{avg}\left(\widetilde{\boldsymbol{\alpha}}_1, \cdots \widetilde{\boldsymbol{\alpha}}_h\right)\right) & \text{in last layer} \end{cases} \tag{5}$$

ViT: To extract the visual contextual information, we utilize the pre-trained ViT model `vit-large-patch16-224` to encode the input image. For this purpose, the input image is split into fixed-size patches, and a linear projection of the flattened patches is used to obtain the patch embedding vectors. The Transformer encoder transforms the embedding vectors. Finally, the embedding $\mathbf{f}_{\text{ViT}}$ of the first classification token, `[cls]`, is fed to the classifier to obtain the prediction probabilities $\boldsymbol{p}_{\text{ViT}}$.

3.2 Bi-modal models

Figure 2 shows our fusion model structure.

Each model M_i has two outputs: the vector of its classification probabilities $\boldsymbol{p}_i$ and the classification features $\mathbf{f}_i$. We concatenate the model classification probabilities and features as a multi-modal representation to make the final decision.

Two fusion strategies—stream-weighting-based decision fusion and representation fusion—are considered. The weight predictor and the classifier in

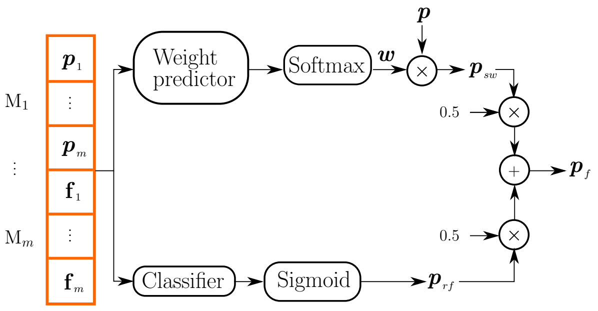


Figure 2: Fusion model structure

Figure 2 both have the same structure as the classifier block in Figure 1. The weight predictor output dimension is the number m of candidate models for fusion. The stream weighting probability $\boldsymbol{p}_{sw}$ is obtained through a weighted combination of the class probability vectors of all uni-modal model outcome probabilities, i.e.

$$\boldsymbol{p}_{sw} = \sum_i \boldsymbol{p}_i \cdot w_i. \tag{6}$$

The classifier output dimension is the same as the number of classes n. A sigmoid function computes the representation fusion probabilities $\boldsymbol{p}_{rf}$ from the combined multi-modal representation. Finally, we average the stream weighting and the representation fusion probabilities. The following model combinations are attempted, where M_i, $i \in \{1, 2, 3\}$ is the i-th pre-trained uni-modal model.

Bi-modal model	M_1	M_2	M_3
BERTC-ViT	BERTC	ViT	-
GCAN-ViT	GCAN	ViT	-
BERTC-GCAN	BERTC	GCAN	-
BERTC-GCAN-ViT	BERTC	GCAN	ViT

3.3 Ensemble learning

Having established a number of possible uni-modal and bi-modal models, we now combine these trained models into ensembles. It has been reported in many studies that ensemble learning can enhance performance in comparison to single learners (Onan et al., 2016; Zhu, 2020; Gomes et al., 2017). Therefore, we consider soft and hard voting ensemble approaches.

We use the Python `sklearn` package[3] for 10-fold cross-validation. Thus, each model structure

[3]`https://github.com/scikit-learn/scikit-learn`

was trained ten times with different inner test sets. Finally, these ten models are used to evaluate the official test set and deliver ten predictions for every sample. The soft voting ensemble method is implemented as follows: $\boldsymbol{p}_{\mathrm{M}_i}$, the ensemble probabilities that are used in the overall class decisions, are computed via

$$\boldsymbol{p}_{\mathrm{M}_i} = \sum_{j=0}^{9} w^j_{\mathrm{M}_i} \cdot \boldsymbol{p}^j_{\mathrm{M}_i}. \tag{7}$$

Here, $\boldsymbol{p}^j_{\mathrm{M}_i}$ denotes the probabilities of model M_i in the j-th fold. The weights $w^j_{\mathrm{M}_i}$ are computed by

$$w^j_{\mathrm{M}_i} = \frac{\mathrm{F1}^j_{\mathrm{M}_i}}{\sum_f \mathrm{F1}^f_{\mathrm{M}_i}}. \tag{8}$$

$\mathrm{F1}^j_{\mathrm{M}_i}$ corresponds with the best F1-score of model M_i over all epochs, computed on the inner test set in fold j. This soft voting ensemble, using the same model structure, but with the multiple outcomes from 10-fold cross-validation, is referred to as a *dataset-level ensemble* in the following.

The second type of ensemble—the *model-level ensemble*—is constructed from the *dataset-level ensemble* results of each model. We use a hard voting strategy with seven candidate models (BERTC, GCAN, ViT, BERTC-ViT, GCAN-ViT, BERTC-GCAN, and BERTC-GCAN-ViT). In this approach, we set the final prediction for a data point to one, if at least half of the considered models vote one, making it a simple majority-voting strategy.

4 Experimental Setup

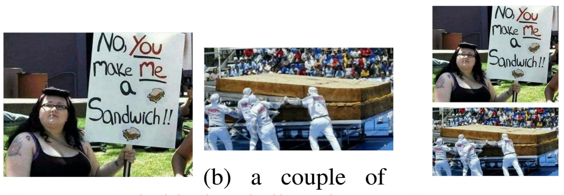
In the following, we describe our data processing and training pipeline in more detail.

4.1 Data pre-processing

The challenge dataset provides a transcription text stream that was obtained via OCR. Via image captioning, we derive a second text stream that contains a description of the image in a few words.

For the OCR text, we first use the Python `ftfy` package[4] to fix the garbled sequences that result from unexpected encodings (the *mojibake*) like "à¶´à¶§à·". Next, all "@", "#" symbols and website addresses are removed from the text. The emojis are converted to text form by the Python `emoji` package[5]. Finally, we remove non-English characters and convert the text to lowercase.

For image captioning, we utilize a pre-trained encoder-decoder attention model (Xu et al., 2016)[6]. Although the translation from image to text is not very accurate, most likely owing to issues like the overlaid meme text, it was nonetheless beneficial for our classification task. We found that the description becomes more precise, when we split the memes into their constituent sub-images where applicable. In that case, the image caption is extracted over every sub-image as well as the entire meme. Finally, the image captions are combined with the word "and" and then concatenated with the OCR text, separated by ". ". With this rule, the final text of the meme in Figure 3 is: "*mgo ci aindo make make me sandwich!!. a couple of baseball players standing next to each other and a woman holding a sign in front of a sign and a woman standing next to a group of people.*"



(a) a woman holding a sign in front of a sign

(b) a couple of baseball players standing next to each other

(c) a woman standing next to a group of people

Figure 3: In (a) and (b), we see "sub-images" and corresponding captions. (c) shows the meme and its caption (when not considering the sub-image structure).

We use the entire meme as the image input for ViT. All memes are first resized to 256×256 and center-cropped to 224×224 dimensions. The ViT model uses all 3 RGB channels, so we retain the RGB structure, thus the input image dimension is 3×224×224. We regularize the entire image database to range 0 to 1, then normalize each individual image to have zero mean and unit variance.

4.2 Loss function

We decided to use the binary cross-entropy (BCE) loss for both subtasks.

Due to the imbalance in the class distributions (see Table 1), in sub-task B, we weighted the class-

[4]https://github.com/rspeer/python-ftfy

[5]https://github.com/carpedm20/emoji

[6]https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning

specific loss terms by their support as follows:

$$w_c = \frac{\frac{\text{NoS}}{\text{NoS}(c)}}{\sum_{c'} \frac{\text{NoS}}{\text{NoS}(c')}}, c \in [\text{Shm}, \text{Ste}, \text{Obj}, \text{Vio}] \quad (9)$$

where NoS is the total number of samples in the training set and NoS(c) represents the number of true instances for class c. The loss is then computed through the weighted combination of the single BCE terms:

$$\mathcal{L}_1 = \sum_c w_c \cdot \text{BCE}(\boldsymbol{p}_c^B, \boldsymbol{y}_c^B). \quad (10)$$

Here, $\boldsymbol{p}_c^B$ represents the system's output probability of class c and $\boldsymbol{y}_c^B$ is the binary ground truth for sub-task B.

Additionally, we employ a teacher forcing loss to connect both subtasks. The idea is that an instance should be identified as misogynous and possibly grouped into sub-categories simultaneously. The teacher forcing loss is defined as:

$$\mathcal{L}_2 = \|\boldsymbol{p}^A - \boldsymbol{y}^A\|, \quad (11)$$

where the system's output probability for sub-task A is determined as:

$$\boldsymbol{p}^A = \max\left(\boldsymbol{p}_{\text{Shm}}^B, \boldsymbol{p}_{\text{Ste}}^B, \boldsymbol{p}_{\text{Obj}}^B, \boldsymbol{p}_{\text{Vio}}^B\right). \quad (12)$$

The final loss is computed by

$$\mathcal{L} = 0.7 \cdot \mathcal{L}_1 + 0.3 \cdot \mathcal{L}_2. \quad (13)$$

4.3 Model training

All models are trained using the PyTorch library (Paszke et al., 2019) for 50 epochs. The AdamW optimizer (Loshchilov and Hutter, 2017) is used for backpropagation, using a linear learning rate scheduler with a warm-up to adapt the learning rate during the first four epochs in the training stage. The dropout rate is 0.5. The RoBERTa model parameters in the BERTC and the GCAN model are optimized separately.

In our GCAN model, the adjacency matrix is computed with a sliding window of length 10. An 8-head self-attention is applied over 3 GCAN layers with an attention dimension of 1024.

For all uni-modal models, the batch size is 16 and the initial learning rate is $2 \cdot 10^{-5}$. The RoBERTa and ViT block parameters in Figure 1 are also fine-tuned. The bi-modal models are trained based on the pre-trained uni-modal models. Here, we choose the batch size as 32, the initial learning rate is $5 \cdot 10^{-6}$. As the RoBERTa and ViT block parameters in Figure 1 are already updated during the uni-modal training stage, we froze these parameters in bi-modal re-training.

To avoid overfitting, we adopt early stopping to exit the training process when the computed F1-score on the inner test set does not increase over 4 epochs. Inspired by (Huang et al., 2017), we finally averaged those two epoch-wise model parameters, which had the highest validation F1-score during the training stage.

The models have the same structure for sub-tasks A and B. The only differences are that in sub-task A, the classifier output dimension n is 1, and the BCE is used as the loss function (*Setup A*), whereas in sub-task B, the classifier output dimension n equals 4 and training uses the weighted BCE with teacher forcing (Equation 13) as the loss function (*Setup B*). All models are trained using NVIDIA's Volta-based DGX-1 multi-GPU system, using 3 Tesla V100 GPUs with 32 GB memory each.

5 Results

In summary, we investigated two configurations, displayed in Table 2. *Setup A* represents the binary classification for sub-task A, resulting in an output dimension $n = 1$. *Setup B* additionally deals with the multi-label classification of sub-task B, returning an output of dimension $n = 4$. All results are evaluated on the official test set.

Setup	**Task**	**Dimension**	**Loss**
Setup A	sub-task A	$n = 1$	BCE
Setup B	sub-tasks A/B	$n = 4$	Weighted BCE & Teacher Forcing

Table 2: Summary of the considered configurations.

5.1 Results for *Setup A* (Sub-task A)

In the first stage, we trained three different uni-modal models (i.e., BERTC, GCAN, and ViT). In the second stage, we optimized the bi-modal models (i.e., BERTC-ViT, GCAN-ViT, BERTC-GCAN, and BERTC-GCAN-ViT). The evaluation results in terms of the macro-average F1-score are displayed in Figure 4 and Table 3, showing the performance in identifying misogynous memes. To assess the statistical significance of performance differences, we applied a 10-fold cross validation and computed the Mann-Whitney-U test (Mann and Whitney, 1947).

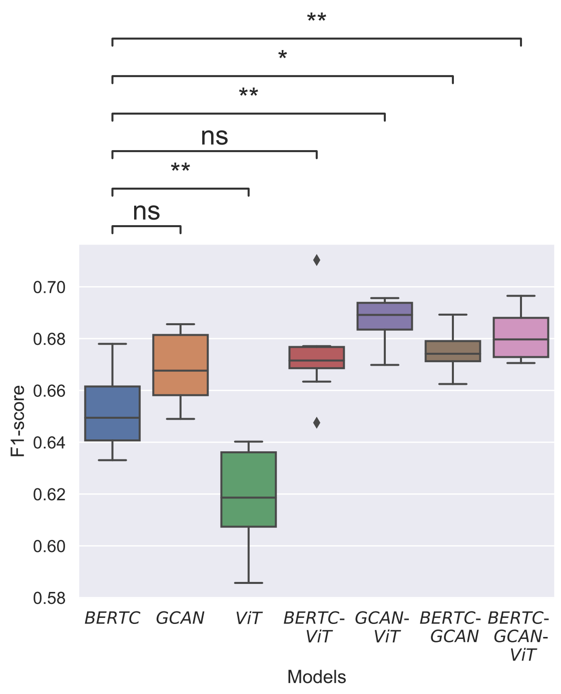


Figure 4: Macro-average F1-scores for sub-task A based on 10-fold cross validation. Asterisks indicate a statistically significant difference, where ** denotes 1e-04 < p <= 1e-03, * corresponds to 1e-02 < p <= 5e-02, and ns indicates results where p > 5e-02.

As we can see, the text-only models (BERTC and GCAN) generally show a superior performance compared to the image-only model (ViT). The results in Figure 4 clearly indicate robust performance for our bi-modal models. They are more accurate and robust. In summary, the GCAN-ViT model yields the best results w.r.t. the reported median F1-score.

Model	Ensemble	Model	Ensemble
BERTC	0.663	GCAN-ViT	**0.707**
GCAN	0.674	BERTC-GCAN	0.677
ViT	0.619	BERTC-GCAN-ViT	0.689
BERTC-ViT	0.697	-	-

Table 3: Macro-average F1-scores of soft voting ensembles for sub-task A.

Table 3 lists the averaged F1-scores for soft voting ensembles, obtained by combining all learned models from the 10-fold cross-validations. The results show that our GCAN-ViT model outperforms all other models, achieving an F1-score of 0.707.

5.2 Results for *Setup B* (Sub-tasks A/B)

Next, we addressed sub-task B, i.e. to classify the misogynous memes into four, potentially overlapping, categories. Similar to *Setup A*, we trained the same uni- and bi-modal models, but incorporating a different loss (see Table 2). For sub-task B, the weighted-average F1-score is applied. The results are presented in Figure 5.

Interestingly, the models optimized for sub-task B also perform better for sub-task A. In this case, we set the estimated label "misogynous" to 1 if at least one of the labels for "shaming", "stereotype", "objectification", or "violence" is 1.

Figure 5a depicts the sub-task A results while Figure 5b shows the corresponding performance for sub-task B. Again, we see that the bi-modal model GCAN-ViT outperforms all other models.

In addition, Tables 4 and 5 show the results for soft and hard voting ensembles. By comparing Table 4 with Table 3 (both tables represent soft voting results), we observe significantly improved F1-scores for *Setup B*.

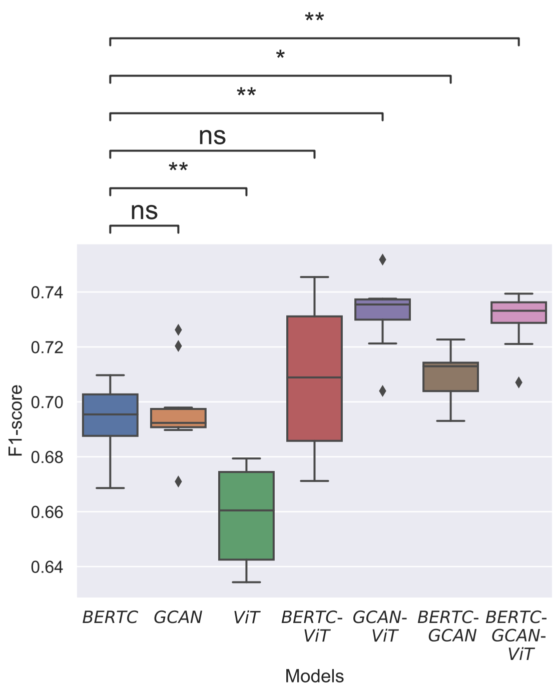
Model	Sub-task A	Sub-task B
BERTC	0.714	0.684
GCAN	0.725	0.695
ViT	0.666	0.641
BERTC-ViT	0.746	0.692
GCAN-ViT	**0.758**	**0.704**
BERTC-GCAN	0.724	0.696
BERTC-GCAN-ViT	0.755	0.704

Table 4: F1-scores of soft voting ensembles for *Setup B* (sub-tasks A and B).

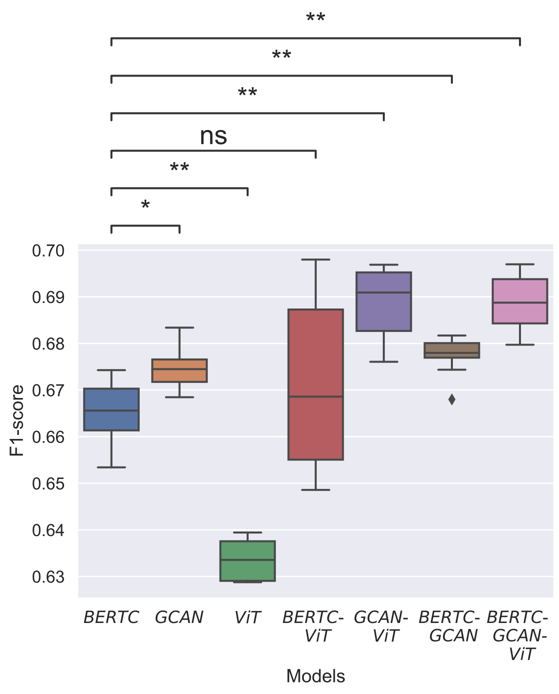
Combination	Sub-task A	Sub-task B
Three uni-modal models	0.728	0.698
Four bi-modal models	0.752	**0.709**
All seven models	**0.755**	0.706
Oracle model combination	0.762	0.716

Table 5: Model-level hard voting ensemble performance with *Setup B* for sub-task A and B.

As a last experiment, we applied hard voting on the ensembles. Again, sub-task A results are derived from sub-task B.



(a) Results for sub-task A.



(b) Results for sub-task B.

Figure 5: Performance for *Setup B*. The notation is defined in Figure 4.

Table 5 shows the results of different combinations. Generally, the combination of the four bi-modal models in the 2nd row outperforms a combination of three uni-modal models in the 1st row. If we combine all uni- and bi-modal models (3rd row), the F1-score is 0.755 for sub-task A, and 0.706 for sub-task B.

The results in bold print represent our submitted approaches for both sub-tasks, showing an F1-score of 0.755 for sub-task A and 0.709 for sub-task B.

After the challenge ended, we again evaluated all possible subset combinations of the seven candidate models. The followed combinations give the best achievable results by knowing the official test set reference labels: ViT, BERTC-GCAN-ViT, BERTC-ViT, GCAN-ViT achieves an F1-score of 0.762 for sub-task A, while an ensemble consisting of BERTC-ViT and BERTC-GCAN-ViT yields an F1-score 0.716 on sub-task B. These results are shown for comparison in the final row of Table 5 as oracle results.

6 Conclusion

This paper presents our ensemble-based approach to address two sub-tasks of the SemEval-2022 MAMI competition. The challenge aims to identify misogynous memes and classify them into—potentially overlapping—categories. We train different text models, an image model, and via our proposed fusion network, we combine these in a number of different bi-modal models.

Among the uni-modal systems, all text models show a far better performance than the image model. As expected, our proposed graph convolutional attention network (GCAN), which also considers the graph structure of the input data while using pre-trained RoBERTa word embeddings as node features, consistently outperforms the pre-trained RoBERTa model.

The proposed fusion network further improves the performance by combining the ideas of stream-weighting and representation fusion. We additionally adopt 10-fold cross-validation and use a dataset-level soft voting ensemble to obtain better and more robust results. Finally, our model-level hard voting ensemble integrates the soft voting ensemble predictions of our best uni- and bi-modal models. Our experiments indicate that this layered ensemble approach can significantly improve the model accuracy. Ultimately, our submitted system results in an F1-score of 0.755 for sub-task A and 0.709 for sub-task B.

Overall, we believe that the identification of misogyny in memes is best addressed through bi-modal recognition, considering both textual and image information. Concerning the text-based classification, we found a graph convolutional attention neural network to be beneficial as an integrative model for Transformer embeddings. This helps in the text classification, when the documents are short, as for the given meme classification task.

To cope with the bi-modality of the task at hand, we have implemented a range of systems for integrating the information from both streams. An idea that proved to be effective here was that of bringing together the strengths of early fusion and decision fusion in a joint framework. This allowed us to dynamically adjust the contributions of the two modalities through dynamic stream weighting, while still being able to combine information at the feature level across the streams, thanks to the representation fusion branch of our bi-modal systems.

Acknowledgements

The work was supported by the PhD School "SecHuman - Security for Humans in Cyberspace" by the federal state of NRW.

References

Naganna Chetty and Sreejith Alathur. 2018. Hate speech review in the context of online social networks. *Aggression and violent behavior*, 40:108–118.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Heitor Murilo Gomes, Jean Paul Barddal, Fabrício Enembreck, and Albert Bifet. 2017. A survey on ensemble learning for data stream classification. *ACM Computing Surveys (CSUR)*, 50(2):1–36.

Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. 2017. Snapshot ensembles: Train 1, get M for free. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

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

I. Loshchilov and F. Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Zhibin Lu, Pan Du, and Jian-Yun Nie. 2020. VGCN-BERT: augmenting BERT with graph embedding for text classification. *Advances in Information Retrieval*, 12035:369.

H. Mann and D. Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60.

Aytuğ Onan, Serdar Korukoğlu, and Hasan Bulut. 2016. Ensemble of keyword extraction methods and classifiers in text classification. *Expert Systems with Applications*, 57:232–247.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*.

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. *arXiv preprint arXiv:1910.02334*.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global.

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

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Ling Wang, Chengyun Zhang, Renren Bai, Jianjun Li, and Hongliang Duan. 2020. Heck reaction prediction using a transformer model based on a transfer learning strategy. *Chemical Communications*, 56(65):9368–9371.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2016. Show, attend and tell: Neural image caption generation with visual attention.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, 01, pages 7370–7377.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

RIT Boston at SemEval-2022 Task 5: Multimedia Misogyny Detection By Using Coherent Visual and Language Features from CLIP Model and Data-centric AI Principle

Lei Chen*
Rakuten Institute of Technology (RIT)
Boston, MA
lei.a.chen@rakuten.com

Houwei Chou*
RIT
houwei.chou@rakuten.com

Abstract

Detecting MEME images to be misogynous or not is an application useful on curbing online hateful information against women. In the SemEval-2022 Multimedia Automatic Misogyny Identification (MAMI) challenge, we designed a system using two simple but effective principles. First, we leverage on recently emerging Transformer models pre-trained (mostly in a self-supervised learning way) on massive data sets to obtain very effective visual (V) and language (L) features. In particular, we used the CLIP (Radford et al., 2021) model provided by OpenAI to obtain coherent V and L features and then simply used a logistic regression model to make binary predictions. Second, we emphasized more on data rather than tweaking models by following the data-centric AI principle. These principles were proven to be useful and our final macro-F1 is 0.778 for the MAMI task A and ranked the third place among participant teams.

1 Introduction

Systematic inequality and discrimination to women does not appear offline but also in online communication. MEME is an image characterized by a visual content with an overlaying text added MEME creators. Although most of MEMEs are created with the intention of making funny jokes, some of MEMEs are created as a form against women. Therefore, identifying misogynous MEMEs is important for curbing such misuse.

In the SemEval-2022, the task 5, Multimedia Automatic Misogyny Identification (MAMI) (Fersini et al., 2022), was organized for this purpose. The challenge consists of two sub-tasks, including the task A, which is determining a MEME be misogynous or not, and the task B which distinguishes non-misogynous and 4 overlapped misogyny subtypes.

*Equal contributor

We participated in the MAMI challenge and focused on the task A. Our solutions focused on exploring various pre-trained Transformer models for extracting textual and visual features and utilizing a simple logistic regression (LR) model to make binary predictions. In addition, following a new trend in AI research, which is relying on the power provided by data more, i.e., data-centric AI (Ng, 2021b,a), we expanded available training data by manually marking more samples from the evaluation set. As a result, by jointly utilizing these methods, our team ended up on obtaining the third rank in the task A.

2 Related work

Automatic Misogyny Identification (AMI) has become an active research topic in natural language processing (NLP). For example, in the IberEval-2018, the AMI task was introduced as a new task (Fersini et al., 2018). The task consists of the three sub-tasks, i.e., misogyny identification, misogynistic behavior categorization, and target classification. The misogyny related annotations are made on both Spanish and English tweets. Among 11 different teams from 5 countries, (Pamungkas et al., 2018) ranked first in the misogyny identification task on both languages. It proposed an SVM-based architecture and explored several sets of features, including lexical features relying on the lexicon of abusive words.

Another AMI challenge (Fersini et al., 2020) was organized at the Evalita-2020 evaluation campaign and used Italian tweets. Its sub-task A is about misogyny and aggressiveness identification (4 class labels). A total 8 teams from 6 different countries participated in the challenge. Though a few teams used traditional word embedding to be textual features, most of the participants explored richer sentence embedding such as BERT (Devlin et al., 2019) or Roberta. Regarding modeling, the used methods are diverse, ranging from simple logistic regression (LR), to Convolutional Neural Network (CNN), even to fine-tuning pre-trained models.

Data plays an important role on AMI research development. How to organize misogyny labels is an important research question. (Guest et al., 2021) created a new dataset for tackling online misogyny. Its dataset consists of $6,567$ labels for Reddit posts and comments. A new hierarchical taxonomy has been proposed and a careful training was provided to annotators for obtain high-quality labels.

In the misogyny detection works described above, only textual clues are utilized. In a related topic, detecting hateful speech, image clues have been widely used to better reflect the fact that human communication is naturally multimodal. A Hateful Memes Challenge competition was held at NeurIPS 2020 (Kiela et al., 2020). The task is to classify a meme (i.e., an image and associated texts) to be hateful or not. In the challenge, a set of language-visual pre-trained models, such as UNITER, VILLA, and ERNIE-ViL, have been widely used for extracting semantically coupled textual and visual features. For example, the winning solution utilized four types of VL transformers (Zhu, 2020). The multimodal hateful meme detection prompts more follow-up research. For example, (Zhou et al., 2021) proposes using a triplet-relation network to improve encoding on texts, images, and captions generated on images. The improved encoding helps final prediction performance. (Pramanick et al., 2021) propose MOMENTA (multimodal framework for detecting harmful memes and their targets) that uses both global and local perspective to detect all kind of hateful memes. In addition, this framework can be easily explainable and can generalize to unseen contexts.

3 Task and Data

The MAMI challenge consists of the two sub-tasks. The task A is a binary classification task on identifying a MEME to be misogynous (labeled as 1) or not (labeled as 0). The task B is a multi-label classification task on identifying a MEME to be non-misogynous or misogyny sub-types that can be overlapped, i.e., *shaming, stereotype, objectification, and violence*. Table 1 reports on counts of 0/1 labels on the five types of labels. Note that on the misogyny label that is the task A's prediction goal, half of MEMEs in the training data are misogynous ($label = 1$). Among four types of misogyny sub-types, we can find that their distributions are not balanced. For example, most frequent sub-type is stereotype with 2810 positive MEMEs while the least frequent label is the violence with only 953 positive MEMEs. For the task A, the evaluation metric is macro-F1. For the task B, the evaluation metric is micro-F1 among five sub-types.

4 Methods

Regarding extracting features from MEME posts' text and image parts, we chose using pre-trained Transformer models to utilize their highly effective feature representations. On texts, we considered two ways, including fine-tuning BERT (Devlin et al., 2019) model and using sentence representations based on BERT, such as Universal Sentence Encoding (USE) embedding (Cer et al., 2018) and SBERT (Reimers and Gurevych, 2019). A major difference between these two ways is that pre-trained BERT model weights are updated in the fine-tuning process. In a contrast, when using USE or SBERT embedding features, these pre-trained models are kept intact.

Regarding the visual encoder processing MEME images to visual representations, we chose a newly emerging Transformer model similar to the BERT model on texts. In recent years, Transformer based visual models have become popular (Han et al., 2020). Among the many visual Transformer models, we selected the ViT model (Dosovitskiy et al., 2020), which is a pure Transformer that is applied directly on an image's $P \times P$ patch sequence. In the implementation, it follows the original Transformer's design as much as possible. ViT utilizes the standard Transformer's encoder part as an image classification feature extractor and adds a MLP head to determine the image labels. The ViT model is pre-trained using a supervised learning task on a massive image data set. The size of the supervised training data set impacts ViT performance significantly. When using Google's in-house JFT 300M image set, ViT can reach a performance superior to other competitive ResNet (He et al., 2016) models. We used the open-sourced pre-trained models on the ImageNet 21K dataset.[1] After converting a MEME image to $P \times P$ patches, ViT converts these patches to visual tokens. After adding a special [CLS] visual token to represent the en-

[1] `https://github.com/google-research/vision_transformer`

labels	misogynous	stereotype	shaming	objectification	violence
0	5000	8726	7190	7798	9047
1	5000	1274	2810	2202	953

Table 1: Count of label 1 (positive) and 0 (negative) on the misogyny label and other 4 types of sub-types of misogyny in the training set with $n = 10,000$ MEMEs

tire image, the $M = P \times P + 1$ long sequence is fed into a ViT model to output an encoding as $\mathbf{v} = (v_0, v_1, v_2, ...v_M)$, where $M = P \times P$.

CLIP model (Radford et al., 2021) is a seminal work from OpenAI. As shown in Figure 1, on a massive set of image-text pairs, about 350 million, CLIP can pre-train a quite powerful vision-language (VL) joint model by using a simple cross-modal contrastive learning. The trained model shows many impressive applications, like superior performance on many zero-shot image classification tasks. Note that the advantage of using the CLIP model is that the extracted visual and language features have been mapped into a unified embedding space. This will facilitate the next step that combines the two types of features (V for visual features and L for language features) and then uses a simple LR model for predicting misogyny.

After obtaining visual (V) and language (L) features, one solution could be using sophisticated multimodal fusion methods as shown in (Chou et al., 2020) to consider inter-actions between the two types of features. However, after our pilot experiments, we did not find noticeable gains by using such advanced fusion methods compared to the simple *early fusion*, i.e., simply concatenating both V and L features. Therefore, we focused on utilizing the early fusion method in this challenge. Figure 2 depicts our proposed model. The image and text part of a MEME post are sent into the CLIP model to extract both V and L features. Then, the V and L features are combined and fed into a LR model to make a binary prediction on misogyny.

Besides the LR model, we also considered another novel way, DeepInsight (Sharma et al., 2019) that is suggested recently. People were impressed by Convolution Neural Network (CNN) on its universal ability on extracting useful image features. Therefore, DeepInsight was proposed for converting a tabular feature vector into a 2D image and using a CNN model to do classification. On some tasks, such method shows its effectiveness on extracting features from complicate tabular data. In

Model	macro F1
BERT fine-tuning	0.608
SBERT embedding + LR	0.650
USE embedding + LR	0.671
ViT fine-tuning	0.633
visualBERT fine-tuning	0.642
USE,ViT + LR	0.720
CLIP VL features + LR	0.765

Table 2: Macro-F1 on misogyny detection from various models that are based on using pre-trained Transformer models to extract features.

addition, a set of techniques that have shown to be useful for improving image classification performance, such as data augmentation in the training stage, i.e., mix-up (Zhang et al., 2017), or in inference stage, testing time augmentation (TTA), are already developed on the image classification task and can be easily applied.

In recent years, a new trend in AI research has emerged and it emphasizes the power brought by data sets (Ng, 2021a). On some AI tasks, the performance increase can be achieved by adding a set of labeled samples and sometimes such new data set could be small. In a contrast, performance increases could be hard to achieve when trying different models. For example, in the challenge (Ng, 2021b), all participants were required to solve the problem by only using data-related methods while keeping using one identical model. We also explored this new approach. In this challenge, we explored the data-centric AI approach by manually annotating more samples in the evaluation set. Although we don't have an access to the coding manual used in the MAMI challenge, we checked the training and trial data that were provided with manual labels. After learning the how-to, we annotated a subset of testing samples and then added these labeled samples into our model's training.

5 Results

Table 2 reports on experiment results on various models based on pre-trained Transformers.

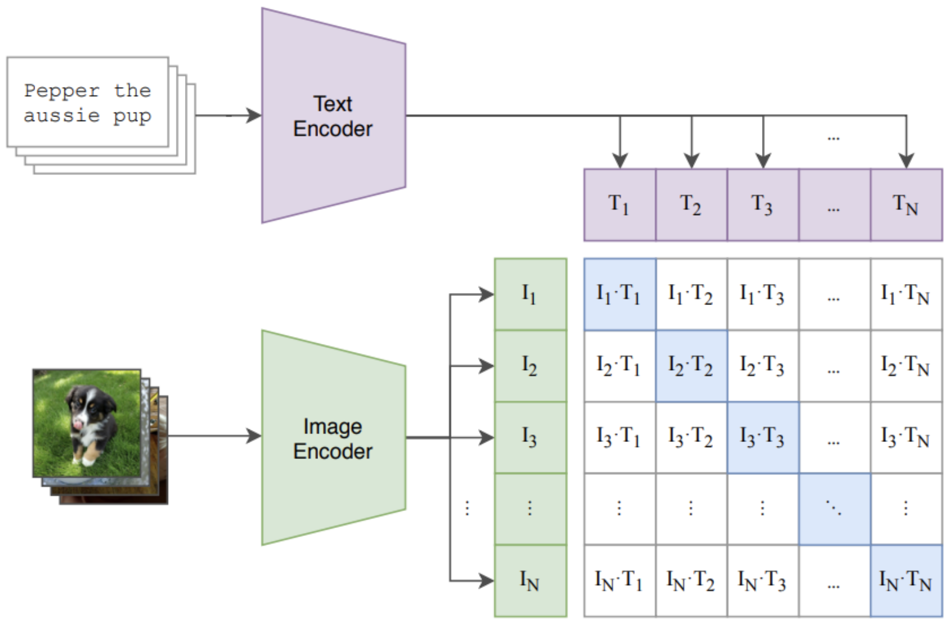

Pepper the aussie pup
Text Encoder
Image Encoder
T1 T2 T3 ... TN
I1 I2 I3 ⋮ IN
I1·T1 I1·T2 I1·T3 ... I1·TN
I2·T1 I2·T2 I2·T3 ... I2·TN
I3·T1 I3·T2 I3·T3 ... I3·TN
IN·T1 IN·T2 IN·T3 ... IN·TN

Figure 1: CLIP model is pre-trained on a large number of text-image pairs in a contrastive learning way

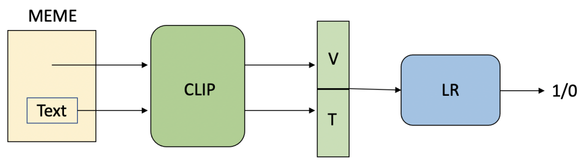


Figure 2: Our model is based on using a logistic regression to detect misogyny by using VL features extracted by the CLIP model

When using BERT fine-tuning, i.e., adding a fully-connected layer on the [CLS] token after BERT model's output layer and fine-tuning two models together by using the cross-entropy (CE) loss, the macro-F1 is 0.608. A different way is obtaining sentence level representations and then feeding these dense features into a LR model implemented in the scikit-learn Python package. We tried both SBERT and USE sentence level representations. This way of using sentence level representations in fact shows improved performance. The macro-F1 can be improved to 0.605 for using the SBERT features and 0.671 for using the USE features.

On image features, we explored ViT model fine-tuning and observed that images play an important role on the misogyny detection. The performance simply using images is 0.633, which is higher than using texts by fine-tuning a BERT model.

Regarding fusing both textual and image features for making a multimodal classification, we tried two methods. The first is fine-tuning a joint Visual-Language (VL) model, visualBERT (Li et al., 2019). However, the performance is not very impressive. Its performance is 0.642, only a slight gain on top of either of uni-modal's performance. The second method is using an *early fusion* by concatenating textual (USE embedding) and image features (ViT embedding) and then fed into a logistic regression (LR) for predicting misogyny. By doing so, the performance can be improved to 0.720.

However, the USE and ViT embeddings are learned from separate models and may not exist in a unified space. To address this issue, We tried the CLIP model for the two attractions, i.e., (1) having coherent textual and visual features in an unified space and (2) image encoder training is on a massive image set with about 350 million images. Consistent to our prediction, after switching to CLIP features, the misogyny prediction's macro-F1 value immediately increased to 0.765.

To explore other possible sophisticated models besides using an LR model, we explored the DeepInsight (Sharma et al., 2019). After converting the dense VL features output from the CLIP model into 2D images, we use a ResNet34 CNN model pre-trained on the ImageNet dataset and converted the misogyny detection into a CNN-based image classification. However, as shown in Table 3, the performance, i.e., 0.751, is worse than using an

Model	macro F1
DeepInsight + CNN	0.751
+ mixup	0.758
+ TTA	0.726

Table 3: Macro-F1 on misogyny detection by using the method converting visual and textual embeddings to 2D images and then using CNN model to do a classification.

Model	macro F1
CLIP VL features + LR	0.765
+ 50 labeled samples	0.767
+ 150 labeled samples	0.772
+ 250 labeled samples.	0.777
using semi-supervised learning	0.778

Table 4: Macro-F1 on misogyny detection by introducing more labeled samples annotated on the test set.

LR model directly. When using the mix-up augmentation, we observed a further performance gain to 0.728. Surprisingly, The TTA method did not show any help. One possible explanation is that we used dense vectors rather than regular tabular data whose feature columns represent some real physical values.

Table 4 reports on the results of utilizing the data-centric AI principle. We can find that by adding increasing number of labeled samples (from the evaluation set), we can keep increasing macro-F1 values. When using labels created by us on 25% of the evaluation set, we can reach a macro-F1 to 0.777. We also used a simple pseudo-label semi-supervised method that is provided by the sci-kit learn Python package and treated all evaluation set ($n = 1,000$) to be unlabeled data. This gave us another small gain to reach our final result of 0.778.

6 Discussions

Multimedia misogyny detection is an important natural language processing application. It uses powerful AI technologies to against misinformation or even harmful information appearing in online communication. For a world emphasizing equal roles between genders, finding misogyny information and removing them is critical for a healthy online communication platform. In this challenge, our methods have been focusing on (a) relying on various pre-trained Transformer models to provide high quality multimodal features and (b) applying the data-centric AI principle to rapidly improve model performances with controllable human efforts. Regarding text encoding, we found that running a simple LR model on top of sentence level representations works consistently better than fine-tuning BERT models. Joint VL model, e.g., visualBERT, does not work quite well in this challenge. However, CLIP, which is trained on a large-sized text-image pair data set in a self-supervised learning approach, can provide high-quality multimodal features and these features can be conveniently used in down-stream classification tasks. On top of highly effective multimodal features, utilizing sophisticated models becomes secondary. In our experiments, simply using an LR model gave us better result than using other complicate models, e.g., DeepInsight. At last, the data-centric AI principle is worth noting. By focusing on our efforts on expanding labeled training data, we can consistently improve our misogyny prediction performance.

References

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

H. Chou, Y.H. Lee, L. Chen, Y. Xia, and W.T. Chen. 2020. CBB-FE, CamemBERT and BiT Feature Extraction for Multimodal Product Classification and Retrieval. In *Proc. SIGIR'20 e-Com workshop*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, and Sylvain Gelly. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. Ami@ evalita2020: Automatic misogyny identification. In *EVALITA*.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018. Overview of the task on automatic misogyny identification at ibereval 2018. *Ibereval@ sepln*, 2150:214–228.

Ella Guest, Bertie Vidgen, Alexandros Mittos, Nishanth Sastry, Gareth Tyson, and Helen Margetts. 2021. An expert annotated dataset for the detection of online misogyny. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1336–1350, Online. Association for Computational Linguistics.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, and Yixing Xu. 2020. A Survey on Visual Transformer. *arXiv preprint arXiv:2012.12556*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Andrew Ng. 2021a. A.i. needs to get past the idea of big data.

Andrew Ng. 2021b. Data-centric ai competition.

Endang Wahyu Pamungkas, Alessandra Teresa Cignarella, Valerio Basile, Viviana Patti, et al. 2018. 14-exlab@ unito for ami at ibereval2018: Exploiting lexical knowledge for detecting misogyny in english and spanish tweets. In *3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018*, volume 2150, pages 234–241. CEUR-WS.

Shraman Pramanick, Shivam Sharma, Dimitar Dimitrov, Md Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021. Momenta: A multimodal framework for detecting harmful memes and their targets. *arXiv preprint arXiv:2109.05184*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, and Jack Clark. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Alok Sharma, Edwin Vans, Daichi Shigemizu, Keith A Boroevich, and Tatsuhiko Tsunoda. 2019. Deepinsight: A methodology to transform a non-image data to an image for convolution neural network architecture. *Scientific reports*, 9(1):1–7.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Yi Zhou, Zhenhao Chen, and Huiyuan Yang. 2021. Multimodal learning for hateful memes detection. In *2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–6. IEEE.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

TeamOtter at SemEval-2022 Task 5: Detecting Misogynistic Content in Multimodal Memes

Paridhi Maheshwari [*]
Stanford University
paridhi@stanford.edu

Sharmila Reddy Nangi [*]
Stanford University
srnangi@stanford.edu

Abstract

We describe our system for the SemEval 2022 task on detecting misogynous content in memes. This is a pressing problem and we explore various methods ranging from traditional machine learning to deep learning models such as multimodal transformers. We propose a multimodal BERT architecture that uses information from both image and text. We further incorporate common world knowledge from pretrained CLIP and Urban dictionary. We also provide qualitative analysis to support out model. Our best performing model achieves an F1 score of 0.679 on Task A (Rank 5) and 0.680 on Task B (Rank 13) of the hidden test set. Our code is available at https://github.com/paridhimaheshwari2708/MAMI.

1 Introduction

In this era of the internet, memes have become a new form of communication, which predominantly contain an image and a small caption. While their general purpose is to invoke humour or irony, they are also increasingly being used as a source of harmful, offensive and misogynistic content. Detecting such content in an automated manner is an important problem to avoid the spread of hate.

Memes pose a unique multimodal challenge as their underlying implication is not just a simple combination of the image and text, but a subtle inference that comes naturally to humans. Another complexity is that memes are highly contextual and the component image and text pieces might be completely uncorrelated. Understanding this fusion of modalities is a challenging task for machines. Our aim is to automatically identify misogynistic multimodal memes using machine learning.

2 Related Work

The task of identifying misogyny in memes is a relatively new area and is closely related to hate detection. While there has been a lot of work on identifying hateful content in unimodal data (Gandhi et al., 2019; Fortuna and Nunes, 2018), there is little work on multimodal hate detection. Recently, Facebook Hateful Memes Challenge (Kiela et al., 2020) explored fusion of text and vision models along with advanced architectures like cross-modal BERT (Lu et al., 2019). A major problem with these large pretrained models is the domain gap between memes and training data. Some works try to solve this with better pretraining (Zhu, 2020) and disentangling hate from meme representations (Lee et al., 2021). In this work, we build on these technologies for our specific use-case of misogyny detection and incorporate common world knowledge from Urban Dictionary (Wilson et al., 2020) and CLIP (Radford et al., 2021) to address the domain gap.

3 Method

3.1 Baselines

The task of detecting misogynistic content in memes can be posed as a classification task based on visual and textual features. We start with simple baselines, namely SVM, Naive Bayes and Logistic Regression, and also experiment with *unimodal* feature space, i.e, training classifiers with *text only* and *image only* features. For text only models, we incorporate the TF-IDF technique with bag-of-words concepts to compute features. To capture visual cues from images, we leverage pretrained VGG-16 (Simonyan and Zisserman, 2014) for feature extraction. Since memes are a complex combination of text and image, we require cues from both modalities and we therefore, move towards *multimodal* methods for classification.

3.2 Deep Learning Architectures

We leverage various Deep Learning (DL) techniques for this task. We first start with unimodal techniques, namely LSTM and CNN architectures.

[*] These authors contributed equally

Model	Binary Classification				Multi-class Multi-label Classification			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Using Only Text Features								
Logistic Regression	**0.802**	**0.802**	**0.801**	**0.801**	**0.844**	**0.800**	0.455	0.524
SVM	0.794	0.807	0.790	0.790	0.838	0.673	0.556	**0.596**
Naive Bayes	0.794	0.795	0.795	0.794	0.827	0.707	0.363	0.371
Using Only Image Features								
Logistic Regression	0.634	0.634	0.634	0.634	0.762	0.462	0.399	0.425
SVM	0.631	0.631	0.631	0.631	0.767	0.468	0.385	0.417
Naive Bayes	0.633	0.632	0.632	0.632	0.677	0.441	0.558	0.480
Using Both Image and Text Features								
Logistic Regression	0.760	0.760	0.759	0.634	0.808	0.583	0.516	0.545
SVM	0.786	0.786	0.786	0.786	0.818	0.593	**0.577**	0.584

Table 1: Evaluation of various baselines.

For text, we use the GloVe (Pennington et al., 2014) embeddings to initialize individual words and pass this sequence through an LSTM layer. Finally, this embedding is fed to FC layers that outputs a score for each class. For image, we extract the feature representations from a pretrained VGG-16 (Simonyan and Zisserman, 2014) model and pass through a classifier head which is composed of FC layers. All models are trained end-to-end using binary cross entropy loss for every class independently. Note that we do not pose this as softmax classification as each meme can belong to multiple classes simultaneously, i.e., multi-label classification. To handle class imbalance in the dataset, we give more importance to the positive examples. Specifically, we weigh the positive component of the binary cross entropy loss with the ratio of negative to positive occurrences per class.

Since our data is inherently multimodal, we propose advanced DL methods that incorporate both textual and visual features. This is important because memes are complex entities and the fusion of both modalities is necessary to understand the full meaning of the meme (which might not be apparent from a single modality alone). We experiment with the following mulitmodal networks:

1. **CNN + LSTM**: This architecture does a simple late-fusion of the two unimodal designs. We concatenate image and text features and pass through a FC classifier for prediction.
2. **VQA**: There has been significant work in multimodal learning on Visual Question Answering, which requires subtle reasoning around both modalities to answer complex queries. Given similar reasoning in memes, we experiment with the VQA model (Antol et al., 2015). Both image and text (question) features are transformed to a common space and fused via element-wise product, which is then passed to a FC layer to get class scores (answers).
3. **MUTAN**: This model (Ben-Younes et al., 2017) tries to effectively mix and merge information from the two modalities. It uses a multimodal Tucker decomposition to efficiently parametrize bilinear interactions between visual and textual representations. It demonstrates improved performance on the visual question-answering task by learning interpretable embedding spaces.

Recently, Bidirectional Encoder Representations from Transformers (BERT) models (Devlin et al., 2019) trained on large-corpus have proven to provide state-of-the-art results for diverse NLP applications. Given an input sentence, a pretrained BERT model gives a hidden representation for each token in the sentence along a pooled output for the entire sentence. These representations are rich in contextual knowledge and we explore different ways to use this information as follows:

4. **Concat BERT**: The pooled output for text is concatenated with the image feature, and passed through a FC classifier.
5. **Average BERT**: Similar to the previous setting, but the average of the final hidden state is taken as the text feature.
6. **Gated BERT**: The final hidden state is averaged to get text feature. We combine the text and image feature using a Multimodal Gated Layer (Ovalle et al., 2017), which weights relevance of each modality and combines them to output prediction classes.

3.3 Common World Knowledge

Language in memes is informal and often contains slang words. We propose to use the Urban dictionary which is a crowd-sourced repository of common slangs along with their definitions. Particularly, we initialize our constituent words with embeddings pretrained on the Urban Dictionary (Wilson et al., 2020) instead of GloVe vectors. These features perform well on extrinsic tasks such as sentiment analysis and sarcasm detection where some knowledge of colloquial language is required.

Popular vision algorithms (such as VGG-16) are trained on object detection tasks and they require explicit supervision from labels. This limits their usability. More recently, pretraining on image-text matching (Radford et al., 2021) has gained traction by outperforming other methods. Since the images are crawled from the internet, we believe that the distribution captured by CLIP (Radford et al., 2021) are more relevant and representative of the online media today, and hence, more suitable for our task.

3.4 Joint Learning

In the previous sections, we were considering the two tasks independently and training separate models. Given the synergy between the two tasks, we propose a joint learning framework where we use weight sharing between networks to exploit the commonalities and learn improved features. We propose two approaches to achieve this:

1. **Multi-Task Learning**: We start with a multimodal deep network as a shared embedding layer for both modalities, and followed by two different classifier heads, one for each task.
2. **Hierarchical Learning**: We utilize the inherent hierarchy between the two tasks where the second classifier kicks in only when the probability for "misogynous" class from first classifier is greater than 0.5. The model architecture is same as the multi-task setup, but now the second classifier head for finer categorization is only trained on misogynous items.

4 Experiments and Results

4.1 Task and Dataset

We work on the "*Multimedia Automatic Misogyny Identification*" task (Fersini et al., 2022) at SemEval 2022. The problem comprises of two sub-tasks: (i) *Binary Classification* to categorize a given meme as misogynous or not; (ii) *Multi-class Multi-label Classification* to further classify misogynous memes into fine-grained, overlapping categories (shaming, stereotype, objectification, violence). Our dataset consists of 10,000 memes and we partition them into 70% / 20% / 10% for train, validation and test respectively. We only report metrics on this data split as we do not have the ground truth labels for the competition's hidden test set. We measure the performance using these metrics: average accuracy per class, and weighted-average precision, recall and F1 scores where the weights are determined by the support of that class.

4.2 Textual and Visual Cues

Prior work on detecting sexism in memes (Fersini et al., 2019) use specially curated textual and visual cues. We curated the profanity scores for text using a pretrained model on toxic comment classification (Pearson coeff. -0.05), sentiment polarity from Textblob (Pearson coeff. -0.012), and percentage of skin in images (Pearson coeff. 0.125). Thus, many intuitive cues showed no correlation with misogyny, exemplifying the difficulty of our task.

4.3 Baselines

Table 1 presents the baseline results. We extend these linear models to the multi-label setting as a *one-vs-all* task, where separate classifier are trained for each class. We observe the following: (i) Textual models perform better than image only models. (ii) Performance of text + image models is similar to text only methods, implying that TF-IDF vectors are a strong indicator for meme classification.

4.4 Deep Learning Architectures

The results are tabulated in Table 2 and we make the following observations: (i) Using both image and text significantly improves performance over the unimodal variants. We further provide qualitative comparison of unimodal and multimodal methods in Figure 1, which also illustrates the complexity of the task and the subtle relations between the two modalities. (ii) Similar to the baselines, text only methods give better results than image only methods. (iii) BERT-based models show significant improvement in performance for multi-label classification task. (iv) For the multi-class, multi-label classification problem, there is a skewed distribution of positive and negative examples within a class. Hence, performance varies across different classes, as shown in Figure 2. Here, training the

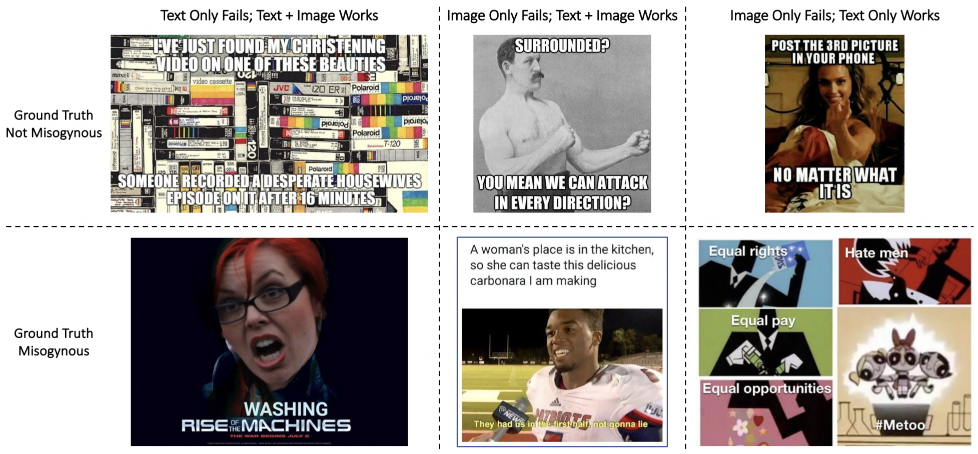

Text Only Fails; Text + Image Works
Image Only Fails; Text + Image Works
Image Only Fails; Text Only Works
Ground Truth Not Misogynous
I'VE JUST FOUND MY CHRISTENING VIDEO ON ONE OF THESE BEAUTIES
SOMEONE RECORDED A DESPERATE HOUSEWIVES EPISODE ON IT AFTER 16 MINUTES
SURROUNDED?
YOU MEAN WE CAN ATTACK IN EVERY DIRECTION?
POST THE 3RD PICTURE IN YOUR PHONE
NO MATTER WHAT IT IS
Ground Truth Misogynous
WASHING
RISE OF THE MACHINES
A woman's place is in the kitchen, so she can taste this delicious carbonara I am making
They had us in the first half, not gonna lie
Equal rights
Hate men
Equal pay
Equal opportunities
#Metoo

Figure 1: Qualitative evaluation of Text Only, Image Only and Text + Image models.

Model	**Binary Classification**				**Multi-class Multi-label Classification**			
	Accuracy	**Precision**	**Recall**	**F1**	**Accuracy**	**Precision**	**Recall**	**F1**
Using Only Text Features								
LSTM	0.772	0.773	0.770	0.770	0.654	0.504	0.712	0.558
LSTM (CWK)	0.778	0.779	0.776	0.777	0.644	0.453	0.571	0.478
Using Only Image Features								
CNN	0.717	0.727	0.712	0.711	0.673	0.497	0.629	0.524
CNN (CWK)	0.839	0.838	0.838	0.838	0.769	0.580	0.790	0.654
Using Both Image and Text Features								
CNN + LSTM	0.799	0.798	0.799	0.798	0.700	0.522	0.761	0.598
VQA	0.785	0.789	0.787	0.784	0.669	0.503	0.821	0.599
MUTAN	0.821	0.821	0.820	0.820	0.639	0.483	**0.855**	0.594
Concat BERT	0.715	0.719	0.712	0.711	0.780	0.531	0.477	0.501
Average BERT	0.742	0.746	0.740	0.740	0.796	0.563	0.541	0.540
Gated BERT	0.728	0.728	0.727	0.727	0.800	0.587	0.523	0.553
CNN + LSTM (CWK)	0.836	0.837	0.834	0.835	0.770	0.585	0.798	0.658
VQA (CWK)	0.828	0.829	0.826	0.827	0.771	0.592	0.800	0.662
MUTAN (CWK)	0.828	0.827	0.827	0.827	0.781	0.607	0.791	0.670
Concat BERT (CWK)	**0.840**	**0.841**	**0.841**	**0.840**	0.798	0.628	0.723	0.658
Average BERT (CWK)	0.837	0.837	0.836	0.836	0.838	**0.655**	0.700	0.676
Gated BERT (CWK)	0.839	0.839	0.839	0.838	**0.845**	**0.665**	0.710	**0.684**

Table 2: Evaluation of various deep learning architectures. Here, CWK refers to common world knowledge sources, namely Urban Dictionary (Wilson et al., 2020) and CLIP (Radford et al., 2021), for text and image respectively.

Model	**Binary Classification**				**Multi-class Multi-label Classification**			
	Accuracy	**Precision**	**Recall**	**F1**	**Accuracy**	**Precision**	**Recall**	**F1**
Multi-Task Learning								
CNN + LSTM (CWK)	0.832	0.833	0.830	0.831	0.750	0.598	0.810	0.662
VQA (CWK)	**0.845**	**0.845**	**0.844**	**0.844**	0.771	0.601	0.790	0.663
MUTAN (CWK)	0.836	0.835	0.835	0.835	0.769	0.585	0.803	0.658
Average BERT (CWK)	0.843	0.843	0.843	0.843	0.757	0.598	**0.816**	0.665
Concat BERT (CWK)	0.838	0.838	0.838	0.838	0.734	0.580	0.804	0.646
Gated BERT (CWK)	0.842	0.842	0.842	0.842	0.725	0.571	0.803	0.639
Hierarchical Learning								
CNN + LSTM (CWK)	0.839	0.838	0.839	0.838	0.773	0.590	0.804	0.663
VQA (CWK)	0.843	0.842	0.842	0.842	0.770	0.598	0.794	0.662
MUTAN (CWK)	0.829	0.828	0.829	0.828	0.772	0.587	0.777	0.653
Average BERT (CWK)	0.842	0.842	0.842	0.842	**0.799**	**0.613**	0.769	**0.671**
Concat BERT (CWK)	0.838	0.838	0.838	0.838	0.796	0.609	0.751	0.661
Gated BERT (CWK)	0.835	0.835	0.835	0.835	0.788	0.599	0.761	0.658

Table 3: Evaluation of various deep learning architectures using joint learning techniques.

models with weighted cross entropy plays an important role in the precision-recall trade off. Note that F1 score is a better measure than accuracy because of the class imbalance.

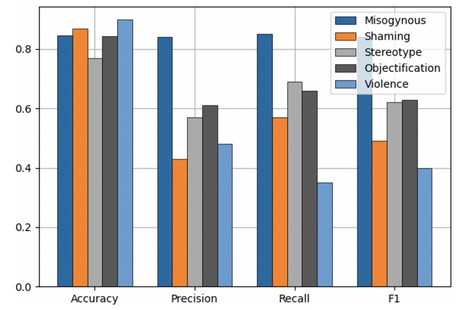


Figure 2: Class-wise performance across metrics.

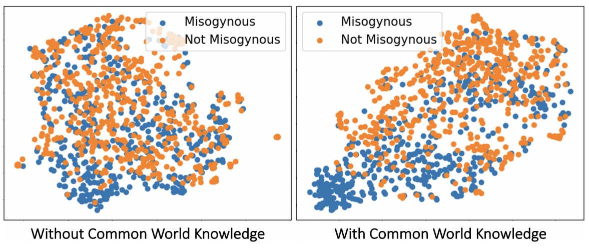


Figure 3: Multimodal features projected into two dimensional space along with misogynous-or-not labels.

4.5 Common World Knowledge

Incorporating Common World Knowledge (CWK) from Urban Dictionary and CLIP provides a substantial boost in performance across all models, and this is because visiolinguistic models are able to learn more discriminative features. To illustrate this, we consider two set of multimodal features (with and without CWK) and run dimensionality reduction using Uniform Manifold Approximation and Projection (McInnes et al., 2018). We visualize the feature space in lesser variables and plot the misogynous-or-not class in Figure 3. It can be seen that the features without CWK are not able to differentiate between the classes, whereas features with CWK result in better separation, and are therefore, more effective for our task. We provide further qualitative evidence in Figure 4.

4.6 Joint Learning

The results for multi-task and hierarchical learning are presented in Table 3. We observe that there is an improvement in the binary classification task, and we reason that the joint learning paradigm provides significantly new information about subclasses from the multi-class setting to the binary task. However, results for the multi-class setting are comparable to the independent models.

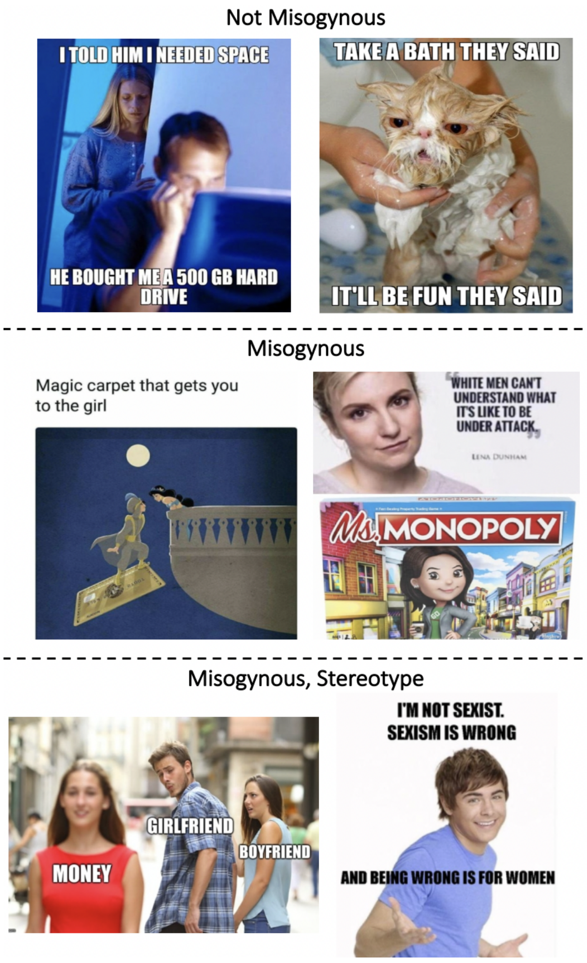

Not Misygnous
I TOLD HIM I NEEDED SPACE
HE BOUGHT ME A 500 GB HARD DRIVE
TAKE A BATH THEY SAID
IT'LL BE FUN THEY SAID
Misogynous
Magic carpet that gets you to the girl
WHITE MEN CAN'T UNDERSTAND WHAT IT'S LIKE TO BE UNDER ATTACK
LENA DUNHAM
Ms. MONOPOLY
Misogynous, Stereotype
MONEY
GIRLFRIEND
BOYFRIEND
I'M NOT SEXIST. SEXISM IS WRONG
AND BEING WRONG IS FOR WOMEN

Figure 4: Examples where Gated BERT fails, but Gated BERT with CWK classifies the memes correctly.

5 Conclusion and Future Work

Our work focused on the task of misogyny detection in multimodal memes. We demonstrated that using a combination of visual and textual, i.e, multimodal features outperforms the unimodal counterparts. In addition to simple baselines, we have also experimented with advanced DL architectures inspired from VQA and multimodal transformers. Further, we have shown how incorporating common world knowledge from Urban dictionary and pretrained CLIP can significantly help in identifying misogynistic content, along with qualitative evidence. Finally, the proposed joint learning paradigm can exploit the synergy between the two tasks, instead of training models independently.

Acknowledgements

We thank the anonymous reviewers for providing feedback on our manuscript. This work was done as part of the CS 229: Machine Learning course taught at Stanford University. We gratefully acknowledge the support and feedback from our instructors Andrew Ng, Moses Charikar and Carlos Guestrin, and the entire course staff.

References

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Hedi Ben-Younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. 2017. Mutan: Multimodal tucker fusion for visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2612–2620.

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

Elisabetta Fersini, Francesca Gasparini, and Silvia Corchs. 2019. Detecting sexist meme on the web: A study on textual and visual cues. In *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*, pages 226–231. IEEE.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30.

Shreyansh Gandhi, Samrat Kokkula, Abon Chaudhuri, Alessandro Magnani, Theban Stanley, Behzad Ahmadi, Venkatesh Kandaswamy, Omer Ovenc, and Shie Mannor. 2019. Image matters: Detecting offensive and non-compliant content/logo in product images. *arXiv preprint arXiv:1905.02234*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Roy Ka-Wei Lee, Rui Cao, Ziqing Fan, Jing Jiang, and Wen-Haw Chong. 2021. Disentangling hate in online memes. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5138–5147.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

John Edison Arevalo Ovalle, Thamar Solorio, Manuel Montes-y Gómez, and Fabio A González. 2017. Gated multimodal units for information fusion. In *ICLR (Workshop)*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Steven Wilson, Walid Magdy, Barbara McGillivray, Kiran Garimella, and Gareth Tyson. 2020. Urban dictionary embeddings for slang nlp applications. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4764–4773.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

taochen at SemEval-2022 Task 5: Multimodal Multitask Learning and Ensemble Learning

Chen Tao[1,2] **and Jung-jae Kim**[1*]
[1]Institute for Infocomm Research, A*STAR, Singapore
[2]Nanyang Techonological University, Singapore
taoc0002@e.ntu.edu.sg, jjkim@i2r.a-star.edu.sg

Abstract

We present a multi-modal deep learning system for the Multimedia Automatic Misogyny Identification (MAMI) challenge, a SemEval task of identifying and classifying misogynistic messages in online memes. We adapt multi-task learning for the multimodal subtasks of the MAMI challenge to transfer knowledge among the correlated subtasks. We also leverage on ensemble learning for synergistic integration of models individually trained for the subtasks. We finally discuss about errors of the system to provide useful insights for future work.

1 Introduction

Multimodal machine learning processes data from different modalities (e.g. visual, auditory, lingual) to infer their combined meaning. This topic has seen tremendous development, encompassing visual question answering (VQA) (Stanislaw Antol, 2015), image captioning (Chen et al., 2015), multimodal classification and beyond. SemEval 2022 Task 5 "Multimedia Automatic Misogyny Identification" (MAMI) (Fersini et al., 2022) also requires multimodal machine learning to analyze both visual and textual information from memes (image, caption) in order to identify and classify misogynistic memes. The MAMI challenge has the following two subtasks:

- Subtask A: Classification of a meme as either misogynistic or not
- Subtask B: Categorisation of the type of misogyny if the meme is identified as misogynistic in Subtask A. There are 4 types (or sub-categories) of misogyny: shaming, stereotype, objectification and violence

As introduced by a multimodal research survey (Baltrušaitis et al., 2019), data representation, fusion, and co-learning are the primary challenges in multimodal classification problems. To address those challenges, we explore the following relevant topics: data augmentation, multimodal pre-training, multi-task learning and ensemble learning. Data augmentation helps enrich under-represented features in data. Multimodal pre-trained models may represent multimodal data like those of the MAMI challenge better than unimodal pre-trained models. We also adapt multi-task learning to transfer knowledge in training data among related tasks. As the misogyny sub-categories are correlated to each other, this approach is especially useful in Subtask B. Our final predictions are determined by majority voting among the top performing models.

This paper is structured as follows: In Section 2, we describe the task description and dataset. Section 3 explains in details the models and methods we incorporate into the final system architecture. Section 4 discusses the evaluation results of the models and methods, including multi-task learning and ensemble learning. We conduct error analysis in Section 5 and conclude our findings in Section 6.

2 Task Description and Dataset

SemEval 2022 Task 5 "Multimedia Automatic Misogyny Identification" (MAMI) (Fersini et al., 2022) is a classification task that aims at identifying and classifying misogynistic memes in social media. While most memes are funny and harmless, some deliver misogynistic content and have strong, negative influence due to their high speed of spread. Such memes need to be detected and removed from online sites to avoid gender-related hate. In particular, the MAMI task targets memes, each of which is essentially an image characterized by a pictorial content with an overlaying text a posteriori introduced by human, thus a multi-modal (image+text) analytics task.

The MAMI task has two subtasks. In Subtask A, participants must classify memes into misogynistic and non-misogynistic. The evaluation metric is

[*]Corresponding Author

macro-average F1 score. Subtask B is to identify the type of misogyny from the possibly overlapping categories: shaming, stereotype, objectification, and violence if the meme is misogynistic. The evaluation metric is weighted-average F1-Measure, where the F1 scores of the four categories are averaged with weights by support, i.e. the number of true instances for each label.

The training dataset contains 10,000 memes with English captions, assembled from social media platforms. Another 1,000 memes are given as the test dataset. Besides memes in JPEG file format, the task also provides transcriptions of memes captions in a separate text file. Parts of the data are collected and evaluated as described in (Gasparini et al., 2021).

3 Methods

We present a system that utilizes a multi-task learning method to deal with dependencies between the subtasks, fills in the lacking parts of the training dataset by using data augmentation methods, and combines the outputs of multiple base models with ensemble methods.

3.1 Multimodal Base Models

The Hateful Memes Challenge (Kiela et al., 2020b), calling for state-of-the-art models for hate speech detection in multimodal memes, published baseline codes called MMF, which are modular and highly scalable. We thus chose MMF as the codebase of our base models. We extracted 100 features from each meme with the image feature extraction function of MMF and the configurative specifications produced by HateDetectron (Kiela et al., 2020b), the second runner-up of the Hateful Memes Challenge. The feature extractor is backboned by Faster-RCNN (Ren et al., 2015) and ResNet-152 (He et al., 2016).

Below are detailed descriptions of all the base models offered in MMF that we used.

Unimodal (text): We included a unimodal model as a control group. The modality for this model is configurable. After experimentation, we decided to go for text as the single modality for prediction, as it rendered better results than using only images or feature vectors.

ConcatBOW/BERT/MMF: MMF has baseline models for multimodal classification tasks. We obtain text representations from bag-of-word (BOW), BERT, and MMF models and image representations as feature vectors. The "Concat" models concatenate the two representations (text+image) as fusion, and the fusion results are then passed through a Multi-Layer Perceptron (MLP) to output predictions.

LateFusionMMF: The Late fusion model takes the mean between the text and image representations, instead of concatenation.

ViLBERT: Vision-and-Language BERT (ViLBERT) (Lu et al., 2019) is a BERT-based multimodal model pre-trained on the Conceptual Captions (Sharma et al., 2018) dataset, comprising captioned images. It uses two pre-trained strategies: (1) reconstructing image regions or words for masked inputs based on the unmasked portions and (2) prediction of multimodal image-text alignment.

The fundamental difference from the original BERT architecture lies in the attention mechanism, with co-attention used in place of self-attention. By exchanging key-value pairs in multi-headed attention, ViLBERT conducts vision-attended language attention in the visual stream and language-attended vision attention in the linguistic stream. ViLBERT has shown state-of-the-art performance on multiple vision-and-language tasks.

Visual BERT: Like ViLBERT (Li et al., 2019), Visual BERT is another BERT-based multimodal model. It reuses the self-attention system in transformers to implicitly match image regions with texts. The visual embeddings comprise segment embeddings, positional embeddings, and feature vectors of bounding boxes, generated by Faster-RCNN (Ren et al., 2015). They are then fed into the transformer together with text embeddings. With visual and linguistic inputs pre-trained together, the model can discover interesting alignments and implicitly construct effective joint representations.

Visual BERT was pre-trained task-agnostically on the COCO (Chen et al., 2015) dataset for two tasks: masked language modelling and sentence-image prediction. For better adaptation to a particular domain, Visual BERT is usually fine-tuned using masked language modelling on task data before it is applied to downstream tasks.

MMBT: The basic idea of Multimodal Bitransformers (MMBT) (Kiela et al., 2020a) is to apply self-attention to both modalities (i.e. text and image) all at once. This is achieved by utilizing segment embeddings to differentiate between the two modalities, the same method for typical question answering tasks to separate question from paragraph.

A fundamental difference between MMBT and other self-supervised architectures like ViLBERT and VisualBERT is that MMBT only pre-trains individual modalities unimodally. Such design has the plug-and-go advantage if a better vision or language model emerges. It is trivial to replace the pre-trained models for different modalities since they are pre-trained separately. On the other hand, MMBT cannot fully gauge the powerful attention mechanism on multimodal data during pre-training.

3.2 Multi-task Learning

Our baseline models' results (see Section 4) show a significant gap between Subtasks A and B and also uneven performance among the misogynistic sub-categories of Subtask B. This means the models fall short of generalizing the meaning of misogyny. Therefore, we adapted multi-task learning (MTL) to uncover the shared knowledge among all the misogynistic sub-categories.

A MTL learns multiple tasks simultaneously by constructing a generalized representation for the data in different yet related contexts. It is suitable for our dataset since we have five inter-correlated outputs (1 from Subtask A and 4 from Subtask B). To realize multi-task learning, we altered the architecture of the Visual BERT baseline model. While the encoder portion was kept unchanged, we duplicate the classification layer of the decoder into five, each associated with a sigmoid layer for prediction of one of the five outputs. We used binary cross-entropy (BCE) loss combined with Kullback–Leibler divergence loss (KL) as the loss function and summed all the losses generated by individual classification layers. For comparison purpose, we also tried MTL on only the four misogyny types of Subtask B, but the five layers configuration yielded better results.

In fact, we tried to fine-tune Visual BERT on Subtask A first before further fine-tuning it on Subtask B, because we view the misogynistic classification as the main task, and the type categorization task stems from it. Contrary to our expectations, we observed a decrease in model performance. Instead of the misogyny classification inferring the types of misogyny, the results could indicate that the types of misogyny dictate the misogyny classification.

3.3 Data Augmentation

Upon inspection of the validation dataset errors, we found that around 30% of them are blurry or have low resolution, and that about 17% of them express sarcasm against men but are misclassified as misogyny. To address this issue, we created new data by augmenting 10% of the original training set that compare males with females. We augmented them with nine transformations: (1) rotation, (2) Gaussian noise, (3) blurring, (4) horizontal flip, (5) contrast, (6) Affine transformation, (7) distortion, (8) elastic transformation, and (9) change in hue and saturation, with 100 memes per transformation. This boosted the robustness of our model to deal with various data qualities. The data augmentation library we used is *imgaug* (Jung et al., 2020).

3.4 Ensemble Learning

Ensemble learning (Opitz and Maclin, 1999) is an effective method that makes better achievements by combining multiple models' outputs. Taking advantage of the "wisdom of the crowd", an ensemble model can outperform a single contributing model.

Based on our baseline results (see Section 4), all the baseline models except Concat BOW and Unimodal Text have competency on at least one subtask. Thus, the ensemble model pool only excludes the two underperforming models. We experimented with two ensemble strategies, 1) an Multi-layer Perceptron (MLP) and 2) a majority voting layer that averages output probabilities, and compared their results, where both take as input the concatenation of the outputs of the selected baseline models and predict the final outputs per subtask[1].

4 Results

We show the evaluation results of the baseline models, multi-task learning and ensemble methods, and then based on the results, present the overall architecture of the final system we used for our official submission of the MAMI challenge in Section 4.4.

4.1 Baseline comparison

The baseline models serve as the starting point for our experimentation. We trained and evaluated all models included in Section 3.1 individually on all subtasks (misogyny, shaming, stereotype, objectification, violence). Table 1 summarizes the evaluation results.

[1]We refer to the lowercase "subtask" as the misogyny type columns (misogyny, shaming, stereotype, objectification, and violence), in comparison to the capitalized "Subtask" (Subtasks A and B) in the MAMI challenge.

Model	Misogyny	Shaming	Sterotype	Objectification	Violence
Unimodal (text)	80.9%	68.8%	72.2%	74.7%	65.3%
ConcatBOW	74.9%	67.2%	63.5%	73.3%	64.0%
ConcatBERT	**83.7%**	68.0%	71.6%	**77.5%**	**72.0%**
ConcatMMF	**84.4%**	69.6%	**72.7%**	**77.1%**	**74.0%**
Late Fusion	82.8%	**70.9%**	**72.3%**	76.1%	69.1%
ViLBERT	83.4%	69.8%	72.1%	**79.7%**	**75.2%**
VisualBERT	**84.4%**	**72.0%**	**73.8%**	75.5%	71.5%
MMBT	82.3%	**71.4%**	70.9%	73.6%	67.2%

Table 1: Baseline performance evaluated on the validation set. Scores in this table are macro-averaged F1-scores. The best **three** performance for each subtask are highlighted in bold.

Since there was no noticeable change in model performance after we tuned the hyperparameters, we set them as fixed values specified as follows: Enabled early stopping, learning rate of 2e-05, epsilon of 1e-05, batch size of 16, learning rate ratio of 0.6, and warm-up steps of 500.

As seen in Table 1, model performance varies significantly from task to task. Among all the subtasks, Shaming and Violence are the hardest to learn, carrying a non-negligible difference of more than 10% from the Misogyny subtask. This difference in performance will be addressed in the next section.

An interesting note is that early fusion (Snoek et al., 2005) models like Visual BERT are comparable to late fusion (Gunes and Piccardi, 2005) models such as Concat BERT and MMBT. Based on this finding, modal correlations may be picked up at both the decision level as well as the low-dimensional feature level, although the learnt correlations could be different.

Concat BOW performs the worst since it uses simple bags of words as text representation. Its performance is even worse than the unimodal text model. All other multimodal models achieved better results than the unimodal model, showing the involvement of visual information contributes to the overall understanding of the meme contents. We selected the best three models for each subtask for later experiments.

4.2 Multi-task learning evaluation

The alteration of Visual BERT for multi-task learning resulted in an overall advancement of 2.6% for Subtask B and a slight improvement of 1.3% for Subtask A. We also explored different specifications of the classification layers, adjusting the number of hidden layers and activation functions. However, none of them showed significant influence on model's performance.

Method	Subtask A	Subtask B
Top-3 models + MLP	**71.6%**	68.2%
Top-3 models + majority voting	66.1%	**69.5%**
Top-6 models + MLP	70.2%	67.9%
Top-6 models + majority voting	67.4%	69.2%

Table 2: Performance comparison between ensemble learning methods. Scores in this table are of the metrics used for Subtasks A and B, and evaluated on the test set.

4.3 Ensemble evaluation

Table 2 summarizes the evaluation results of two ensemble methods (MLP, majority voting) with the top-*k* models (*k*=3,6)[2]. For each subtask, the top-3 models used are highlighted in bold face in Table 1, and all models except Concat BOW and Unimodal Text are selected as the top-6 models. Refer to Table 1 for the top-3 models selected for each subtask.

The evaluation results of the ensemble methods do not identify a single best method. The majority voting with top-3 models shows the best performance for Subtask B, while MLP is the best for Subtask A. Based on these mixed results, we select different models and ensemble methods for the two Subtasks A and B, which is illustrated in the next section.

4.4 Overall system architecture and evaluation

After considering the evaluation results above, we selected the following three models as parts of the final system architecture:

[2]We refer to the lowercase "subtask" as the misogyny type columns (misogyny, shaming, stereotype, objectification, and violence), in comparison to the capitalized "Subtask" (Subtasks A and B) in the MAMI challenge.

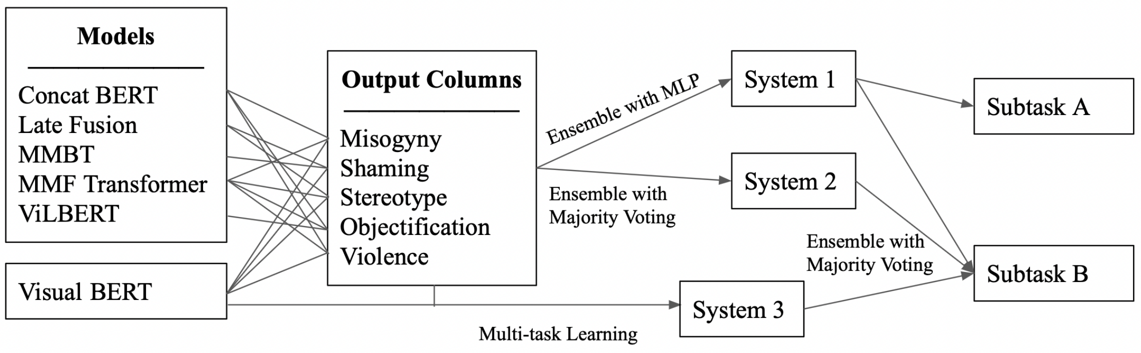


Figure 1: The high-level architecture of the multimodal multitask learning and ensemble learning framework

1. A Visual BERT based multi-task learning model
2. An ensemble learning model that feeds the output probabilities from the top-3 models individually trained on each subtask to a Multilayer Perceptron (MLP)
3. An ensemble learning model that does majority voting among the top-3 models individually trained on each subtask

In Subtask B, results are generated by applying yet another majority voting layer to these three models. In Subtask A, results are directly taken from the outputs of the ensemble model with MLP (the second model on the list), since we found applying majority voting to Subtask A produced suboptimal results. During the evaluation phase of the competition, we achieved 71.6% in Subtask A and 70.6% in Subtask B, ranked 4th and 6th on the leaderboard, respectively. Figure 1 illustrates how the models are integrated into the final system architecture.

5 Error Analysis

5.1 Analysis on Subtask A

Despite the competitiveness of our system, it still misinterprets the misogyny of some memes in Subtask A. A deeper examination reveals the most common error clusters as discussed below.

First, memes often include references to news, celebrities and cultural practices, e.g. those containing members of "the squad" in the 2019 US House election. This issue may be addressed by detecting entities in memes and collecting their details from the Web to better represent the entities.

Second, the current system falls short of extracting meaning from text positions in memes and the order of sentences (e.g. four-frame mangas). This issue may be addressed by correcting the sentence order and learning their alignment to meme frames.

Third, memes of anti-violence propaganda are often misclassified into the violence category since they include violent-looking images, while the caption is about fighting against violence. In contrast, other misogynistic memes portray violent acts against women in the caption and praise their actions in the image, e.g. with a thumbs-up icon or trophies.

Fourth, memes about comparison between men and women are often confusing. We noticed that these memes either mock males by comparing them to females or describe the real difference between males and females in a funny way. This issue may be addressed by using data augmentation of switching gender-referring words (e.g. man, woman, boy, girl) in the captions of those memes.

5.2 Analysis on Multi-task Learning

In the hold-out set from the training data, our model achieved 83% for Subtask A and 73% for Subtask B. Nevertheless, the same model only yielded 69.7% and 67.5% on the test data. This discrepancy may indicate that training and test data come from different distributions. Conventional multi-task learning is vulnerable to out-of-distribution because it assumes the predicted targets are independent given the input. Unfortunately, the target sub-categories are dependent on each other. This issue might be addressed by adapting, e.g. generative multi-task learning (Makino et al., 2022), which considers the dependency between targets.

6 Conclusion

In this paper, we presented our system for the Multimedia Automatic Misogyny Identification (MAMI) task in SemEval 2022. We augmented the data with visual transformation techniques and extracted features from memes by using multimodal baseline models. We further enhanced the system by adapting multi-task learning and ensemble learning methods. We leave issues such as incorporating external information about entities, frame layout in memes, gender-related text data augmentation and cross-subtask dependency in multi-task learning as future works.

Acknowledgements

We thank Yew Lee Tan for his technical advice. This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project #A18A2b0046).

References

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2019. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Berta Chulvi Aurora Saibene, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Francesca Gasparini, Giulia Rizzi, Aurora Saibene, and Elisabetta Fersini. 2021. Benchmark dataset of memes with text transcriptions for automatic detection of multi-modal misogynistic content.

H. Gunes and M. Piccardi. 2005. Affect recognition from face and body: early fusion vs. late fusion. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages 3437–3443 Vol. 4.

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

Alexander B. Jung, Kentaro Wada, Jon Crall, Satoshi Tanaka, Jake Graving, Christoph Reinders, Sarthak Yadav, Joy Banerjee, Gábor Vecsei, Adam Kraft, Zheng Rui, Jirka Borovec, Christian Vallentin, Semen Zhydenko, Kilian Pfeiffer, Ben Cook, Ismael Fernández, François-Michel De Rainville, Chi-Hung Weng, Abner Ayala-Acevedo, Raphael Meudec, Matias Laporte, et al. 2020. imgaug. `https://github.com/aleju/imgaug`. Online; accessed 01-Feb-2020.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, Ethan Perez, and Davide Testuggine. 2020a. Supervised multimodal bitransformers for classifying images and text.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020b. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems*, volume 33, pages 2611–2624. Curran Associates, Inc.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. VisualBERT: A simple and performant baseline for vision and language.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*.

Taro Makino, Krzysztof Geras, and Kyunghyun Cho. 2022. Generative multitask learning mitigates target-causing confounding.

D. Opitz and R. Maclin. 1999. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*.

Cees Snoek, Marcel Worring, and Arnold Smeulders. 2005. Early versus late fusion in semantic video analysis. pages 399–402.

Jiasen Lu Margaret Mitchell Dhruv Batra C Lawrence Zitnick Devi Parikh Stanislaw Antol, Aishwarya Agrawal. 2015. VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

MilaNLP at SemEval-2022 Task 5: Using Perceiver IO for Detecting Misogynous Memes with Text and Image Modalities

Giuseppe Attanasio and **Debora Nozza** and **Federico Bianchi**
Bocconi University
Via Sarfatti 25, 20136
Milan, Italy
{giuseppe.attanasio3,debora.nozza,f.bianchi}@unibocconi.it

Abstract

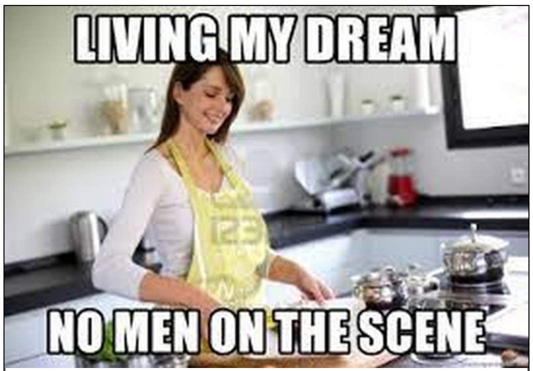
Warning: This paper contains examples of language that some people may find offensive or upsetting.

In this paper, we describe the system proposed by the *MilaNLP* team for the Multimedia Automatic Misogyny Identification (MAMI) challenge. We use Perceiver IO as a multimodal late fusion over unimodal streams to address both sub-tasks A and B. We build unimodal embeddings using Vision Transformer (image) and RoBERTa (text transcript). We enrich the input representation using face and demographic recognition, image captioning, and detection of adult content and web entities. To the best of our knowledge, this work is the first to use Perceiver IO combining text and image modalities. The proposed approach outperforms unimodal and multimodal baselines.

1 Introduction

Monitoring and detecting hateful content online is of paramount importance to limit the spread of hate, misconception, and prejudice. Content on social media platforms poses several challenges, from fast-paced, large-scale generation requiring automatic solutions, to ever-changing information mediums, like internet memes.

Misogynous memes are an unfortunately common phenomenon. Based on sexist preconceptions, these memes target and degrade women for humor. This intricate nature makes them hard to detect with classical computational models as hate is conveyed by associating known visual concepts with specific textual wording. In the Multimedia Automatic Misogyny Identification task (Fersini et al., 2022) novel systems are required to detect misogynous memes in English. The task is divided into two sub-tasks. Sub-task A requires solving the sole misogynous meme identification (i.e., a binary task). Sub-task B requires recognizing more specific categories, namely *stereotype*, *shaming*, *objectification*, and *violence*. Figure 1 (top) shows an example from the dataset.



F: White, 20-29, Female - Count: 1
W: "No men meme"
C: "A poster of a woman in a kitchen"
A: False

Figure 1: Sample from the training set (top) annotated with *Misogynous* and *Stereotype* labels. We enriched the meme with additional information (bottom), namely detected faces (F), web entities (W), caption (C), and adult content (A).

We propose a novel architecture where unimodal[1] components extract salient information from the meme. We present all information to a late fusion layer that distills it into a latent representation. We use renowned unimodal encoders networks and Perceiver IO (Jaegle et al., 2022) as the late fusion layer. Notably, while jointly learning from all modalities, Perceiver IO easily extends to multi-task learning. To the best of our knowledge, this work is the first to use Perceiver IO combining text and image modalities. We hence effectively address, with the same architecture, both sub-tasks A and B.

[1]We use *unimodal* whenever a single modality is involved, e.g., when dealing with Image content only. By extension, *multimodal* refers to a mixture of unimodals, e.g., Image and Text.

The proposed system outperforms both unimodal and multimodal baselines. The results show that Perceiver IO is an effective and efficient method to jointly fuse input representations from different modalities in multi-task setups. However, we achieved sub-par performance against other competing solutions. We ranked 25th (13 F1 points worse than the best system) out of 69 competing teams on sub-task A and 15th (4 F1 points worse than the best system) out of 42 competing teams on sub-task B. Our system is not specialized in either of the sub-tasks and hence it under-performs against task-engineered solutions. We report a brief error analysis in Section 5.

We release the code to replicate our experiments at `https://github.com/MilaNLProc/milanlp-at-mami`.

2 Background

In the last years, the task of hate speech detection has attracted considerable attention from the Natural Language Processing and Computer Vision communities. Among the research work in this area, only a limited number of approaches have focused on the problem of misogyny detection, which is a concrete problem in social media platforms. Nozza (2021) shows that hate speech detection models do not transfer across different hate speech targets, further demonstrating the need for ad-hoc misogyny detection approaches and datasets. Indeed, the corpora made available as part of shared tasks (Fersini et al., 2018, 2020b; Basile et al., 2019; Mulki and Ghanem, 2021) enabled a variety of NLP approaches to the problem of automatic misogyny detection on Twitter posts (Indurthi et al., 2019; Fersini et al., 2020a; Attanasio and Pastor, 2020; Lees et al., 2020, inter alia).

The Multimedia Automatic Misogyny Identification task focuses on the problem of misogyny detection with the new perspective of multimodality. The most similar research effort in the direction of hateful memes detection is the Hateful Memes Challenge (Kiela et al., 2020) and the MMHS150K corpus (Gomez et al., 2020).

On the other hand, multi-modal models that combine image and text are now becoming incredibly popular in Natural Language Processing (Cao et al., 2020; Lu et al., 2019; Tan and Bansal, 2019; Radford et al., 2021; Bianchi et al., 2021; Su et al., 2020, inter alia) due to their capabilities to solve zero-shot tasks.[2]

In this paper, we focus on the use of Perceiver IO (Jaegle et al., 2022) to combine the information coming from different sources such as the meme image, the text, and other features.

3 System overview

Following recent work addressing similar tasks (Pramanick et al., 2021; Zhu, 2020; Lee et al., 2021, inter-alia), we decompose the multimodal learning of hateful memes into two stages. First, we embed with unimodal encoders different input sources. Next, we adopt a multimodal late fusion to jointly learn from different modalities. With this setup, we tackle both sub-tasks A and B with no additional data other than the one provided for the task.

We build unimodal streams using pretrained, modality-specific encoder networks. Each encoder contributes an input representation to the subsequent fusion layer. We then use Perceiver IO (Jaegle et al., 2022) as a late fusion approach over the concatenation of modality-specific representations. Perceiver IO produces a structured, multidimensional output. We leverage this ability to jointly learn misogyny and other relevant aspects (e.g., aggressiveness, objectification, etc.) from the input. With that, we effectively solve both sub-tasks A and B of the challenge using a single model in a multi-task learning setting.

The system architecture is shown in Figure 2. To the best of our knowledge, this is the first attempt to use Perceiver IO as a multimodal late fusion layer for multi-task learning.

In the following sections, we further describe what type of unimodal source we considered (Section 3.1) and how we adapted Perceiver IO to multimodal, multi-task learning (Section 3.2).

3.1 Unimodal streams

The provided memes are characterized by two features: the meme image and the transcription of the over-imposed text. Building on ideas from recent work (Blaier et al., 2021; Pramanick et al., 2021), we enrich the meme with semantic information including image caption, face and demographics, detection of adult content, and web entities. We report a sample in Figure 1.

[2]See also Frank et al. (2021) for a detailed study and ablation analysis on the capabilities of these models.

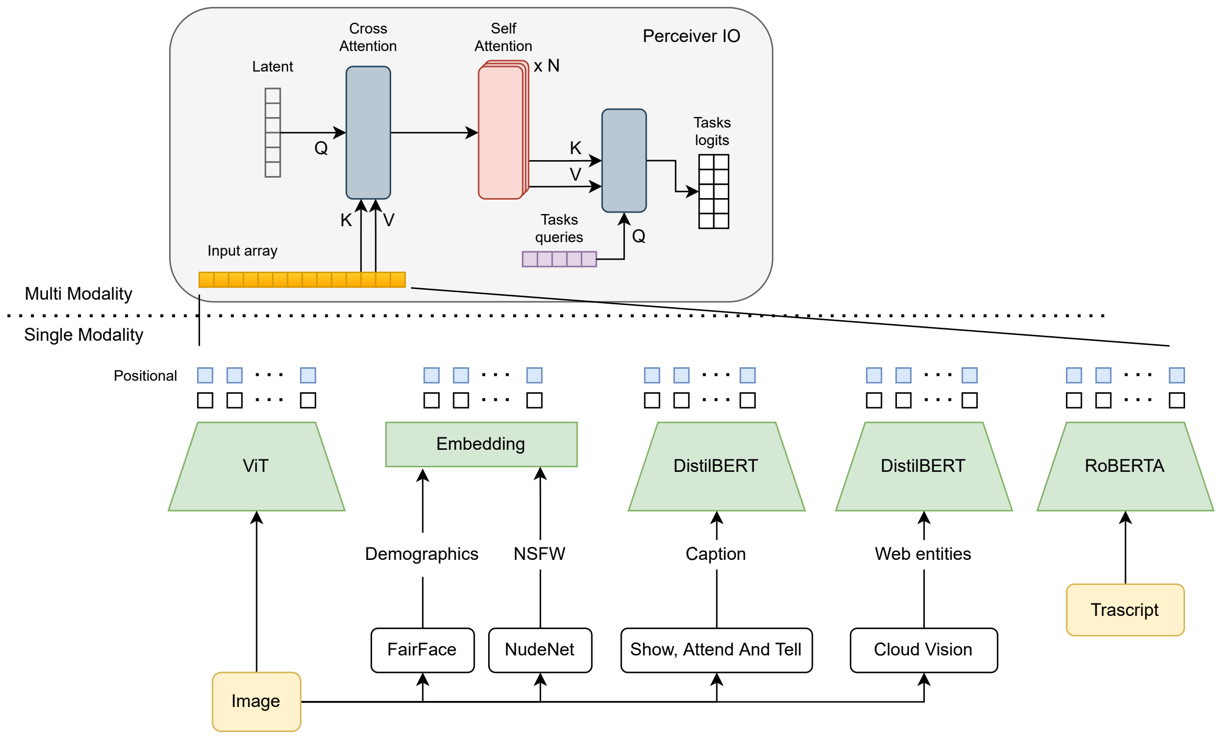


Figure 2: System overview. Light yellow boxes are the raw meme components. Green shapes are unimodal encoders. White boxes are external models used to enrich the meme. The darker yellow array represents the input array to the Perceiver IO fusion layer.

We derive a different input stream from each of the features of the semantically-augmented meme (Figure 2, bottom).

Image We encode raw images using a pretrained Vision Transformer (Dosovitskiy et al., 2021) (ViT). In this stream, the input image is divided into patches of 16x16 pixels and fed through a stacked transformer architecture. We use the last hidden token embeddings as the output of the stream.

Text Transcript We encode the text transcript using a pretrained RoBERTa (Liu et al., 2019) and use the last hidden token embeddings as the output of the stream. We choose RoBERTa based on its performance and its hurtful sentence completion (HONEST) score (Nozza et al., 2021, 2022b).

Face and Demographics Some images contain one or more faces. We use a pretrained Fair-Face (Karkkainen and Joo, 2021) model to detect faces and demographics. For each face found, we extract three categorical attributes, namely *Age*[3], *Gender*, and *Ethnicity*. Figure 1 (bottom, F) reports an example of face and demographics extracted with FairFace from a training sample. We encode face demographics with a simple embedding layer.

[3]FairFace produces age ranges, e.g., *60-75*, *75+*.

Adult Content We use NudeNet[4], a pretrained classification and detection model for nudity detection and censoring, to detect if the image contains adult content. We encode this information with additional embeddings.

Image caption We include an automatically generated image caption to have a textual description of the meme content. We use a pretrained Show, Attend and Tell model (Xu et al., 2015) to generate the caption for both training and test memes. Figure 1 (bottom, C) reports an example of an automatically generated caption. We encode the caption using a pretrained DistilBERT (Sanh et al., 2019).

Web Entities We use Google Cloud Vision API to detect web entities starting from the meme image.[5] As a summary of the extracted entities, we use the textual field `best guess labels` provided by the API. We encode this text using a pretrained DistilBERT. Note that this is a different model than the one used for captions.

[4]https://github.com/notAI-tech/NudeNet
[5]https://cloud.google.com/vision/docs/detecting-web

3.2 Late fusion with Perceiver IO

We concatenate all the information extracted by unimodal streams into a single input array (Figure 2, top). Ideally, this array contains all the raw unrouted information necessary for the task. We use a Perceiver IO-based late fusion layer for information distillation and multi-task learning.

Perceiver IO builds on Perceiver (Jaegle et al., 2021) a modality-independent neural architecture with two crucial advantages over unimodal models. First, it is designed not to leverage inductive biases as in modality-specific architectures. Because of that, it effectively learns from the input of any shape and nature. Second, Perceiver distills inputs in a much smaller latent representation for memory efficiency. On top of that, Perceiver IO (Jaegle et al., 2022) adds a decoding step to produce a structured output of arbitrary size and semantics. The output is generated using decoder queries cross-attending the latent representation. The number of queries and their dimension defines the output shape.

We use this feature of Perceiver IO to address both sub-tasks A and B at once. Specifically, we define five different *task queries*, one per characteristic of the misogynous meme identification problem, i.e., *misogyny*, *shaming*, *aggressiveness*, *objectification*, and *violence*. In this multi-task setup, we generate logits and extract probability distributions for each of the five aspects.

4 Experimental setup

The provided training set counts 10,000 samples. Class labels are balanced on misogyny ($p_1 = 0.5$) but unbalanced on *shaming* ($p_1 = 0.18$), *stereotype* ($p_1 = 0.28$), *objectification* ($p_1 = 0.22$), and *violence* ($p_1 = 0.095$). We validated our models and baselines using three-fold cross-validation over the training set. We measure performance with F1 Macro on the binary misogyny detection task.

For ViT, RoBERTa, and DistilBERT, we use implementations and checkpoints from the transformers library (Wolf et al., 2020) and HuggingFace Hub. We use monolingual English checkpoints for text models. For Perceiver IO, we use the lucidrains's PyTorch implementation.[6]

Unimodal encoders For ViT, we used the `google/vit-base-patch16-224-in21k` checkpoint. We used the standard feature processor which entails 1) resizing to a maximum shape of 224x224 and channel normalization. We also augmented the image using color jitter ($hue = 0.1$), random horizontal flip ($p = 0.5$), affine transformations,[7] contrast ($p = 0.3$) and equalization ($p = 0.3$) variations. We discard the CLS last token embedding and use the sequence of the remaining 196 tokens as the image representation.

For RoBERTa, we used the `roberta-base` checkpoint and its standard tokenizer padding and truncating up to a maximum of 32 tokens. We performed text augmentation via token removal ($p = 0.15$).

For DistilBERT, we used the `distilbert-base-uncased` checkpoint and its standard tokenizer padding and truncating up to a maximum of 32 tokens. Note that this configuration is duplicated for both the text transcript and web entity streams.

Perceiver IO We did not use a Perceiver IO pretrained checkpoint. We manually fine-tuned the number of latent variables in $\{64, 128, 256, 512, 1024\}$, latent dimension in $\{128, 256, 512, 1024\}$, and number of self-attention layers in $\{6, 12\}$, and settled on the configuration `[265, 512, 6]`. Self-attention layers are applied sequentially and do not share weights. We use 5 decoder queries to produce a structured output of shape $(5, 2)$. We project the output using a final linear layer and consider the result as the tasks logits. We also tried to use a different linear projection layer per task but with no measurable improvement in performance.

Training setup We trained the entire system end-to-end, i.e., we jointly optimized all unimodal encoders *and* Perceiver IO. Following Bianchi et al. (2021), we tried to freeze the encoders for an arbitrary number of steps and then unfreeze them, with no significant improvement.

We manually tuned the learning rate in the range $[10^{-5}, 10^{-6}]$. We then used 10^{-5} with linear decay and 10% of total steps as warmup steps. We set weight decay to 10^{-2}. We trained the system for four epochs using Focal Loss (Lin et al., 2020) to account for class imbalance. We set alpha to 0.25 and gamma to 2.

[6]https://github.com/lucidrains/perceiver-pytorch

[7]For the full set of affine transformations please refer to our repository.

Model	F1
Lexicon BoW + LR	66.4
RoBERTa	77.4
Vision Transformer	73.1
Perceiver IO	82.9
Perceiver IO (FWCA)	**83.3**

Table 1: F1 performance (%) on our three-fold cross-validation setup of unimodal (top) and multimodal (bottom) models in the misogyny identification task (sub-task A).

Model	F1
Best system	**83.4**
Perceiver IO (FWCA)	69.9
Provided baseline	64.0
Best system	**73.1**
Perceiver IO (FWCA)	69.3
Provided baseline	62.1

Table 2: Performance on sub-task A (top) and sub-task B (bottom) compared to best systems and *baselines* provided by task organizers.

5 Results

The provided test set counts 1,000 samples. Target labels are balanced in terms of misogyny but imbalanced for the rest of the categories. Unbalancing is slightly more marked than the training set. We report the prior frequency of the positive class as p_1 in Table 3.

In the following, we compare the performance of the proposed system which encodes detected faces (F), web entities (W), the image caption (C), and adult content detection (A). We refer to the system as Perceiver IO (FCWA). It achieves an overall F1 (macro) score of 69.9% in sub-task A and an F1 (weighted) score of 69.3% in sub-task B. We rank 31th (13 F1 points worse than the best system) on sub-task A and 18th (4 F1 points worse than the best system) on sub-task B. The system outperforms several baselines provided by the task organizers. On sub-task A, we outperform 1) sentence embeddings from a pretrained Universal Sentence Embedding model, 2) an image classifier fine-tuned from a VGG model, and 3) a classifier based on the concatenation of the first two representations plus a single layer neural network. On sub-task B, we outperform 1) a multi-label model based on the concatenation of deep image and text embeddings and 2) a hierarchical multi-label model based on text representations. Results are reported in Table 2.

Category	F1	P	R	p_1
misogynous	69.91	75.07	71.10	0.50
shaming	65.98	65.69	66.31	0.15
stereotype	67.79	68.58	67.34	0.35
objectification	70.06	71.72	69.30	0.35
violence	74.29	82.73	70.27	0.15

Table 3: Performance on the test set in terms of F1 (%), Precision (P), and Recall (R) with macro averaging. Prior frequency (p_1) of the positive class in the training set.

5.1 Quantitative analysis

In the proposed dataset, misogyny characteristics (e.g., *shaming* or *objectification*) apply only to misogynous content. Hence, we consider performance on sub-task A (binary misogyny detection) as a proxy for the overall quality of the model configuration. We report performance on our cross-validation over the training set in Table 1.

We compared our system with internal unimodal baselines. For the textual modality, we tested a Bag-of-Word (BoW) representation[8] extracted from the HurtLex lexicon (Bassignana et al., 2018) fed to a Logistic classifier and RoBERTa. For the visual modality, we tested Vision Transformer. Our system outperforms by a large margin these unimodal baselines.

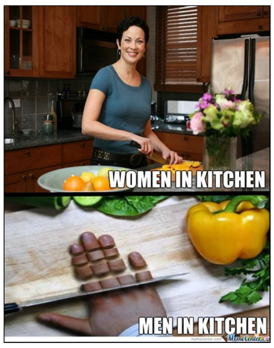
Further, we validated the intuition of semantically enriched memes with input ablation. Results are reported in Table 1. Specifically, we removed all streams but the encoders of the raw image and the transcript (Perceiver IO). Cross-validation results show that our enriched input (Perceiver IO (FWCA)) improves classification performance.

5.2 Error analysis

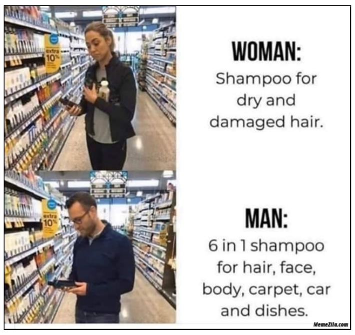
In the post-evaluation phase of the task, we studied the labels predicted by our system on the 1,000 test memes. Results are reported in Table 3 separately by category.

The system effectively learned all categories (F1 is always greater than 65%) but with differences. The misogynous identification task has a 70% F1

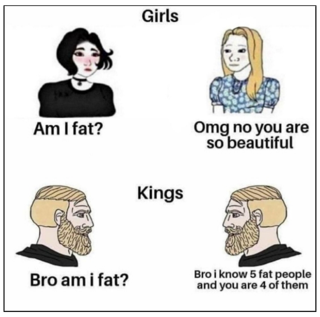
[8]We use a binary presence matrix, i.e., the rows are the transcript of the meme, the columns each of the terms of the lexicon, and the cell is 1 if the transcript contains the term at least once or 0 otherwise.



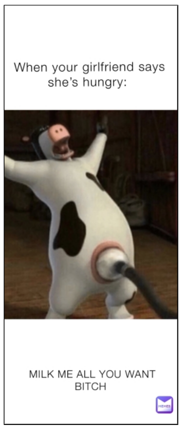
(a)


WOMAN:
Shampoo for dry and damaged hair.
MAN:
6 in 1 shampoo for hair, face, body, carpet, car and dishes.

(b)

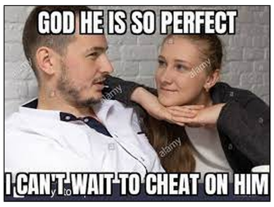

Girls
Am I fat?
Omg no you are so beautiful
Kings
Bro am i fat?
Bro i know 5 fat people and you are 4 of them

(c)

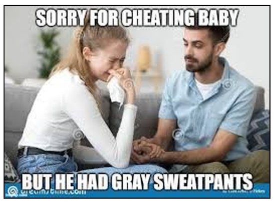

When your girlfriend says she's hungry:
MILK ME ALL YOU WANT BITCH

(d)

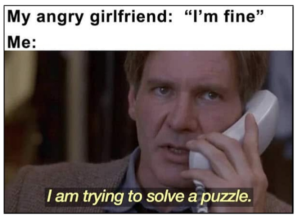
Figure 3: A sample of false positives from the test set on the misogyny identification task.



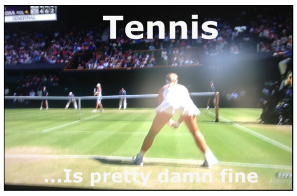
(a)



(b)


My angry girlfriend: "I'm fine"
Me:
I am trying to solve a puzzle.

(c)



(d)

Figure 4: A sample of false negatives from the test set on the misogyny identification task.

score composed of a promising 75% in precision and 71% recall. While performance between *stereotype* and *objectification* are close, the system underperformed in the *shaming* category. We think this behavior is due to the broad kind of both visual and textual content that can convey *shameful* messages and is hence more difficult to learn. Finally, the model performed best in *violence*, where it is probably simpler to identify visual clues that let intend a violent message.

We also manually inspected the errors of our model. We conducted the analysis separately by false positives (Figure 3) and false negatives (Figure 4). In the following, we speculate on the possible causes of these errors.

Weak Adult Content detection We noticed several wrong annotations extracted by NudeNet in both false positives and false negatives (e.g., all memes in Figure 3 are labeled as NSFW). We believe this might have introduced noise in our training that further propagated in classifying test memes.

Bias towards composite memes Several memes have a composite nature. They contrast some behavior or reaction of two groups (e.g., boys and girls) using a predefined structure. A typical example is achieved by organizing one group on top of another (Figure 3b and 3c).

We noticed several composite memes among false positives. We argue this kind of meme has a strong association bias with the positive class (*misogynous = 1*). Indeed, we believe that, in absence of relevant information, the system leverages the structure of the memes and wrongly produces a positive prediction.

Hard stereotypes Several memes contain non-hateful wording and image content. However, they convey misogynous messages because the combination of image and text leverages a well-known stereotype about women. We argue that the correct classification of these memes must involve either a solution explicitly modeling the stereotype or sufficient training data to become *aware* of it. We call "hard stereotypes" those memes whose stereotypical message is not backed by sufficient training data. We noticed several hard stereotypes in our misclassified memes. Figure 4 shows four examples of those.

Gender Bias We noticed several memes misclassified as false negatives depicting only a man in the scene. We believe the system might have learned a spurious bias associating the presence of men in the image with the negative class (*misogynous = 0*).

5.3 Further considerations

Quality of generated information In our experiments, automatically generated image captions often describe coherently the content of the image, although we do not expect them to generalize well to complex and diverse concepts.

Instead, FairFace extracts almost always precise face counts and age, ethnicity, and gender for each face even in crowded images (there are cases of 14 people in a single image). Web entities are often significant but provide coarse-grained information (e.g., "internet cat meme").

Unconditioned prediction During the post-evaluation phase, we identified a potential weakness in the proposed architecture.

Decoding queries in Perceiver IO are independent by design. This behavior enhances the model's generalization capabilities as it needs to learn internal representations useful for all outputs. However, it also prevents any conditioning between them. In our task, the *misogynous* aspect influences the remaining ones in the sense that only misogynous memes are characterized by one or more categories for sub-task B. We are not explicitly modeling this condition and hence probably hindering the performance.

6 Conclusion

We addressed both sub-tasks A and B of the Multimedia Automatic Misogyny Identification shared task with a novel Perceiver IO-based system. We take advantage of pretrained encoders and external services to extract and enrich with salient information the input meme. Then, we use Perceiver IO as a multimodal, multi-task late fusion layer of several unimodal streams. To our knowledge, this is the first time Perceiver IO has been used to combine text and image modalities.

The proposed system outperforms unimodal and multimodal baselines but underperforms against more specialized, task-specific competitor systems. We ranked 25th out of 69 competing teams on sub-task A and 15th out of 42 competing teams on sub-task B. In future work, we will explore improved input preprocessing (e.g., for the OCR-based text provided), and model ensembling. Additional effort should be put into identifying and mitigating unintended bias that may be present in our multimodal misogyny detection model following approaches proposed for text modality (Nozza et al., 2019; Attanasio et al., 2022a,b). The development of these multi-modal hate speech classifiers can be useful for the automatic evaluation of large pre-trained models (Nozza et al., 2022a).

Acknowledgements

This project has partially received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 949944, INTEGRATOR), and by Fondazione Cariplo (grant No. 2020-4288, MONICA). The authors are members of the MilaNLP group and the Data and Marketing Insights Unit of the Bocconi Institute for Data Science and Analysis. Giuseppe Attanasio did part of the work at the DataBase And Data Mining Group at the Polytechnic University of Turin. Computing resources were provided by the SmartData@PoliTO center on Big Data and Data Science.

References

Giuseppe Attanasio, Debora Nozza, Dirk Hovy, and Elena Baralis. 2022a. Entropy-based attention regularization frees unintended bias mitigation from lists. In *Findings of the Association for Computational Linguistics: ACL2022 (Forthcoming)*. Association for Computational Linguistics.

Giuseppe Attanasio, Debora Nozza, Eliana Pastor, and Dirk Hovy. 2022b. Benchmarking Post-Hoc Interpretability Approaches for Transformer-based Misogyny Detection. In *Proceedings of the First Workshop on Efficient Benchmarking in NLP*. Association for Computational Linguistics.

Giuseppe Attanasio and Eliana Pastor. 2020. PoliTeam @ AMI: Improving sentence embedding similarity with misogyny lexicons for automatic misogyny identificationin italian tweets. In Valerio Basile, Danilo Croce, Maria Maro, and Lucia C. Passaro, editors, *EVALITA Evaluation of NLP and Speech Tools for Italian - December 17th, 2020*, pages 48–54. Accademia University Press.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A multilingual lexicon of words to hurt. In *5th Italian Conference on Computational Linguistics, CLiC-it 2018*, volume 2253, pages 1–6. CEUR-WS.

Federico Bianchi, Giuseppe Attanasio, Raphael Pisoni, Silvia Terragni, Gabriele Sarti, and Sri Lakshmi. 2021. Contrastive language-image pre-training for the italian language. *arXiv preprint arXiv:2108.08688*.

Efrat Blaier, Itzik Malkiel, and Lior Wolf. 2021. Caption enriched samples for improving hateful memes detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9350–9358, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jize Cao, Zhe Gan, Yu Cheng, Licheng Yu, Yen-Chun Chen, and Jingjing Liu. 2020. Behind the scene: Revealing the secrets of pre-trained vision-and-language models. In *Computer Vision – ECCV 2020*, pages 565–580, Cham. Springer International Publishing.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elisabetta Fersini, Debora Nozza, and Giulia Boifava. 2020a. Profiling Italian misogynist: An empirical study. In *Proceedings of the Workshop on Resources and Techniques for User and Author Profiling in Abusive Language*, pages 9–13, Marseille, France. European Language Resources Association (ELRA).

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018. Overview of the EVALITA 2018 task on automatic misogyny identification (AMI). *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2018)*, 12:59.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020b. AMI @ EVALITA2020: Automatic misogyny identification. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021. Vision-and-language or vision-for-language? on cross-modal influence in multimodal transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9847–9857, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1459–1467.

Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. 2019. FERMI at SemEval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 70–74, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J Henaff, Matthew Botvinick, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. 2022. Perceiver IO: A general architecture for structured inputs & outputs. In *International Conference on Learning Representations*.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR.

Kimmo Karkkainen and Jungseock Joo. 2021. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1548–1558.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.

Roy Ka-Wei Lee, Rui Cao, Ziqing Fan, Jing Jiang, and Wen-Haw Chong. 2021. *Disentangling Hate in Online Memes*, page 5138–5147. Association for Computing Machinery, New York, NY, USA.

Alyssa Lees, Jeffrey Sorensen, and Ian Kivlichan. 2020. Jigsaw @ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model. In *Proceedings of Seventh Evaluation Campaign of Natu-*

ral Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), Bologna, Italy. CEUR.org.

T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. 2020. Focal loss for dense object detection. volume 42, pages 318–327, Los Alamitos, CA, USA. IEEE Computer Society.

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

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Hala Mulki and Bilal Ghanem. 2021. Working notes of the workshop arabic misogyny identification (armi-2021). In *Forum for Information Retrieval Evaluation*, FIRE 2021, page 7–8, New York, NY, USA. Association for Computing Machinery.

Debora Nozza. 2021. Exposing the limits of zero-shot cross-lingual hate speech detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 907–914, Online. Association for Computational Linguistics.

Debora Nozza, Federico Bianchi, , and Dirk Hovy. 2022a. Pipelines for Social Bias Testing of Large Language Models. In *Proceedings of the First Workshop on Challenges & Perspectives in Creating Large Language Models*. Association for Computational Linguistics.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2021. HONEST: Measuring hurtful sentence completion in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2398–2406, Online. Association for Computational Linguistics.

Debora Nozza, Federico Bianchi, Anne Lauscher, and Dirk Hovy. 2022b. Measuring Harmful Sentence Completion in Language Models for LGBTQIA+ Individuals. In *Proceedings of the Second Workshop on Language Technology for Equality, Diversity and Inclusion*. Association for Computational Linguistics.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *IEEE/WIC/ACM International Conference on Web Intelligence*, WI '19, page 149–155, New York, NY, USA. Association for Computing Machinery.

Shraman Pramanick, Shivam Sharma, Dimitar Dimitrov, Md. Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021. MOMENTA: A multimodal framework for detecting harmful memes and their targets. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4439–4455, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

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

Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 2048–2057. JMLR.org.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

UniBO at SemEval-2022 Task 5: A Multimodal bi-Transformer Approach to the Binary and Fine-grained Identification of Misogyny in Memes

Arianna Muti and **Katerina Korre** and **Alberto Barrón-Cedeño**
Department of Interpreting and Translation
Alma Mater Studiorum–Università di Bologna
Forlì, Italy
{arianna.muti2,aikaterini.korre2,a.barron}@unibo.it

Abstract

We present our submission to SemEval 2022 Task 5 on Multimedia Automatic Misogyny Identification. We address the two tasks: Task A consists of identifying whether a meme is misogynous. If so, Task B attempts to identify its kind among shaming, stereotyping, objectification, and violence. Our approach combines a BERT Transformer with CLIP for the textual and visual representations. Both textual and visual encoders are fused in an early-fusion fashion through a Multimodal Bidirectional Transformer with unimodally pretrained components. Our official submissions obtain macro-averaged F_1=0.727 in Task A (4th position out of 69 participants) and weighted F_1=0.710 in Task B (4th position out of 42 participants).

1 Introduction

Evolving from the two previous editions of the Automatic Misogyny Identification initiatives (Fersini et al., 2018, 2020), the Multimedia Automatic Misogyny Identification shared task at SemEval 2022 (MAMI) targets multimodal data for the first time. Within MAMI, Fersini et al. (2022) propose two classification tasks:

Task A A basic task about misogynous meme identification, where a meme should be categorized either as misogynous or not.

Task B An advanced task, where the type of misogyny should be recognized among potentially overlapping categories: stereotyping, shaming, objectification and violence.

The increasing volume of meme posts on social media renders the development of models able to identify malicious instances timely. The task is more challenging than when dealing with text alone because, in general, both the textual and the visual channels play an indivisible role in conveying the desired message.[1]

We build upon our previous experience in identifying misogyny and aggressiveness in text (Muti and Barrón-Cedeño, 2020) and approach both multimodal tasks with a supervised multi-modal bi-transformer model (MMBT) (Kiela et al., 2020a). We use *bert-base-uncased-hatexplain* (Mathew et al., 2020) and *bert-base-uncased* (Devlin et al., 2019) for the textual embeddings, and CLIP (Radford et al., 2021) for the visual ones. We also build two unimodal baselines to compare against.[2]

Our experiments aim at understanding if and to what extent our multimodal model outperforms the two unimodal ones that address the problem separately. Since meme classification is a challenging task due to its multimodal nature, we shed some light on which component should weigh more in the decision process —text or image— by observing the impact of both modalities in the predictions.

Our official submission for Task A achieved a macro-averaged F_1 score of 0.727, whereas that for Task B obtained a weighted F_1 score of 0.710, positioning our team in the 4th position in the task, for both tasks.

In addition, we identify the linguistic and visual elements which are potentially responsible for the misclassification. We perform an error analysis to check whether misclassified memes rely heavily on external world knowledge and/or are subtle and implicit, as we believe that those two aspects cause struggle to the models.

The rest of the paper is structured as follows. Section 2 provides essential definitions for this task, such as misogyny and memes. We then move on to dataset description as well as a summary of related work. Section 3 describes our models for both tasks.

[1]This is different from other multimodal scenarios, such as visual question answering or image captioning, where one of the two modalities tends to be the dominant one (Zhu, 2020).

[2]Our model is publicly available at https://github.com/TinfFoil/Unibo-at-SemEval-2022-MAMI.

	train	test
Not Misogynous	5,000	500
Misogynous	5,000	500

Table 1: Class distribution for the binary Task A misogynous or not.

Section 4 describes the experimental setup. Our results are presented and discussed in Section 5. Section 6 presents our error analysis. Section 7 concludes with a summary of our findings.

2 Background

2.1 Definitions

Memes are relatable acts of communication made of visual and textual artifacts, where often an image is superimposed with text with a humorous purpose (MacDonald and Wiens, 2022). To be fully understood, memes require context and real-world knowledge. They are often satirical, implying humour and sarcasm in a subtle way (Sharma and Pulabaigari, 2020). These factors cause the identification of phenomena in them —such as expressions of misogyny— difficult.

Humour does not always come as harmless fun and that is the case with misogynous memes. Such memes contribute to the establishment of a rape culture (Ridgeway, 2014), where violence and sexual harassment are tolerated, belittled, normalized, excused and transformed into jokes. Therefore, developing automatic approaches to tackle misogyny has both technological and social value.

According to the MAMI guidelines (Fersini et al., 2022), a meme is misogynous when it conveys an offensive, sexist or hateful message (be it weak or strong, implicitly or explicitly) targeting a woman or a group of women. Four kinds of misogyny are considered for this task:

Shaming occurs when memes insult or offend women because of their physical aspect.

Stereotyping represents a fixed idea or set of characteristics; physically or ideological.

Objectification represents a woman like an object through the over-appreciation of her physical appeal (sexual objectification) or by depicting her as an object (a human being without any value as a person).

Violence shows physical or verbal violence toward women.

	train	test	preds
Shaming	1,274	146	130
Stereotype	2,810	350	379
Objectification	2,202	348	334
Violence	953	153	102

Shaming	Stereotyping	Objectification	Violence	train	test	preds
✓				400	24	32
	✓			1,247	32	152
		✓		992	37	96
			✓	250	19	21
✓	✓			286	32	20
✓		✓		161	25	40
✓			✓	11	2	0
	✓	✓		412	152	118
	✓		✓	302	40	31
		✓	✓	116	38	23
✓	✓	✓		301	45	32
✓	✓		✓	55	3	1
✓		✓	✓	12	5	0
	✓	✓	✓	162	36	20
✓	✓	✓	✓	45	10	5
			Total	4,752*	500	591

* 248 of the misogynous memes lack type annotation.

Table 2: Number of instances per class for the multi-label Task B (top). Class distribution (bottom). Column **preds** shows the predictions of our best submitted model (Multi; cf. Section 5).

2.2 Datasets

The datasets are balanced with respect to Task A (see Table 1). The same instances include the multi-label annotation for Task B. Table 2 shows the number of instances for each label combination.

Stereotyping is the most represented class, with $3.2k$ instances overall, followed by objectification ($2.2k$) and shaming ($1.2k$); violence is the least represented, with less than $1k$. The label overlapping plays an important role in our results analysis (c.f., Section 6). As Table 2 shows, stereotyping and objectification tend to come together, whereas shaming and violence do not. We will explore whether our models capture this intersection in Section 6.

2.3 Related Work

The identification of misogyny in textual form was explored in the series of shared tasks on Automatic Misogyny Identification (AMI), held at IberEval (Anzovino et al., 2018) and EVALITA (Fersini et al., 2018, 2020). AMI at IberEval 2018 focused on identifying misogyny on English and Spanish tweets and on classifying the misogynous tweets into seven categories: discredit, stereotype, objectification, sexual harassment, threats of violence, dominance, and derailing. AMI at EVALITA 2018 targeted the analysis of Italian and English tweets. Task A addressed misogyny identification as well. Task B aimed at recognizing whether the target of a misogynous post was a specific person or women in general, and at classifying the positive instances in the aforementioned categories. AMI at EVALITA 2020 targeted the detection of misogyny and aggressiveness in Italian tweets (Task A) and the identification of unintended bias (Task B).

Multimodality has been explored for the automatic analysis of memes. Sharma and Pulabaigari (2020) worked in the task of identifying whether an image is a meme or not. Two recent SemEval tasks have targeted memes as well. Sharma et al. (2020) proposed an emotion identification task. The best performing system consisted of a text-only approach, a feed-forward neural network (FFNN) wth word2vec embeddings (Keswani et al., 2020). Dimitrov et al. (2021) proposed a shared task on the identification of propaganda techniques. Feng et al. (2021) approached it with a pre-trained transformer using text with visual features. They extracted grid features using ResNet50 (He et al., 2016) and salient region features using BUTD (Anderson et al., 2018). They further used these grid features to capture the high-level semantic information in the images. Moreover, they used salient region features to describe objects and to caption the event present in a meme. They built an ensemble of fine-tuned DeBERTA+ResNet, DeBERTA+BUTD, and ERNIEVIL (Yu et al., 2021) models.

Multimodal hate speech has attracted the interest of the research community only recently. In 2019, Facebook AI launched the Hateful Memes Challenge (Kiela et al., 2021b), which consisted in identifying hate speech in memes: hateful vs not. It is constructed such that unimodal models struggle and only multimodal models can succeed: difficult examples ("benign confounders") are added to the dataset to make it hard to rely on unimodal signals. The most successful approaches used both *early fusion* and *late fusion* (Kiela et al., 2021a), with the former achieving the best results. Those include VilBERT (Lu et al., 2019), VisualBERT (Li et al., 2019), MMF (Singh et al., 2020), MMBT (Kiela et al., 2020a) and CLIP (Radford et al., 2021). In late fusion approaches, systems for each modality are trained independently. The scores produced by each model are joined during inference to produce the final prediction (Kiela et al., 2020b).

In early fusion the different modalities are combined at an early stage to learn one single classification model (Kiela et al., 2020b). The top-performing model applied optical character recognition to find and remove the text from the input images in order to improve the quality of object detection, named entity identification and human race detection, using all these tags as input for different transformer models (Zhu, 2020).

Although MAMI is the first shared task on misogyny detection on memes, there has been preliminary work on automatic detection of sexist memes. Fersini et al. (2019) explored unimodal and multimodal approaches both with late and early fusion to understand the contribution of textual and visual cues on the MEME dataset, a dataset containing 800 sexist and not sexist memes. The sexist memes were also annotated according to aggressiveness and irony. From their work emerged that a unimodal textual model performs better than image-based ones. Concerning multimodality, late-fusion worked better.

3 System Overview

Our approach is based on the multimodal bi-transformer model (MMBT) (Kiela et al., 2020a). MMBT fuses image and text embeddings in an early fashion. MMBT jointly finetunes unimodally pretrained text and image encoders by projecting image embeddings to the text token space. Figure 1 represents the model architecture. MMBT combines two segments: segment 0 corresponds to the text, whereas segment 1 corresponds to the picture. They are fed together to use attention over both modalities at the same time. Each token is indexed according to its position from 0 to the maximum text length, which we set to 80. Each image representation is indexed from 0 to 640.

The original MMBT combines BERT (Devlin et al., 2019) and ResNet (He et al., 2016). We con-

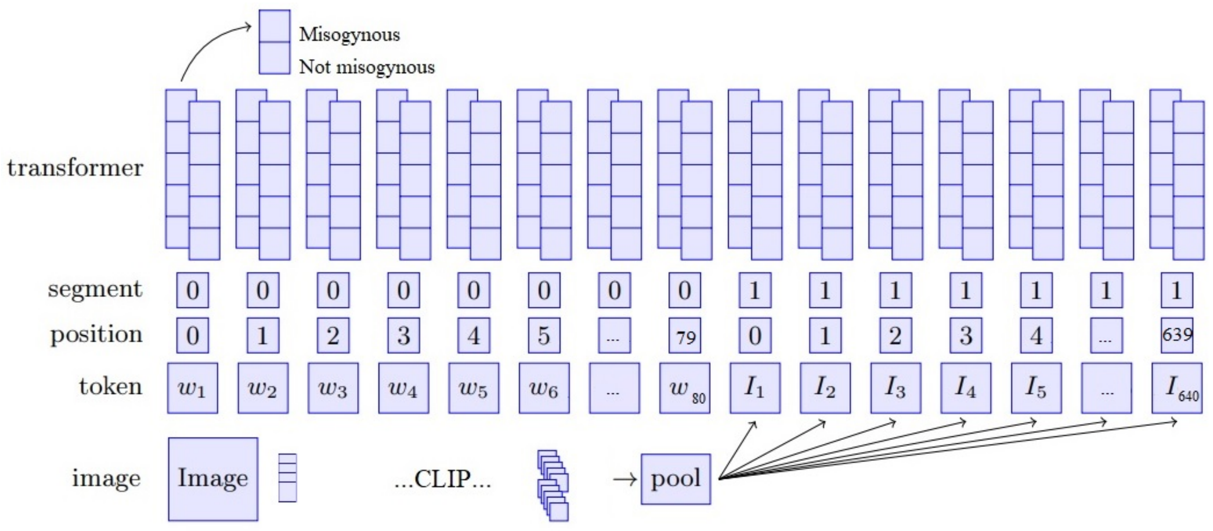


Figure 1: Representation of the MMBT model architecture combining CLIP and BERT; adapted from (Kiela et al., 2020a).

sider other models. For the textual embedding we tried with *bert-base-uncased-hatexplain* (Mathew et al., 2020), a version of BERT trained on identifying hate speech. For the image embedding we used CLIP, since it has outperformed all alternative models in a large variety of multimodal tasks, including OCR, action recognition in videos, geo-localization, and various types of fine-grained object classification (Radford et al., 2021). CLIP is pre-trained on the task of predicting which caption should be tied together with a given image. In this way it learns state-of-the-art image representations from scratch, enabling zero-shot transfer of the model to downstream tasks.

The two embeddings are fused through MMBT. For Task A we use the sigmoid activation function for the output layer and threshold at 0.5 to discriminate between misogynous or not. For Task B we adopt a binary relevance approach (Zhang et al., 2017), combining four binary classification models. The output for each classifier is a sigmoid function too. We opt for this approach after observing that treating the classes separately increased the performance in a multi-class model predicting misogynous, misogynous-aggressive or none (Muti and Barrón-Cedeño, 2020). This approach allows us to predict multiple mutually non-exclusive classes.

We apply a heuristic to refine the multi-label decisions in Task B. All four decisions are turned off if an instance had not been predicted as misogynous by our Task-A model.

Pre-processing Since CLIP requires square images, following Neskorozhenyi (2021) we produce 288×288 pixel versions of all memes. The memes come in different sizes and orientations, hence we rescale them until the largest side reaches 288 pixels respecting the aspect ratio and pad to fill the empty pixels in the square. Then, we slice the resized images into three equal parts horizontally if the image orientation is landscape and vertically if it is portrait to obtain both global and local image features. We extracted four vectors for each image: a vector for each part encoding spatial information and one for the whole image. We used the Pillow library (Clark, 2015) to perform these operations.

No preprocessing is applied to the text, other than applying the BertTokenizer (Devlin et al., 2019).

4 Experimental Setup

We shuffled the training set and take 10% of the data for development preserving the class distribution through stratified random sampling (Pedregosa et al., 2011).

We trained three alternative models to identify the best possible configuration. **Uni$_{\textbf{txt}}$** is a BERT-based unimodal system that considers the text alone. **Uni$_{\textbf{img}}$** is a CLIP-based unimodal system that considers the image alone. **Multi** is a multimodal system, fusing BERT and CLIP embeddings through MMBT.[3]

[3]We tried a variation of **Uni$_{\textbf{txt}}$** for Task A. We augmented the training data with the tweets corpus from AMI at Evalita

model	**variation**	**macro F_1**
Multi_5	after 5 epochs	0.703
Multi_6	after 6 epochs	**0.727**
Uni_{txt}	bert-base-uncased	0.656
Uni_{txt}	bert-Hatexplain	0.569
Uni_{img}	with fine tuning	0.703
Uni_{img}	zero-shot	0.417

Table 3: Official macro-averaged F_1-measures for our submissions to Task A.

Hyperparameters For Multi, we tried learning with different numbers of learning epochs, in range [3, 6]. The best validation performance was obtained after 5 epochs in Task A in the development set. We trained over 5 epochs in Task B. For both Uni_{txt}we train over 4 epochs. For the Uni_{img}we train over 5 epochs. In all cases we saved the model only when an increase in the performance was obtained. Since we also aimed at assessing how CLIP performs in making zero-shot predictions in this task, we used CLIP without fine-tuning on the training set, to check whether it could be effectively used to detect misogyny without prior annotation. We considered batch sizes of 16 and 32, with the former consistently performing better. We used a learning rate of 2e-4, the MADGRAD optimizer (Defazio and Jelassi, 2021), and a binary cross-entropy loss function.

The results reported in Section 5 are obtained with a model trained during 5 epochs for Task B and 6 epochs for Task A with 16 as the batch size.

Evaluation metrics We stick to the official MAMI evaluation metrics: macro-averaged F_1-measure for the binary Task A and weighted-averaged F_1-measure for the multi-label Task B.

5 Results

In this section we present the results obtained by our submissions to both Task A and Task B.

5.1 Task A

Table 3 shows the results of our submitted runs for Task A. The highest score is obtained after training the multimodal model Multi during 6 epochs: F_1=0.727. Considering the textual information alone runs short; the highest performance being obtained when the Uni_{txt} model is trained upon a *generic* BERT: 0.656. As expected, the zero-shot Uni_{img}model performs the worst, with a performance lower than that of a random model. A proper fine-tuning of the Uni_{img} model turns into the runner-up performance with F_1=0.703. The improvement of the Uni_{img}over the Uni_{txt}model by five points suggests that the visual information is captured better than the textual one. The reason might be that the text is too short and out of context to be captured effectively by BERT.

2018 (Fersini et al., 2018). Since no improvement was observed in the model, the results are neglected.

model	**masked with**	**weighted F_1**
Multi	Multi_5	**0.710**
Multi	Multi_6	0.588
Uni_{txt}	Uni_{txt} bert-base-uncased	0.660

Table 4: Official weighted F_1-measures for our three submissions to Task B. Column *masked with* specifies the model from Task A used to mask the output labels.

5.2 Task B

Table 4 shows the results of our submitted runs for Task B. In this case, we trained one single Multi model during 5 epochs. The difference between the two configurations is in the masking of the multi-label classification. The most successful multimodal model gets F_1=0.710, after masking with respect to Task A's Multi_5 model. Masking on the basis of Task A's Multi_6 model causes a performance drop of twelve points. Multi_5 predicts more misogynous instances than Multi_6 (678 vs 653). Multi_6 blacks out more predictions which are false positives and hence a potentially correct decision by the multi-label model gets ignored. The text-alone approach, masked by the corresponding Task A model, runs short by five points.

In Table 5 we zoom into the performance of our Task B Multi model for each of the four classes. The model struggles the most when trying to spot stereotyping and shaming. This reflects the nature of misogyny. Stereotyping and shaming tend to be less explicit, and hence harder to spot —even for human beings. On the contrary, violence, which is the most explicit, is more likely to be identified. Stereotyping is the class that has been overpredicted the most (cf. Table 2).

6 Qualitative Analysis

In this section we present a qualitative analysis of the results to further examine the strengths and weaknesses of our approach.

	prec	recall	F$_1$
Shaming	0.52	0.46	0.49
Stereotype	0.54	0.58	0.56
Objectification	0.69	0.66	0.67
Violence	0.73	0.48	0.58

Table 5: Per-class performance on the positive class for model Multi; our best submission to Task B.

	Uni$_{\text{txt}}$	Uni$_{\text{img}}$	Multi
false positives	0.20	0.23	0.21
false negatives	0.14	0.06	0.06
true positives	0.36	0.44	0.44
true negatives	0.30	0.27	0.29

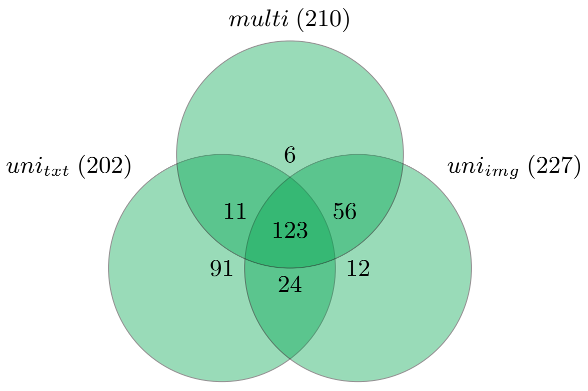
Table 6: Error analysis across all models for Task A showing relative frequencies.

6.1 Analysis on Task A

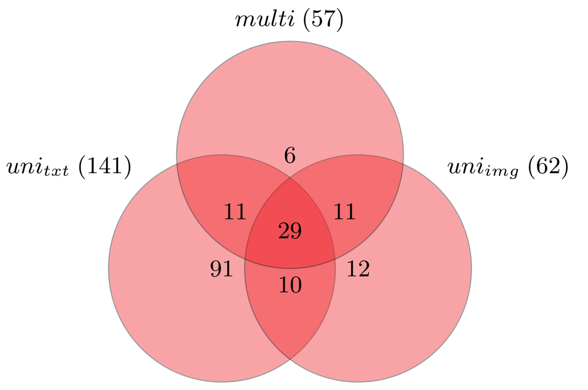
To address the question of which component for detecting misogyny in multimodal settings is more important, we looked at the distribution of the kind of errors made by the different models, as well as the overlapping instances among the four categories. Table 6 shows the relative frequencies. The amount of false negatives is much lower than that of false positives across all models. Considering a practical application, false negatives could have a greater impact as they are the misogynous instances that could not be detected, and could therefore lead to harm. On the other hand, blocking instances that were not misogynous but were classified as such could be considered censorship.

Table 6 shows the prediction analysis of the best runs for each modality. Looking at each model individually, Uni$_{\text{txt}}$ has less false positives but more false negatives than the other two models. Uni$_{\text{img}}$ has the highest number of false positives, and the same number of false negatives as the Multi model. This means that the textual model performs worse than the others in capturing misogyny, while the visual one tends to overpredict misogyny more than the other two models.

Figure 2 shows the intersections and differences in both false positives and false negatives by the three models. There are more false positive than false negative instances across all models, as observed in Table 6. Indeed, the number of common false positives by all models is almost 4 times as high as the number of common false negative values. This indicates that the models tend towards over-predicting misogyny. Taking into account the differences among the sets, Uni$_{\text{txt}}$ accounts for the fewest false positive instances (Figure 2a), while it accounts for the most false negative instances (Figure 2b). Therefore, in this specific multimodal task, where we can be more lenient with false positives than false negatives, a textual model does not seem to be an optimal alternative.



(a) False positives.



(b) False negatives.

Figure 2: Venn diagrams representing the false positive and false negative errors by the three top Multi, Uni$_{\text{txt}}$ and Uni$_{\text{img}}$ models during the testing stage.

Since the model does not allow for a great interpretability of the results, we performed a manual inspection of some interesting instances and the potential reasons behind the errors when classifying them. As Figure 2 shows, 132 instances are misclassified by all three models: 123 are false positives and 29 are false negatives. We observe the following trends after looking at the false negatives:

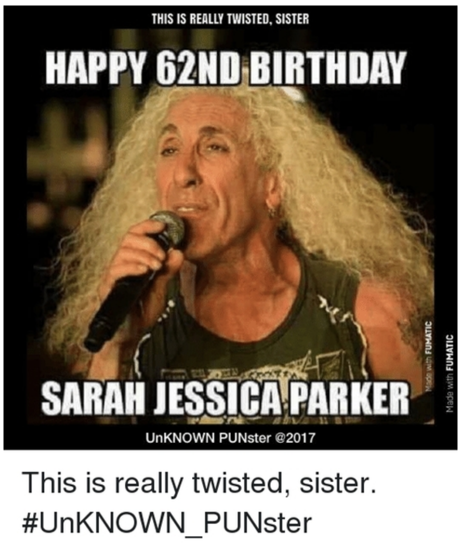
1. The level of misogyny is low or subjective, as the meme is not directly referred to women (e.g., Figure 3a) or misogyny is expressed in a subtle way (e.g., Figure 3b implies the stereotype that women are complicated);

2. Real-world knowledge is required to under-

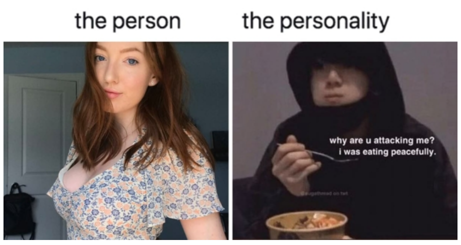
(a) Instance 15846.

My angry girlfriend: "I'm fine"
Me:
I am trying to solve a puzzle.

(b) Instance 16132.



(c) Instance 17028.



(d) Instance 16232.

Figure 3: Instances of Task A false negatives by all three models

stand the meme (Figure 3c can be better understood if we know Sarah Jessica Parker and the Twisted Sister band).

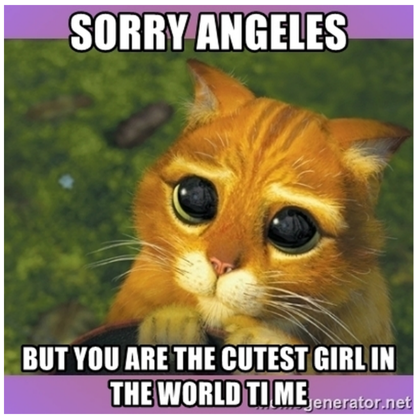


Figure 4: An example of false positive (instance 15094).

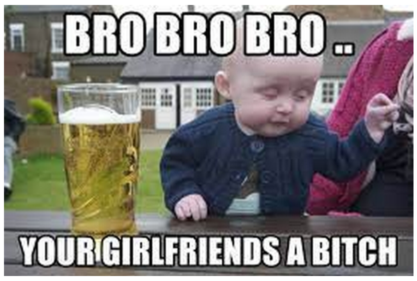


Figure 5: An example of meme properly labeled by text models only (instance 15802).

3. The stance of the text with respect to the image is relevant in order to convey the general meaning (see Figure 3d);

Among the false positives, memes mostly contain:

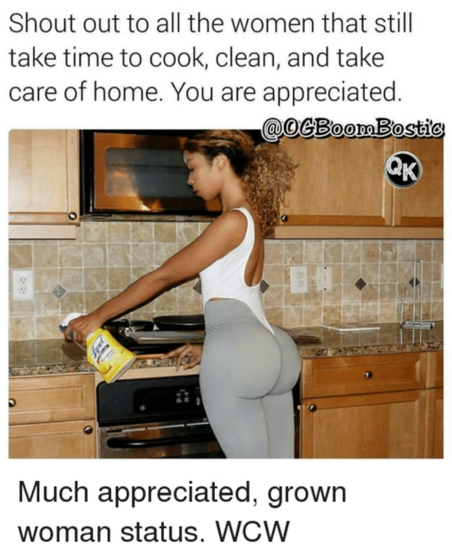
1. Compliments, which are often associated to objectification (e.g. Fig. 4).

2. Images or phrases that often occur in misogynous contents (e.g., women in underwear, kitchen-related terms).

3. Identity terms (e.g., *wife, women, girls*), that tend to co-occur with misogynous contents in the training set.

We also performed an analysis on memes that have been correctly classified by only one model. Among the instances that only the textual model got right, 11 were true positives and 56 true negatives. True positive cases mostly share a strong textual component in conveying misogyny, while the image is either irrelevant, or it is used only to make the sentence ironic (Fig. 5).

Among instances that only the visual model got right, both true positives and true negatives are 11.


Shout out to all the women that still take time to cook, clean, and take care of home. You are appreciated.

Much appreciated, grown woman status. WCW

Figure 6: An example of benevolent sexism that tends to confuse the classifier.

Most true positives have a explicit visual component. For instance, beaten women and texts justifying an aggression or glorifying violence. Among instances that only the multimodal model got right, 10 are true positive and 24 true negative. By observing the true positive instances, contrarily to what is expected, misogyny is not always conveyed by the integration of text and image, as in most of the cases the text is actually dominant.

6.2 Analysis on Task B

We performed a manual inspection focusing on the errors in predicting stereotyping and observed a relatively large amount of compliments toward women, which tend to confuse the classifier. In particular, false negatives are often caused by the presence of *benevolent sexism* (Glick and Fiske, 1997), which shows a subjectively positive attitude towards women that conceals inferiority compared to men, and it is often disguised as a compliment. Figure 6 shows an example.

Now we analyse the label overlaps to determine if our model captured the intersection of the classes. We compare our predictions to the gold labels in Table 2. The size of the intersection between stereotype and objectification is in the same order for gold and predictions: 152 vs 118. The intersection between cases of shaming and violence is practically null, which is well reflected in the model (2 vs 0). Less cases of both shaming and stereotyping than expected are identified (32 vs 20). The same applies to the combinations stereotyping–violence (40 vs 31) and objectification–violence (38 vs 23). The pair shaming–objectification tends to be overpredicted (25 vs 40).

7 Conclusions

We presented our participation to the Multimedia Automatic Misogyny Identification shared task. We addressed two problems: spotting whether a meme is misogynous and, if it is, what kind of misogyny it expresses. We compared unimodal models (text only and image only) with a multimodal model based on Multimodal bi-Transformers. Our image-only model performs better than the text-only one, suggesting that the visual information might be easier to capture than the textual one. Our multimodal approach performs the best in both tasks. The errors come from more false positives than false negatives.

From our error analysis we observed that stereotyping and shaming are the most misclassified categories. This proves that more focus on subtle and implicit forms of misogyny and sexism is needed.

Acknowledgements

A. Muti's research is carried out under project "DL4AMI–Deep Learning models for Automatic Misogyny Identification", in the framework of *Progetti di formazione per la ricerca: Big Data per una regione europea più ecologica, digitale e resiliente*—Alma Mater Studiorum–Università di Bologna, Ref. 2021-15854.

K. Korre's research is carried out under the project "RACHS: Rilevazione e Analisi Computazionale dell'Hate Speech in rete", in the framework of the PON programme FSE REACT-EU, Ref. DOT1303118.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. *arXiv*.

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic Identification and Classification of Misogynistic Language on Twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Alex Clark. 2015. Pillow (pil fork) documentation.

Aaron Defazio and Samy Jelassi. 2021. Adaptivity without Compromise: A Momentumized, Adaptive, Dual Averaged Gradient Method for Stochastic Optimization. *arXiv*.

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

Dimitar Dimitrov, Bishr Bin Ali, Shaden Shaar, Firoj Alam, Fabrizio Silvestri, Hamed Firooz, Preslav Nakov, and Giovanni Da San Martino. 2021. Detecting propaganda techniques in memes. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6603–6617, Online. Association for Computational Linguistics.

Zhida Feng, Jiji Tang, Jiaxiang Liu, Weichong Yin, Shikun Feng, Yu Sun, and Li Chen. 2021. Alpha at SemEval-2021 task 6: Transformer based propaganda classification. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 99–104, Online. Association for Computational Linguistics.

Elisabetta Fersini, Francesca Gasparini, and Silvia Corchs. 2019. Detecting sexist meme on the web: A study on textual and visual cues. In *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*, pages 226–231, Los Alamitos, CA, USA. IEEE Computer Society.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018. Overview of the Evalita 2018 task on automatic misogyny identification (AMI). In *EVALITA Evaluation of NLP and Speech Tools for Italian: Proceedings of the Final Workshop 12-13 December 2018, Naples*, pages 59–66. Torino: Accademia University Press.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. AMI @ EVALITA2020: Automatic misogyny identification. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Peter Glick and Susan T. Fiske. 1997. Hostile and benevolent sexism: Measuring ambivalent sexist attitudes toward women. *Psychology of Women Quarterly*, 21(1):119–135.

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

Vishal Keswani, Sakshi Singh, Suryansh Agarwal, and Ashutosh Modi. 2020. IITK at SemEval-2020 task 8: Unimodal and bimodal sentiment analysis of Internet memes. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1135–1140, Barcelona (online). International Committee for Computational Linguistics.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, Ethan Perez, and Davide Testuggine. 2020a. Supervised Multimodal Bitransformers for Classifying Images and Text. *arXiv preprint arXiv:1909.02950*.

Douwe Kiela, Hamed Firooz, and Aravind Mohan. 2020b. Hateful Memes Challenge and dataset for research on harmful multimodal content. *MetaAI*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Casey A. Fitzpatrick, Peter Bull, Greg Lipstein, Tony Nelli, Ron Zhu, Niklas Muennighoff, Riza Velioglu, Jewgeni Rose, Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, Helen Yannakoudakis, Vlad Sandulescu, Umut Ozertem, Patrick Pantel, Lucia Specia, and Devi Parikh. 2021a. The hateful memes challenge: Competition report. In *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, volume 133 of *Proceedings of Machine Learning Research*, pages 344–360. PMLR.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2021b. The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes. *arXiv preprint arXiv:2005.04790*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. VisualBERT: A Simple and Performant Baseline for Vision and Language. *arXiv preprint arXiv:1908.03557*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. *arXiv preprint arXiv:1908.02265*.

Shana MacDonald and Brianna I. Wiens. 2022. Feminist memes: Digital communities, identity performance and resistance from the shadows. *Materializing Digital Futures: Touch, Movement, Sound and Vision*, page 123. Publisher: Bloomsbury Publishing USA.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection. *arXiv preprint arXiv:2012.10289*.

Arianna Muti and Alberto Barrón-Cedeño. 2020. UniBO @ AMI: A Multi-Class Approach to Misogyny and Aggressiveness Identification on Twitter Posts Using AlBERTo. In *EVALITA Evaluation of NLP and Speech Tools for Italian: Proceedings of the Final Workshop 12-13 December 2018, Naples*.

Rostyslav Neskorozhenyi. 2021. How to get high score using MMBT and CLIP in Hateful Memes Competition. *Towards Data Science*.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Oliver Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. *arXiv 2103.00020*.

Shannon Ridgeway. 2014. 25 everyday examples of rape culture. *Everyday Feminism*.

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Björn Gambäck. 2020. SemEval-2020 task 8: Memotion analysis- the visuo-lingual metaphor! In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 759–773, Barcelona (online). International Committee for Computational Linguistics.

Chhavi Sharma and Viswanath Pulabaigari. 2020. A Curious Case of Meme Detection: An Investigative Study. In *International Conference on Web Information Systems and Technologies (WEBIST)*, pages 327–338.

Amanpreet Singh, Vedanuj Goswami, Vivek Natarajan, Yu Jiang, Xinlei Chen, Meet Shah, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. 2020. MMF: A multimodal framework for vision and language research. `https://github.com/facebookresearch/mmf`.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE-ViL: Knowledge Enhanced Vision-Language Representations Through Scene Graph. *arXiv preprint arXiv:2006.16934*.

Min-Ling Zhang, Yukun Li, Xu-Ying Liu, and Xin Geng. 2017. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, 12:191–202.

Ron Zhu. 2020. Enhance Multimodal Transformer With External Label And In-Domain Pretrain: Hateful Meme Challenge Winning Solution. *CoRR*, abs/2012.08290.

IIITH at SemEval-2022 Task 5: A comparative study of deep learning models for identifying misogynous memes

Tathagata Raha, Sagar Joshi, Vasudeva Varma
IIIT Hyderabad, India
{tathagata.raha, sagar.joshi}@research.iiit.ac.in
vv@iiit.ac.in

Abstract

This paper provides a comparison of different deep learning methods for identifying misogynous memes for SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. In this task, we experiment with architectures in the identification of misogynous content in memes by making use of text and image-based information. The different deep learning methods compared in this paper are: (i) unimodal image or text models (ii) fusion of unimodal models (iii) multimodal transformers models and (iv) transformers further pretrained on a multimodal task. From our experiments, we found pretrained multimodal transformer architectures to strongly outperform the models involving fusion of representation from both the modalities.

1 Introduction

With the social media turning out to be a medium for propagation of hate speech and other perils, misogyny and sexism is incident upon women in explicit and implicit ways. Although memes have turned out to be a potent mechanism for exchanging humorous messages, they have been turning out to also be bearers of such malicious content. With this motivation, the task of Multimedia Automatic Misogyny Identification (MAMI) (Fersini et al., 2022) was proposed with two subtasks: (1) determining whether a meme is misogynous as a binary classification problem (2) finegrained misogyny classification into categories of stereotype, shaming, objectification and violence as a multilabel problem. In our work, we have compared different deep learning approaches for identifying misogyny in memes and also further classifying them into different kinds of misogyny.

We base our experiments on unimodal architectures making use of only either the textual or the image content in memes. The unimodal architectures were naturally superseded by their multimodal counterparts, since they made use of both the modalities in misogyny identification. Among the multimodal architectures, we initially experimented with simple fusion-based approaches which involved combining the image and text representations. These experiments were followed by trying out multimodal transformer architectures in which we made use of MMBT (Kiela et al., 2019), ViLBERT (Lu et al., 2019) and VisualBERT (Li et al., 2019). We used these architectures pretrained on unimodal as well multimodal objectives. We found VisualBERT and ViLBERT trained using multimodal objectives to perform competitively on the task. In order to further improve the capability of the models for misogynous content identification, we tried out further pretraining the models on a dataset for classifying hateful memes. This strategy involving a task-adaptive further pretraining stage turned out to further boost the performance of the models showing the benefit obtained from larger datasets for adapting models to a finegrained downstream task. Figure 1 shows the training stages of such an architecture.

Our best model achieved a macro-F1 score of 0.712 for Subtask 1, while the best performing model for Subtask 2 gave a weighted F1 of 0.706.

2 Related Work

Misogyny detection. Sexism and misogyny has been a long-studied problem, with (Barreto and Ellemers, 2005) and (Dardenne et al., 2007) bringing out the differences in explicit (hostile) and vieled (ambivalent) sexism, with the latter being observed to be subtly undermining and perilous to women. With misogynist remarks - a category of hate speech - being prevalent on social media, the dataset introduced by (Waseem and Hovy, 2016) for hate speech detection on tweets includes sexism as one of the categories in a multiclass problem. In a dataset introduced specifically for misogyny identification on tweets, (Anzovino et al., 2018) also design a taxonomy identifying different manifes-

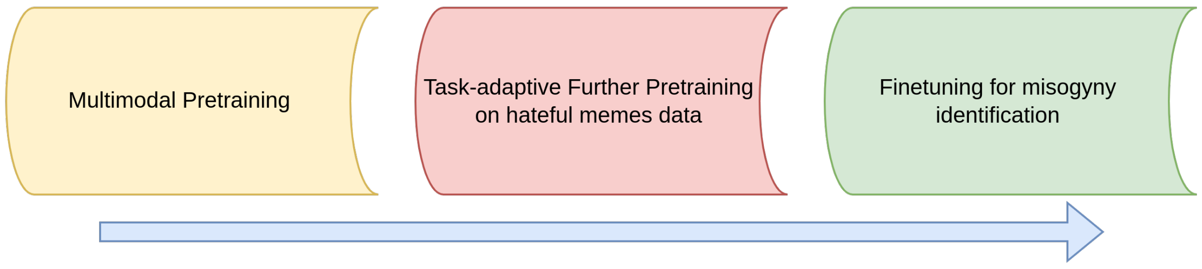

Multimodal Pretraining
Task-adaptive Further Pretraining on hateful memes data
Finetuning for misogyny identification

Figure 1: Stages of training for the model with best performance on misogyny identification

tations of sexism. Focusing on the differences in hostile and benevolent sexism, (Jha and Mamidi, 2017) curated a dataset for classifying the misogyny content in a tweet between the two categories, if the tweet is sexist in nature. Apart from detection of directed hateful content from tweets, there has been work on identifying categories of sexism from personal accounts such as (Karlekar and Bansal, 2018). (Parikh et al., 2019) created a dataset having 23 labeled categories of sexism from sexism accounts without maintaining mutual exclusivity in the categories and proposed a multi-task approach involving three auxiliary tasks for the multilabel classification in (Abburi et al., 2020).

Meme classification. The ubiquity of memes on internet, presence of malicious / hateful content in memes and the challenges involved in meme understanding were discussed in (Sharma et al., 2020). The work presented a new dataset and a challenge for understanding emotions in memes which involved subtasks for identifying the sentiment, humour category and scale (or intensity) of the detected class. (Suryawanshi et al., 2020) introduced a dataset for detection of offensive content in memes. A larger, challenging dataset was introduced in (Kiela et al., 2020) by involving 'benign confounders' to force the multimodal architectures to learn robust representations using both the modalities. The work also introduces formidable baselines with multimodally pretrained transformer encoders. Among the top performing models on this dataset, (Velioglu and Rose, 2020) perform an ensemble of multiple trained VisualBERT models, while ensembling was done in (Muennighoff, 2020) on a set of five predictions from different trained models, with the predictions for each model averaged from 3-5 different runs.

3 Task and dataset overview

The task consists of two subtasks:

- **Course-grained misogyny identification:** For this task, given a meme, we have to predict if a meme is misogynous in nature or not.
- **Fine-grained misogyny identification:** Given that a meme is misogynous, this task further identifies the kinds of misogyny among potential overlapping categories such as stereotype, shaming, objectification, and violence.

The dataset for the task was provided by the workshop organizers. The training set consisted of 10000 memes, whereas the hidden test consisted of 1000 memes. Each row in the dataset contained a unique identifier, the path to the image file for a corresponding meme, the text in the meme, and the label values of misogyny, stereotype, shaming, objectification, and violence.

4 Methodology

In the following section, we discuss our approaches for misogyny detection. We discuss our models in detail and provide a comparison between the models. We have explored unimodal models which use just the text or image as the input. The unimodal fusion models take the representation of the image part of the textual part separately and combine them to give the output. We have also exploited different pretrained multimodal transformers models. We have also experimented with how to further pretraining of these multimodal transformers models affect the quality of the predictions.

4.1 Unimodal models

For unimodal models, we experimented with the following models:

- **Image-Grid:** This is a unimodal image-based classifier that uses convolutional features with average pooling from ResNet-152 (He et al., 2016) architecture.

Setting	Subtask 1 (Macro F1-Score)	Subtask 2 (Weighted F1-Score)
Unimodal-Image-Grid	0.601	0.557
Unimodal-Image-Region	0.606	0.582
Unimodal-Text-BERT	**0.621**	**0.590**
Unimodal-Text-RoBERTa	0.619	0.585
Concat-BERT	**0.648**	**0.611**
Late-Fusion	0.626	0.608
MMBT-Grid	0.651	0.625
MMBT-Region	0.657	0.642
VilBERT	**0.687**	0.671
VisualBERT	0.684	**0.679**
VilBERT CC	0.693	0.683
VisualBERT COCO	0.685	0.689
VilBERT HM	**0.712**	0.698
Visual BERT HM	0.706	**0.702**

Table 1: Results on the testing split for each subtask. Task 1 refers to course-grained identification of misogyny and task 2 refers to the fine-grained identification ofthe types of misogyny.

- **Image-Region** In this unimodal image-based classifier, features from Faster-RCNN (Ren et al., 2015) with ResNeXt-152 (Xie et al., 2016) are used as the backbone network and are pretrained on the Visual Genome dataset (Krishna et al., 2017).

- **Text BERT:** This unimodal text-based approach uses BERT embeddings (Devlin et al., 2018) on the text given as part of the dataset.

- **Text RoBERTa:** Similar to the previous model but it used RoBERTa embeddings (Liu et al., 2019) instead of BERT.

4.2 Unimodal fusions

After taking unimodal representations, we have used the following techniques to fuse the representations to get the final representations before passing to the classifier:

- **Concat-BERT:** In this multimodal approach, an earlier fusion of the output of the unimodal ResNet-152 and BERT embeddings is performed by concatenation, and an MLP is trained for classification.

- **Late Fusion:** This is a simple multimodal approach where the output of ResNet-152 as in Image-Grid and BERT-based models is taken unimodally, and their mean is taken as the final model representation.

4.3 Multimodal transformers

For more advanced models, we have used the following multimodal transformers models:

- **MMBT-Grid:** MMBT (Kiela et al., 2019) is a multimodal supervised bitransformer architecture consisting of individual unimodally pretrained components trained to map multimodal image embeddings to text token space. MMBT-Grid uses features from ResNet-152 for image embeddings.

- **MMBT-Region:** In this approach, the MMBT transformer uses features from Faster-RCNN as in Image-Region for image embeddings.

- **ViLBERT:** ViLBERT (Lu et al., 2019) is a dual-stream multimodal transformer architecture. Here, the VilBERT model without any multimodal pretraining is used. It has BERT initializations for the text stream and uses Faster-RCNN pretrained on Visual Genome dataset to extract image region features.

- **Visual BERT:** Visual BERT (Li et al., 2019) is a multimodal single stream transformer architecture in which the text and image inputs are jointly processed by a stack of BERT-based transformer layers. It uses Faster RCNN for extracting image features.

4.4 Further pretrained models

From the models mentioned in the previous subsection, we have seen VilBERT and VisualBERT perform the best. We move forward with these models to further pretrain them on relevant datasets in a multimodal setting.

- **ViLBERT CC:** ViLBERT architecture used here is pretrained multimodally on the Conceptual Captions (CC) dataset (Sharma et al., 2018) using two pretraining tasks - masked multi-modal modelling (masking 15% of text and image region inputs and reconstructing them with unmasked inputs) and multi-modal alignment prediction (given a pair of image and text, determine if the text describes the image).

- **Visual BERT COCO:** Visual BERT architecture is pretrained multimodally on the Common Objects in Context (COCO) dataset (Lin et al., 2014). The two tasks the model is pretrained on are masked language modeling with an image (some part of the text is masked and is to be predicted using image regions and unmasked text) and sentence-image prediction (given two captions for an image, while one of them is the proper caption for the image, determine if the same holds for the remaining caption as well).

- **VilBERT HM:** For this architecture, we have pretrained the VilBERT architecture on the Hateful Memes dataset (Kiela et al., 2021) with the hypothesis that it will provide better representations given that it has been trained on memes that are hateful in nature. It has been pretrained on masked multi-modal modeling and a new task of meme-caption prediction(where given the image of the meme, the task is to choose the correct text from a given set of options).

- **Visual BERT HM:** Similar to the previous architecture, we have pretrained the Visual BERT model on the Hateful Memes dataset on masked multi-modal modeling task and meme caption prediction.

4.5 Final Setup

After the representation is obtained from any models above, it is passed through a multilayer perceptron classifier to predict the final label. For the second subtask, we used a hierarchical level modeling where the model would predict at first if a meme is misogynous or not, and if it is misogynous, it will perform further fine-grained classification.

4.6 Experimental details

We have used the pretrained models from the MMF framework (Singh et al., 2020) by Facebook AI Research for all of our experiments. We used an 80-20 split to split the dataset into training and validation datasets with a random seed of 42 using sklearn's (Pedregosa et al., 2011) train_test_split functionality. For the hyperparameters, we have used the default hyperparameters of MMF. For subtask 1, we have reported macro F1 score and for subtask 2, we have reported weighted F1.

5 Results

Table 1 contains all the results of our experiment. It can be noted that we mostly used the default hyperparameters from the MMF framework and did not perform rigorous hyperparameter tuning for our experiments, so the model performances still be improved with the search for optimal hyperparameters using cross-validation. We analyze the results of the approaches tried for each subtask. Among the baselines, we saw the unimodal-text models perform better for both the subtasks than the unimodal-image models. Among the unimodal-text models, BERT performed better than RoBERTa. The fusion models performed a bit better than the unimodal models for both the subtasks, with BERT Concatenation performing significantly better than late fusion in the first subtask. The multimodal transformers models give a performance increase over the fusion models, with VilBERT performing the best for Subtask 1 and VisualBERT giving the best performance for the second subtask. Among the multimodal transformer architectures, the ones pretrained with multimodal objectives turned out to be better in performance than those trained using unimodal objectives. Given that VisualBERT and VilBERT performed the best, we further pretrained them in a multimodal task-adaptive setting. VilBERT pretrained on the HatefulMemes dataset gave the best results for the first subtask, whereas VisualBERT pretrained on the HatefulMemes dataset was our best model for the second subtask.

6 Conclusion

In our worked, we have provided a comparative analysis of architectures for solving the task of misogyny identification. Although our best-performing model did achieve a substantial improvement over the baselines, the scores still indicate scope for further improvement. This also brings forth the challenging nature of the task in itself. Furthermore, using ensemble models like Vilio (Muennighoff, 2020) can result in better-performing models which can be tried out in future scope.

References

Harika Abburi, Pulkit Parikh, Niyati Chhaya, and Vasudeva Varma. 2020. Semi-supervised multi-task learning for multi-label fine-grained sexism classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5810–5820, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mary E. Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *NLDB*.

Manuela Barreto and Naomi Ellemers. 2005. The perils of political correctness: Men's and women's responses to old-fashioned and modern sexist views. *Social Psychology Quarterly*, 68(1):75–88.

Benoît Dardenne, Muriel Dumont, and Thierry. Bollier. 2007. Insidious dangers of benevolent sexism: consequences for women's performance. *Journal of personality and social psychology*, 93 5:764–79.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Akshita Jha and Radhika Mamidi. 2017. When does a compliment become sexist? analysis and classification of ambivalent sexism using twitter data. In *Proceedings of the Second Workshop on NLP and Computational Social Science*, pages 7–16, Vancouver, Canada. Association for Computational Linguistics.

Sweta Karlekar and Mohit Bansal. 2018. SafeCity: Understanding diverse forms of sexual harassment personal stories. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2805–2811, Brussels, Belgium. Association for Computational Linguistics.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *arXiv preprint arXiv:1909.02950*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *ArXiv*, abs/2005.04790.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2021. The hateful memes challenge: Detecting hate speech in multimodal memes.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

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

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Niklas Muennighoff. 2020. Vilio: State-of-the-art visio-linguistic models applied to hateful memes. *ArXiv*, abs/2012.07788.

Pulkit Parikh, Harika Abburi, Pinkesh Badjatiya, Radhika Krishnan, Niyati Chhaya, Manish Gupta, and Vasudeva Varma. 2019. Multi-label categorization of accounts of sexism using a neural framework. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1642–1652, Hong Kong, China. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in

Python. *Journal of Machine Learning Research*, 12:2825–2830.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Björn Gambäck. 2020. SemEval-2020 task 8: Memotion analysis- the visuo-lingual metaphor! In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 759–773, Barcelona (online). International Committee for Computational Linguistics.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*.

Amanpreet Singh, Vedanuj Goswami, Vivek Natarajan, Yu Jiang, Xinlei Chen, Meet Shah, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. 2020. Mmf: A multimodal framework for vision and language research. `https://github.com/facebookresearch/mmf`.

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (MultiOFF) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 32–41, Marseille, France. European Language Resources Association (ELRA).

Riza Velioglu and Jewgeni Rose. 2020. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2016. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*.

Codec at SemEval-2022 Task 5: Multi-Modal Multi-Transformer Misogynous Meme Classification Framework

Ahmed Mahran
Codec AI
Alexandria, Egypt
ahmed@codec.ai

Carlo Alessandro Borella
Codec AI
WC1N 2EB, London, UK
carlo@codec.ai

Konstantinos Perifanos
Codec AI
WC1N 2EB, London, UK
kostas@codec.ai

Abstract

In this paper we describe our work towards building a generic framework for both multi-modal embedding and multi-label binary classification tasks, while participating in task 5 (Multimedia Automatic Misogyny Identification) of SemEval 2022 competition.

Since pretraining deep models from scratch is a resource and data hungry task, our approach is based on three main strategies. We combine different state-of-the-art architectures to capture a wide spectrum of semantic signals from the multi-modal input. We employ a multi-task learning scheme to be able to use multiple datasets from the same knowledge domain to help increase the model's performance. We also use multiple objectives to regularize and fine tune different system components.

1 Introduction

In this paper, we present the system that we have built to participate in SemEval 2022 task 5 (Fersini et al., 2022), Multimedia Automatic Misogyny Identification (MAMI) challenge. The task is targeted at identification of misogynous memes by basically using the meme's image and pre-extracted English text content as input sources. The task is divided into two main sub-tasks: Sub-task A is a binary classification task where a meme should be categorized either as misogynous or not misogynous, Sub-task B is a multi-label binary classification task, where the type of misogyny should be recognized among the potential overlapping categories: stereotype, shaming, objectification and violence. Generally, meme classification is a challenging task as memes are multi-modal, rely heavily on implicit knowledge, and are subject to human misinterpretation especially among different backgrounds and cultures.

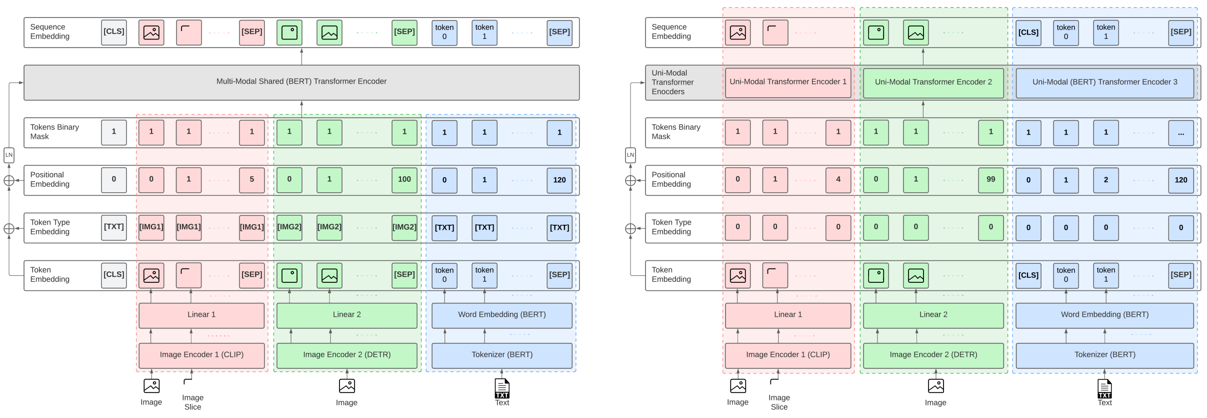
We have used transformer (Vaswani et al., 2017) based architectures and took a transformer based approach to combine them. Transformer based architectures are achieving state-of-the-art performance for Natural Language Processing and Computer Vision related tasks. There are architectures for language, vision, and language and vision combined. There are also architectures for other modalities however in MAMI's scope, we are interested in images and text only.

Pretraining a deep neural network and a transformer based architecture from scratch is a data hungry and computation resources demanding task. Especially in a multi-modal domain where input can have multiple image and text modalities. The literature is rich in a variety of architectures which achieve competitive performance in different tasks for different modalities. In our work, we took an approach towards building a framework that would allow us to combine different pretrained architectures. With relatively few epochs and using relatively less compute resources, our goal was to build and train a classifier framework that could harness the power of pretrained architectures as backbones, using relatively limited resources: no multiple GPUs for training and constrain the cost to be relatively small, in the range of hundred dollars in total.

We have assumed that using as many different backbone architectures which are trained on different tasks for different modalities can allow us to capture a wide spectrum of semantic signals from the input modalities. Then we just need to build a classifier to learn the relationship between these signals and the target classes.

We have also found and discuss below that there are available text and image datasets for hate speech, sexism and hateful memes which sound related to this task and could be used to augment MAMI's dataset. Eventually, our goal was to combine different datasets in a multi-task classifier.

In order to guide the model during training towards the main objective, which is to minimize



(a) Multi-modal shared transformer encoder

(b) Uni-modal multi transformer encoders

Figure 1: Sequence embedding

the classification loss, we have employed different auxiliary objectives. Breaking down the whole architecture into components, each component has a sub-target. We have assumed that if we could assure that each component performs well on the sub-target, then the whole system would perform better on the main target. A component producing a clear signal can facilitate the learning of the downstream dependent components. This could increase model performance in terms of either accuracy or convergence speed. So, if we could formulate an objective function for each component, we can linearly combine them with the main objective function. This helps fine tuning and regularizing the components of the system.

We have built a generic classification framework and applied it to both sub-tasks A and B. During the evaluation phase of the competition, we have achieved, for sub-task A, a macro-average F1-Measure of 0.715, and for sub-task B, a weighted-average F1-Measure of 0.698. During post-evaluation phase we have achieved higher score for sub-task A of 0.761. Our code is available at `https://github.com/ahmed-mahran/MAMI2022`.

2 Background

Datasets: Besides MAMI dataset, there are many datasets that could be used to train our model. The input could be uni-modal as text only or image only, or bi-modal as pairs of image and text. (Vidgen and Derczynski, 2020) reviews 63 publicly available training datasets and they have published a dataset catalogue on a dedicated website [1]. We have used the hateful meme dataset created by Facebook AI (Kiela et al., 2020) which consists of $10K$ memes labeled hateful or not.

Multi-modal frameworks: MMBT (Kiela et al., 2019) concatenates, into a single sequence, linear projection of ResNet (He et al., 2016) output for image pooled to N different vectors, with BERT (Devlin et al., 2018) tokens embeddings for text. The sequence is fed into a transformer encoder, they call it a bi-transformer, after adding positional embeddings and segment embeddings to distinguish which part is image and which part is text. The architecture is generic enough to use different image encoders. As a variant of how we combine signals from image and text, we extend the MMBT architecture to combine more than two encoders however that wasn't our top performing variant. MMCA (Wei et al., 2020) combines Faster R-CNN (Ren et al., 2015) with BERT to compute two embedding types for each modality: self-attention embedding to capture intra-modality interactions, and cross-attention embedding to capture both intra- and inter-modality interactions.

3 System overview

3.1 Tokens embedding

At this stage, we encode each input modality into a sequence of vectors in a unified dimension space. We also generate a binary mask vector with length equal to the sequence's length to indicate which part of the sequence the model should consider. This produces the output at "Token Embedding" and "Tokens Binary Mask" layers illustrated in figure 1. We can use different encoders per modality and generate different sequence types to capture as

[1] hatespeechdata.com

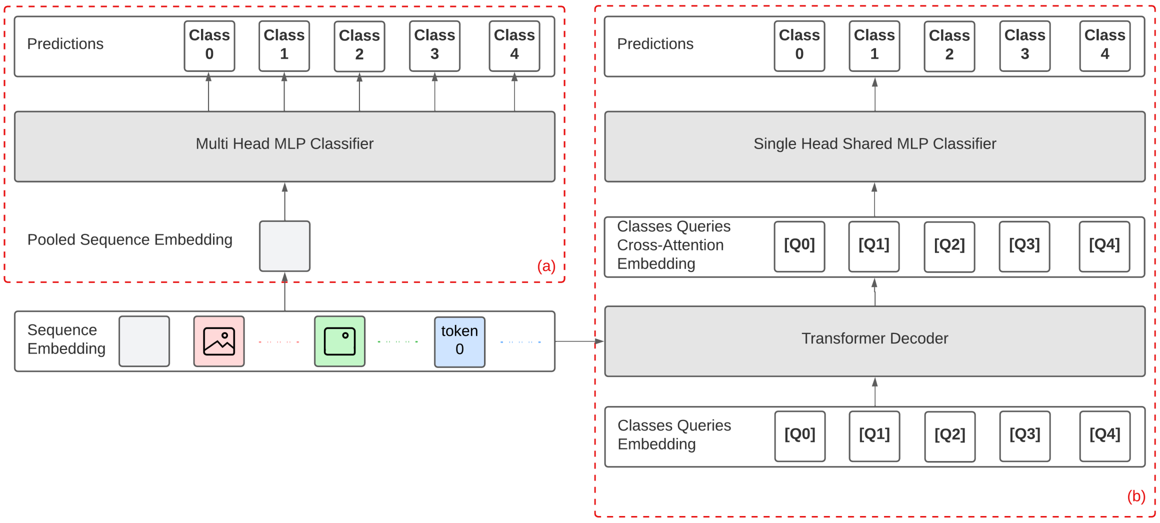

Predictions
Class 0
Class 1
Class 2
Class 3
Class 4
Multi Head MLP Classifier
Pooled Sequence Embedding
(a)
Sequence Embedding
token 0
Single Head Shared MLP Classifier
Classes Queries Cross-Attention Embedding
[Q0]
[Q1]
[Q2]
[Q3]
[Q4]
Transformer Decoder
Classes Queries Embedding
(b)

Figure 2: Classifiers: (a) using pooled sequence embedding, (b) using the whole non-pooled sequence embedding.

many semantic signals as possible. For our setup, we have tested CLIP (Radford et al., 2021) and DETR (Carion et al., 2020) encoders for image modality and BERT (Devlin et al., 2018) encoder for text modality.

CLIP image embedding: We split the image into 4 equal size patches (2×2) then we use CLIP (Radford et al., 2021) to encode the whole image along with its 4 slices into a sequence of 5 vectors. Then we project each vector into the model hidden dimension space. In our experiments, we have used CLIP model named "RN50x4" (which uses as backbone ResNet-50 scaled up 4x using the EfficientNet scaling rule (Tan and Le, 2019))[2].

DETR image objects embedding: We use DETR (Carion et al., 2020) [3] to encode the image into a sequence of 100 image objects representations. DETR's transformer decoder produces a sequence of 100 possible object boxes representations that we use as objects embeddings. However, not all of the objects are real objects as DETR can produce a no-object prediction. So, we use DETR's classifier which is trained on the object detection task to generate masks for no-objects. For each box from the 100, the classifier produces 92 logits which correspond to 92 possible object labels. We take the softmax of the 92 logits and mask out the corresponding object box if the label with the highest softmax probability is the no-object label. We fallback to another masking strategy if all the 100 boxes are masked out; we ignore the logit of the no-object label and then take the softmax of the rest labels to select 4 out of 100 boxes with highest softmax probabilities. Similarly with CLIP output, we project each vector into the model hidden dimension space.

BERT text embedding: We use BERT (Devlin et al., 2018) [4] pre-trained on hate speech (Mathew et al., 2020) to tokenize input text and generate tokens embeddings (for a maximum of 120 tokens). We don't project BERT's embeddings as we use its output space as the model's hidden dimension space (which has length of 768 dimensions).

3.2 Sequence embedding

At this stage, the model generates one combined sequence of vectors using tokens embedding from the tokens' embedding stage. This is the output in the "Sequence Embedding" layer illustrated in figure 1. For the token embedding output, we add token type embedding and positional embeddings that encode the position of each token in the corresponding input sequence per type then we apply a layer normalization (Ba et al., 2016). Along with input masks, the layer normalized sum of embeddings is fed to a multi-layer transformer encoder stage to produce the final sequence embedding at the "Sequence Embedding" layer in figure 1. The output sequence has the same number and dimen-

[2]We have used OpenAI implemantion on `https://github.com/openai/CLIP`

[3]We have used Huggingface Transformers implementation of DETR

[4]We have used Huggingface Transformers implementation of BERT with weights from "Hate-speech-CNERG/bert-base-uncased-hatexplain"

sionality of the input tokens. We have two variants for the transformer encoder stage:

Shared transformer encoder: The general architecture of the shared transformer encoder variant is illustrated in Figure 1a. What distinguishes this variant is that we use a shared multi-layer transformer encoder to capture the intra- and inter-modality interactions. Because of this, for each input embedding type, we add token type embedding which is the same for each input encoder to distinguish which tokens are from which input encoder. In our setup, we have three token types; that is one for each of: CLIP, DETR and BERT encoders. We also append a special [SEP] token at the end of each sequence type and we prepend to the whole combined sequence a global and special [CLS] token. We use the pre-trained BERT transformer encoder as the shared transformer encoder.

Multiple transformer encoders: The general architecture of this variant is illustrated in Figure 1b. The difference here is that instead of using one shared transformer encoder, we use a multi-layer transformer encoder per token type. We still add a token type embedding such that each transformer encoder learns its own token type parameters. We think we can remove this step however we have not tested this. Also, in this variant we don't need to add the extra [SEP] token per type and the global [CLS] token however we add a local [CLS] and [SEP] tokens for BERT only. For CLIP and DETR, we use PyTorch's implementation of the transformer encoder which is described in (Vaswani et al., 2017) with 8 heads and 6 layers but for BERT we keep it as in (Devlin et al., 2018). Then the final output sequence is just the concatenation of all sequences from each transformer encoder.

3.3 Classification

We have two modes of classification depending on the sequence length of the output of the transformer encoder stage. As shown in figure 2, we either use the whole sequence of vectors or pool it to one vector.

Multi-head MLP classifier: We use the pooled sequence embedding as classifier input. In our setup, we have used the first [CLS] token in the shared transformer encoder variant. The classifier is a two feed forward linear layers with a GELU activation in between and a hidden size of 768. The final layer produces number of logits equals to number of classes and we apply a sigmoid activation to compute each binary label probability.

Transformer decoder with single-head shared MLP classifier: Here the whole sequence embedding along with the binary mask is fed into a multi-head transformer decoder as a source sequence. Then the target sequence is formed by a learnable target class query embedding for each class in the target classes. The transformer decoder learns how each class interacts with each input modality signal through the cross-attention mechanism between the source and target sequences. Moreover, the decoder learns the dependency among the target classes through the self-attention mechanism for the target sequence. The decoder output is then fed into a single-head MLP classifier that shares parameters for all classes such that $MLP(q_i)$ is the logit of label i using the corresponding class query embedding, q_i, from the decoder output. The MLP classifier has the same architecture as the previous one in terms of number of layers, type of activation and hidden size, and similarly as well we compute the binary label probability for each class. We use PyTorch's implementation of the transformer decoder which is described in (Vaswani et al., 2017) with 8 heads and 6 layers.

3.4 Multi-task learning

In order to use more training data from other but similar datasets, we have followed a multi-task learning approach. For each dataset $\mathcal{D}$, there is a set of target labels $\mathcal{L}$, we can define as many tasks as the sets of labels in the power set $\mathcal{P}^+(\mathcal{L})$ (excluding the empty set) such that it is possible to use the same label in more than one task. Each task has a separate MLP classifier while all tasks across the datasets share the rest of the parameters including the learnable classes queries. During training, each mini-batch contains data from only one dataset and we compute the targets per task for all the tasks of the dataset.

3.5 Multi-objective

For a training instance i, we use x_i to refer to the input regardless from its actual representation, $y_{i,c} \in \{0, 1\}$ is the value of the target binary label c, and θ is the set of learnable parameters.

Main objective: We use a binary cross entropy to minimize the loss per label.

$$\begin{aligned} \mathcal{L}_0(i,c) = \; & y_{i,c}. \log p(c|x_i,\theta) \\ +(1-y_{i,c}). & \log(1-p(c|x_i,\theta)) \end{aligned} \tag{1}$$

Algorithm 1 Multi-task learning

```
$datasets \leftarrow \{D_1, D_2, ...\}$
$tasksPerD \leftarrow \{(D_1, \{T_1, T_2, ...\}), ...\}$
$labelsPerT \leftarrow \{(T_1, \{l_1, l_2, ...\})\}$
for $epoch \in epochs$ do
    for $b \in mini\text{-}batches$ do
        $dataset \leftarrow$ sample $1$ from $datasets$
        $tasks \leftarrow tasksPerD[dataset]$
        for $task \in tasks$ do
            learn $b$, $task$, $labelsPerT[task]$
        end for
    end for
end for
```

Token encoding projection alignment: We linearly project modal encoding into the hidden model space (as described in section 3.1). In order to preserve a similar structure of data points across spaces, we impose a cosine similarity constraint such that for any two input instances, x_i and x_j, the similarity, $s(.,.)$, between their encoding, $f(.)$, is the same as the similarity between the projection, $g(.)$, of their encoding. We apply this to all pairs in each batch.

$$\begin{aligned}\mathcal{L}_1(i,j) = \ & |s(f(x_i), f(x_j)) \\ & -s(g(f(x_i)), g(f(x_j)))|\end{aligned} \tag{2}$$

Contrastive embedding loss: This is intended at regularizing the embedding space of the MLP classifier to make instances of dissimilar labels more separable. For any two input instances, x_i and x_j, we apply the embedding loss per class c on the input, h^{l-1}, of each layer l of the MLP classifier.

$$\begin{aligned}\mathcal{L}_2(i,j,c) = \\ \frac{1}{2}\sum_l 1 - s_c(y_i, y_j) s_h(h^{l-1}_{i,c}, h^{l-1}_{j,c})\end{aligned} \tag{3}$$

$$s_c(y_i, y_j) = (2y_{i,c} - 1)(2y_{j,c} - 1) \tag{4}$$

$s_c(.,.) \in \{-1, 1\}$ is the labels similarity for class c of two instances; -1 indicates dissimilar labels while 1 indicates similar labels. $s_h(.,.) \in [-1, 1]$ is the cosine similarity of layer input when comparing two instances. It is worth noting that in case of the multi-head MLP classifier, $h^{l-1}_{i,c}$ is the same for all classes. We also apply this loss to all pairs in each batch. This loss encourages the transformer encoder in case of the multi-head MLP classifier, the decoder in case of the shared single head MLP classifier, as well as the hidden layers of the MLP classifier to produce embeddings with structures that capture labels similarity such that instances with the same label value get closer embeddings than instances with different label value.

The overall loss per batch and task given a dataset d:

$$\begin{aligned}\mathcal{L}^d_t = & \frac{1}{N_i N^t_c}\sum_{i,c}\mathcal{L}_0(i,c) \\ & + \frac{1}{N_i(N_i-1)}\sum_{i \neq j}\mathcal{L}_1(i,j) \\ & + \frac{1}{N_i(N_i-1)N^t_c}\sum_{i \neq j, c}\mathcal{L}_2(i,j,c)\end{aligned} \tag{5}$$

The overall loss per batch for all tasks of the dataset is the average task loss:

$$\mathcal{L}^d = \frac{1}{N^d_t}\sum_t \mathcal{L}^d_t \tag{6}$$

N_i is the batch size, N^d_t is the number of tasks for dataset d, and N^t_c is the number of classes for task t.

4 Experimental setup

In addition to MAMI's dataset, we have used Facebook's hateful memes dataset. We have set the tasks configurations as shown in table 1.

Dataset	Task	Labels
MAMI	MAMI	{misogynous, shaming , stereotype , objectification, violence}
	Task_A	{misogynous}
	Task_B	{shaming, stereotype , objectification, violence}
FBHM	Hateful	{hateful}

Table 1: Tasks labels configurations per dataset. FBHM is short for Facebook Hateful Meme.

We have added the redundant task MAMI for the MAMI dataset to make sure that the decoder learns the dependency among all labels as Task_B's labels depend on Task_A's label we wanted to make sure that the model captures this dependency.

We have split both datasets into 80% train and 20% dev sets using stratified sampling. We have used the data provided at post-evaluation period of the competition as the test set. We have used a batch size of 16 and number of epochs as 15. We have used MADGRAD (Defazio and Jelassi, 2021) for optimization. Learning rate was set to 2×10^{-4} and we used a learning rate linear scheduler with a warmup period [5]. Gradients are accumulated every 20 batches so there was a total gradient accumulation steps of: total number of batches$/20 \times$ number of epochs. We have set the warmup period to: total gradient accumulation steps$/10$. For all parameters except biases and layer normalization weights, we have used a weight decay of 5×10^{-4}. We clip gradients to overall norm of 0.5. Our implementation is PyTorch based and we have used HuggingFace Transfomers implementation for both BERT and DETR. We have run our experiments on Google Colab Pro plus using one Tesla P100 GPU. We didn't perform any special data preprocessing, just the requirements for each backbone. Also, to make experiments quicker, we didn't perform any fine tuning for any of the backbones and we pre-generated and stored each backbone output instead of re-evaluating the same data across epochs and experiments.

We used the official accuracy measures to score how the model performs on each task. For the single class tasks, we have used macro-average F1-Measure and we called it scoreA. In particular, for each class label (i.e. true and false) the corresponding F1-Measure will be computed, and the final score will be estimated as the arithmetic mean of the two F1-Measures. For the multi class tasks, we have used weighted-average F1-Measure and we called it scoreB. In particular, the F1-Measure will be computed for each label and then their average will be weighted by support, i.e. the number of true instances for each label.

5 Results

5.1 Ablations

We have the following system configurations [6] which would result in different architecture variations.

Transformer encoder (Xformer Enc) whether to use shared transformer (**Shared**) or multi transformers (**Multi**).

Encoder output pooling (Pooling) whether the whole sequence is pooled using [CLS] embedding (**[CLS]**), this means we use the multi-head MLP classifier and no decoder), or no pooling (**No**) or only text tokens are pooled using text's [CLS] token (**txt [CLS]**), this means we use the decoder with the shared single head MLP classifier.

Token encoding projection alignment (Proj Align) whether to enable it (**Yes**) or not (**No**).

Contrastive embedding loss (Contrastive) whether to enable it (**Yes**) or not (**No**).

Multi-task learning (Multi-task) whether to use Facebook's hateful meme dataset (**Yes**) or not (**No**).

Image encoders (Backbones) whether to use CLIP only (**CLIP**), DETR only (**DETR**), or both together (**CLIP and DETR**).

We plan experiments as tournament of rounds such that in each round we test subset of the configurations fixing the rest. Then we use the winning configuration values for subsequent rounds. For each experiment, we report the max score for Task_A and Task_B on all splits. Appendix A contains more details on scores distributions.

5.1.1 Round 1

At this round we compare transformer encoder and encoder output pooling methods. We perform four experiments with configurations and results sumaized in table 2. The winner of this round is the multi-transformers encoders without output pooling architecture variant.

Experiment	00	01	02	03
Xformer Enc	Shared	Shared	Multi	Multi
Pooling	[CLS]	No	No	txt [CLS]
Proj Algn	No			
Contrastive	No			
Multi-task	No			
Backbones	CLIP and DETR			
Score				
Test - Task_A	0.6819	0.7226	**0.7436**	0.7329
Test - Task_B	0.5886	0.6422	**0.6785**	0.6772
Dev - Task_A	0.8395	0.8355	**0.8564**	0.8519
Dev - Task_B	0.6650	0.6933	**0.7420**	0.7310
Train-Task_A	**0.9253**	0.9121	0.9066	0.8998
Train-Task_B	0.7006	0.7242	**0.7782**	0.7647

Table 2: Experiments configurations for round 1 and corresponding results.

5.1.2 Round 2

At this round, we test the significance of the multi-objective approach. The configurations and corresponding results are summarized in table 3. To-

[5]We used get_linear_schedule_with_warmup from Huggingface Transformers.

[6]We give each item a short name in parenthesis to be able to refer to corresponding items in experiments results tables compactly.

ken encoding projection alignment makes improvements on both tasks on the test split when enabled alone. Also, contrastive embedding loss seems to slightly improve Task_B. After performing statistical tests and comparing distributions of the scores, we pick experiment 10 as the winner variant.

Experiment	02	10	12	13
Xformer Enc	Multi			
Pooling	No			
Proj Algn	No	Yes	No	Yes
Contrastive	No	No	Yes	Yes
Multi-task	No			
Backbones	CLIP and DETR			
Score				
Test - Task_A	0.7436	**0.7504**	0.7358	0.7467
Test - Task_B	0.6785	0.6798	**0.6823**	0.6777
Dev - Task_A	**0.8564**	0.8535	0.8515	0.8535
Dev - Task_B	**0.7420**	0.7387	0.7371	0.7313
Train-Task_A	0.9066	0.8983	**0.9090**	0.8988
Train-Task_B	**0.7782**	0.7679	0.7676	0.7573

Table 3: Experiments configurations for round 2 and corresponding results compared to best configurations from round 1.

5.1.3 Round 3

At this round, we test the significance of the image encoders backbones, namely: CLIP and DETR. The configurations and corresponding results are summarized in table 4. It seems that our use of DETR was incompetent to CLIP.

Experiment	10	20	21
Xformer Enc	Multi		
Pooling	No		
Proj Algn	Yes		
Contrastive	No		
Multi-task	No		
Backbones	CLIP and DETR	CLIP	DETR
Score			
Test - Task_A	**0.7504**	0.7426	0.7010
Test - Task_B	0.6798	**0.6865**	0.6306
Dev - Task_A	0.8535	**0.8560**	0.7979
Dev - Task_B	**0.7387**	0.7347	0.6784
Train-Task_A	0.8983	**0.8990**	0.8186
Train-Task_B	**0.7679**	0.7628	0.6704

Table 4: Experiments configurations for round 3 and corresponding results compared to best configurations from round 2.

5.1.4 Round 4

At this round, we test the significance of the additional training data from Facebook's hateful meme dataset. The configurations and corresponding results are summarized in table 5. This time we train for more 15 epochs (i.e. total 30 epochs). We can notice a significant improvement when using more training data from the external dataset.

Experiment	10	30
Xformer Enc	Multi	
Pooling	No	
Proj Algn	Yes	
Contrastive	No	
Multi-task	No	Yes
Backbones	CLIP and DETR	
Score		
Test - Task_A	0.7504	**0.7609**
Test - Task_B	0.6798	**0.6958**
Dev - Task_A	**0.8535**	0.8502
Dev - Task_B	0.7387	**0.7429**
Train-Task_A	0.8983	**0.9127**
Train-Task_B	0.7679	**0.7815**

Table 5: Experiments configurations for round 4 and corresponding results compared to best configurations from round 2.

5.2 Visualizations

In figure 3, we show t-SNE projections of the transformer decoder output per class and the corresponding class learnt input query embedding. We use data from experiment 30 in section 5.1.4. It is interesting that, for all classes, the class learnt query embedding is positioned on the side with denser positive labels. This indicates that the learnt class queries can be thought of as centers of positive labels.

In figure 4, we show average attention weights per transformer decoder layer. Figure 4a illustrates the dependencies between each class and other classes. As shown in the figure, numbers are very close which indicates that each class depends uniformly on other classes. Figure 4b shows the average cross-attention weights between the source and target sequences from the transformer decoder layers and aggregated per input encoder. Clearly, the model pays more attention to CLIP features, and less attention to BERT text features, and the least attention to DETR objects features. This conforms with results from round 3 in section 5.1.3.

6 Conclusion

In this paper, we propose a generic framework for both multi-modal embedding and multi-label binary classification tasks. We combine and use as backbones different architectures achieving state-of-the-art in different or similar tasks on different modalities to capture a wide spectrum of semantic signals from the multi-modal input. By employing a multi-task learning scheme, we are able to use

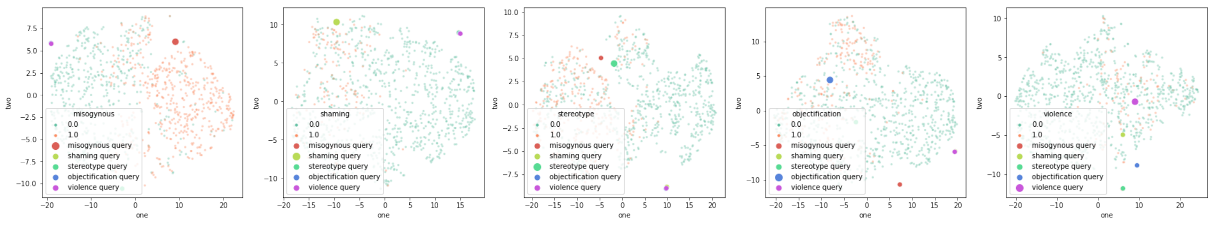
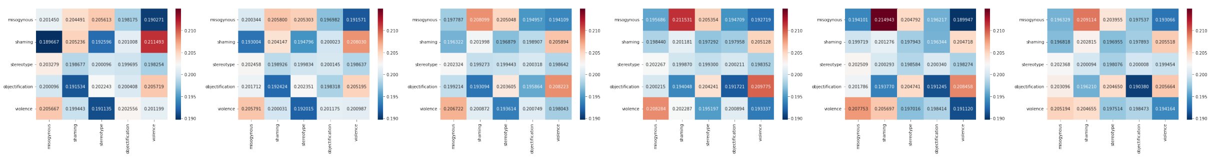
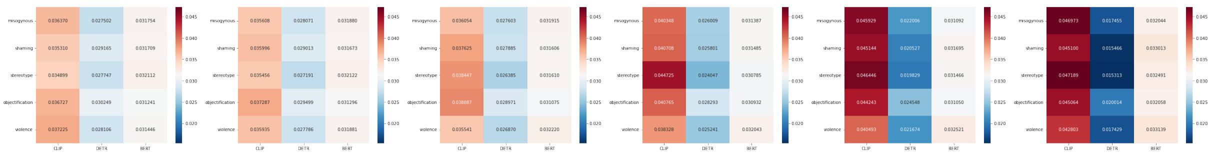


Figure 3: t-SNE projections of transformer decoder output per class (small dots) and the corresponding class query embedding (the biggest dot).



(a) Transformer decoder average self-attention weights



(b) Transformer decoder average source $\times$ target cross-attention weights. Source weights are aggregated per input encoder.

Figure 4: Attention weights visualization per transformer decoder layer. First input layer is on the left while last output layer is on the right.

multiple datasets from the same knowledge domain and increase the model's performance. In addition to that, we use multiple objectives to regularize and fine tune the system components. We have carried out experiments to verify our ideas and the results show the significance of some of the ideas. As a future work, we need to do more experiments with different backbone architectures and more datasets. We can also try more objectives and regularizations; for instance, we can use our observation from figure 3 to make the decoder output for a positive instance closer to the corresponding class query embedding.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Aaron Defazio and Samy Jelassi. 2021. Adaptivity without compromise: a momentumized, adaptive, dual averaged gradient method for stochastic optimization. *arXiv preprint arXiv:2101.11075*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, Ethan Perez, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *arXiv preprint arXiv:1909.02950*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection. *arXiv preprint arXiv:2012.10289*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Bertie Vidgen and Leon Derczynski. 2020. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *Plos one*, 15(12):e0243300.

Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. 2020. Multi-modality cross attention network for image and sentence matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10941–10950.

A Experiments scores distributions

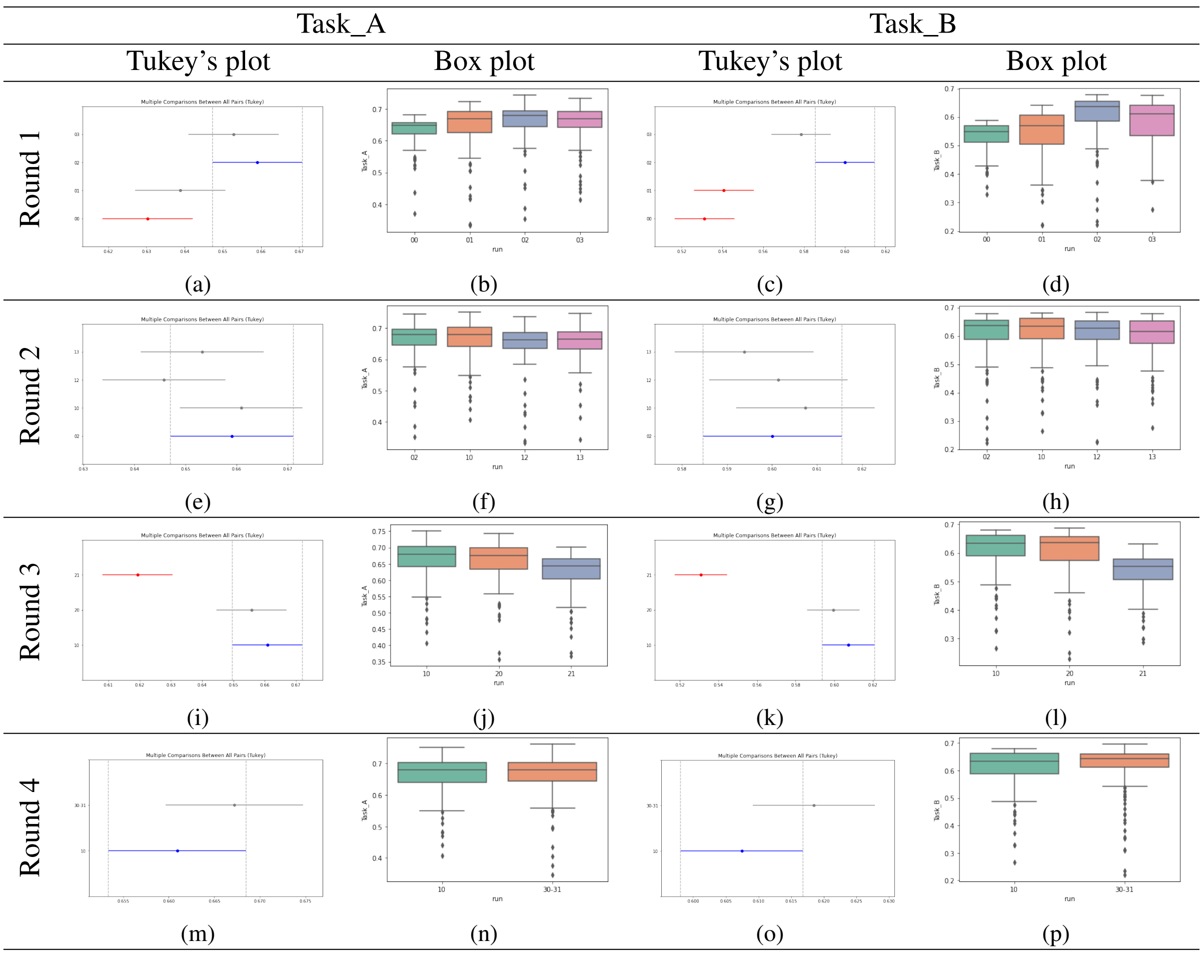


Figure 5: Performance comparison on test split for Task_A and Task_B collected from evaluation phases during model training epochs for different experiments described in 5.1. Tukey's plots visualize a universal confidence interval of scores mean on x-axis for each run on y-axis, any two runs can be compared for significance by looking for overlap. Box plots summarize the distribution of scores on y-axis for each run on x-axis.

I2C at SemEval-2022 Task 5: Identification of Misogyny in Internet Memes

Pablo Cordón Hidalgo
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
pablo.cordon113@alu.uhu.es

Pablo Gonzalez Díaz
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
pablo.gonzalez682@alu.uhu.es

Jacinto Mata Vázquez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
mata@uhu.es

Victoria Pachón Álvarez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
vpachon@uhu.es

Abstract

In this paper we present our approach and system description on *Task 5 A in MAMI: Multimedia Automatic Misogyny Identification.* In our experiments we compared several architectures based on deep learning algorithms with various other approaches to binary classification using Transformers, combined with a nudity image detection algorithm to provide better results. With this approach, we achieved a test accuracy of 0.665.

1 Introduction

Misogyny is hatred or contempt for women. It is a form of sexism used to keep women at a lower social status than men, thus maintaining the societal roles of patriarchy. Misogyny has been widely practiced for thousands of years. It is reflected in art, literature, human societal structures, historical events, mythology, philosophy, and religion worldwide (Manne, 2017).

The Internet represents for many an extension of our offline interactions, and seemingly mundane everyday practices (e.g. participating in social media) form a significant part of our everyday experiences. Unfortunately, it is too common to see examples of harassment towards women and marginalized groups online within these experiences and practices (Drakett et al., 2018).

Women have a solid presence on the web, especially in picture-based web media like Twitter and Instagram: 78% of females utilize online media on numerous occasions each day, in contrast with 65% of men.

A popular way of communicating via social media platforms are MEMEs. A meme is an image portrayed through pictorial content with overlaid text which is written a posteriori, with the fundamental objective of being entertaining and/or ironic. Even though most of memes are created with the goal of making amusing jokes, shortly after their standardization individuals began to use them to disseminate hate against women, leading to sexist and aggressive messages in internet environments that allow people to freely express sexism without the fear of retaliation.

The detection of this disrespectful content is essential to eliminate it as soon as possible and stop spreading misogyny as a "joke".

In MAMI: Task 5, Track A (meme binary classification) (Fersini et al., 2022), participants must determine whether a meme (text + image) is misogynist or not.

Meme sentiment-related tasks analysis is challenging, as memes are used created for various purposes, they are always evolving and often use sarcasm and humour. While misogyny and hate speech detection in text has been widely explored by the NLP community (Badjatiya et al., 2017). Its detection in images and text and how they correlate has not been explored in depth.
In this field we used BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2021) for training and classifying text, and nudenet tool for image classification (https://github.com/notAI-tech/NudeNet).

Our results show that combining text and image classification results are slightly better than using only one of the two methods.

The rest of the paper is organized as follows. Section 2 contains a briefly description of the dataset and its structure, Section 3 features the analysis of some of the previous works related to our task. In section 4 we describe the different models and algorithm used, and their configurations. Section 5 provides details about data and models setup, while Section 6 reports experimental results and the paper is concluded in Section 7.

2 Background

This paper is focused on subtask A: Binary classification. The corpus provided is composed of 5000 1-value rows (misogyny) and 5000 0-value rows (not misogyny), so it is already well-balanced. Each row has the following format:

file_name | misogynous | shaming | stereotype | objectification | violence | text transcription

Where "file_name" is the .jpg link to the image, and "text transcription" is the text extracted from the image.

For our binary classification task, we only need the fields "file_name", "misogynous" and "text transcription" to train the models presented. For models that also used nudity recognition, a column "unsafe" was added later.

The csv file used follows this structure:

- file_name: *"10.jpg"*
- misogynous: "*1*"
- text transcription: "*ROSES ARE RED, VIOLETS ARE BLUE IF YOU DON'T SAY YES, I'LL JUST RAPE YOU quickmeme.com*"

3 Related work

Sentiment analysis of text is a very active research area that still faces multiple challenges such as irony and humour detection (Farías et al., 2016). In this area, the focus of the NLP community has increased towards detection of offensive language, aggression, hate-speech detection (Wei et al., 2021) and specifically misogyny, taking into account it can be expressed in a direct, explicit manner or an indirect, sarcastic manner, and even if this message is genered or not-gendered (Samghabadi et al., 2020).

The analysis of misogyny in memes has already been done in a psychologic point of view (Drakett et al., 2018), but never in computational models. Multimodal analysis research has been extended during the last years, but the focus was mostly on Video and text or speech and text (Pozzi et al., 2016). The specific multi-modality of memes in sentiment analysis has only been addressed recently by investigating their correlation with other comments in online discussions (French, 2017).

The growing usage of memes as an alternative medium of communication on social media has also recently drawn the attention of the online abuse research community.

However, memes completely make sense only if one takes both text and image content into account. These modalities can also lead to totally different perceived sentiment when recombined. For example, a meme whose image is a scary clown and the text is "happy birthday" will have a very different sentiment from a meme with the same text but with an image of a funny clown.

Sabat et al. (2019) performed hate speech detection on memes and showed that images were more important than text for the prediction.

In the other hand, Bonheme and Grzes (2020), investigated the relationship of text and image in sentiment analysis of memes, and found that images and text were uncorrelated. Fusion-based strategies did not show significant improvements and using one modality only (text or image) tends to lead to better results.

4 System overview

We focus on exploring different training techniques for text using BERT and RoBERTa, given their superior performance on a wide range of NLP tasks, while for image we used the python module *nudenet*.

Each text encoder, image classifier and training method used in our model are detailed below.

4.1 Text Encoders

BERT (Devlin et al., 2018): pretrained model BERT-base uncased, released by the authors, was used as embedding layer, tokenizer and classifier. It consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.

RoBERTa (Liu et al., 2021): We use the RoBERTa-base model released by the authors. Like BERT, RoBERTa-base consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.

4.2 Image nudity classification

As memes are mostly done as "joke" and tend to be ironical and use a very refined and deep text-image relationship.

A similar approach to the one used by Messina et al. (2021) was applied, where one of the two modalities acts as the main one and the second intervenes to enrich the first (in our case, text will act as the main modality and image will be used just to enrich the results from the first one).

The goal of this task was to detect misogyny, and we decided to use a Not Safe For Work (NSFW) image classifier.

Nudenet is the classifier used for this task. It gives each image of our dataset an "unsafe" value from 0 (safe) to 1 (unsafe). The Neural Net for Nudity Classification is trained on 160,000 entirely auto-labelled (using classification heat maps and various other hybrid techniques) images.

A NSFW image classifier was used for two main reasons. First, because image-only classification using Convolutional Neural Networks did not reached good results using the training data. We only obtained an accuracy of 0.52. Second, because most of NSFW images are misogynous, as it could be demonstrated by using only the *Nudenet* classifier, obtaining a value of 0.83 for precision in the positive (misogynous) class.

4.3 Models

Based on Convolutional Neural Networks (Konda et al., 2019), BiLSTM Neural Networks (Zhou et al., 2016), and BERT Transformers (Devlin et al., 2018), several models have been developed: (1) BiLSTM Neural Network with RoBERTa as embedding, (2) BiLSTM Neural Network with BERT as embedding, (3) BiLSTM Neural Network with BERT as embedding and nude detection, (4) 1- Dimensional Convolutional Neural Network with BERT Tokenizer, and (5) BERT Transformer with nude detection.

In our models, RoBERTa and BERT were used with word-embedding strategies, as they have an advantage over models like Word2Vec. Each word, under Word2Vec, has a fixed representation regardless of the context within which the word appears. Nevertheless, BERT produces word representations that are dynamically informed by the words around them (Shi and Lin 2019a).

For example, given the two sentences *"The man was director of a bank in his hometown."* and *"The man went fishing by the bank of the river."*, Word2Vec would produce the same word embedding for the word "bank" in both sentences. However, BERT will create different word embedding for "bank" for each sentence.

4.3.1 BiLSTM Neural Network with RoBERTa as embedding

Long Short-Term Memory (LSTM), (Hochreiter and Schmidhuber, 1997) is a widely known recurrent neural network (RNN) architecture. We used Bidirectional LSTM (Schuster and Paliwal, 1997) models for our experiments. A Bidirectional LSTM (BiLSTM) layer processes the text both in the forward as well as backward direction and hence is known to provide better context understanding.

Introduced by Facebook, the Robustly optimized BERT approach RoBERTa, is a retraining of BERT with an improved training methodology, 1000% more data and compute power (Shi and Lin, 2019b).

The RoBERTa model used to extract the word embedding layer for the BiLSTM Neural Network was RoBERTa-base uncased.

4.3.2 BiLSTM Neural Network with BERT as embedding

For this approach, a BERT-based model was used. In particular, we implemented the BERT-base uncased model.

4.3.3 BiLSTM Neural Network with BERT as embedding + nude detection

Python's Nudenet module was used in every image of the given dataset to assign it a value of "unsafety" from 0 (safe) to 1 (unsafe). Then, if the text prediction is 1, the final prediction is set as 1, otherwise if the text prediction is 0 and the image unsafe value is greater than a threshold value, final prediction is set to 1.

The best values for this threshold were the ones with the best accuracy classifying using with nudity in images. These threshold values were 0.45 and 0.60.

4.3.4 1-Dimensional Convolutional Neural Network with BERT as embedding + nude detection

Convolutional neural networks (CNN) (Lecun et al., 1998) are originally designed to process and learn information from image features by applying convolution kernels and pooling techniques which are widely adopted for extracting stationary features; for instance, CNN has shown its adaptability in the field of text mining and NLP tasks. (Kim, 2014) reported series of experiments with CNNs that achieve good results on sentence classification and sentiment analysis tasks.

To improve this model classification, BERT-base uncased model was used to create the word embedding layer. The nudity classification algorithm with a threshold of 0.45. was used to achieve better results.

4.3.5 BERT Transformer with nude detection

Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based machine learning technique for natural language processing (NLP) pre-training developed by Devlin et al. (2018). BERT model is pre-trained from unlabeled data extracted from the BooksCorpus with 800M words and English Wikipedia with 2,500M words.

It uses Transformer, (Vaswani et al., 2017) an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary.

Also, the nudity classification algorithm with a threshold of 0.45. was used to achieve better results.

5 Experimental setup

For each text transcription row in the corpus, a small preprocessing was applied. Every word was undercased, web pages' links, hashtags, usernames, emojis, punctuation symbols, numbers, and words with length less than 2 characters were removed, as well as English stop words using nltk.corpus module (Sarica, 2021).

For all the experiments, we split the training dataset in two parts: 80% for training and 20% for validation using a stratify approach.

The parameters used in the training phase were: batch size of 32 and 5 epochs.

6 Results

Table 1 shows a summarization of the training and test results obtained for each one of the models in the evaluation phase.

According to the official metrics (F1-score for the positive class), our results are all around 0.60 – 0.65, being the best model BiLSTM Neural Network with BERT as embedding + nude detection, with a 0.665 F1-score. With this result, we obtained the 43 place in the ranking.

As we expected, the model that obtained the best results during the training phase also obtained the best result in the evaluation phase.

7 Conclusions

In this paper, several approaches and systems descriptions on Task 5 (Subtask A) in SemEval 2022: Multimedia Automatic Misogyny Identification are detailed. The main aim was to develop various deep learning models and check how multi-modality of text and image could help achieve better classification results.

Six different models were developed and BiLSTM Neural Network with BERT as embedding + nude detection (0.45 threshold) was the definitive one. After training and analyzing each model, we achieved an F1-score of 0.665 in

Model	**Training phase**		**Evaluation phase**
	Accuracy	**F1 - Score**	**F1 - Score**
BiLSTM Neural Network with RoBERTa as embedding	0.793	0.799	0.607
BiLSTM Neural Network with BERT as embedding	0.810	0.832	0.652
BiLSTM Neural Network with BERT as embedding + nude detection threshold 0.45	**0.837**	**0.840**	**0.665**
BiLSTM Neural Network with BERT as embedding + nude detection threshold 0.6	0.835	0.838	0.663
1-D Convolutional Neural Network with BERT as embedding + nude detection	0.813	0.825	0.649
BERT Transformers + nude detection	0.806	0.812	0.639

Table 1: Results obtained with the different models

the evaluation phase for class “1”. We can conclude that merging text and image classifiers improves the results in the task of misogyny detection in memes.

In future works, we intend to improve our image classifiers models. Also, we want to use other pretrained models based on transformers.

References

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. "Deep Learning for Hate Speech Detection in Tweets." Perth, Australia, International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/3041021.3054223.

Lisa Bonheme and Marek Grzes. 2020. *SESAM at SemEval-2020 Task 8: Investigating the Relationship between Image and Text in Sentiment Analysis of Memes* International Committee for Computational Linguistics. doi:10.18653/v1/2020.semeval-1.102.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." *CoRR* abs/1810.04805.

Jessica Drakett, Bridgette Rickett, Katy Day, and Kate Milnes. 2018. "Old Jokes, New Media – Online Sexism and Constructions of Gender in Internet Memes." *Feminism Psychology* 28 (1): 109-127. https://journals.sagepub.com/doi/full/10.1177/0959353517727560.

Delia Farías, Viviana Patti, and Paolo Rosso. 2016. "Irony Detection in Twitter." *ACM Transactions on Internet Technology* 16 (3): 1-24. http://dl.acm.org/citation.cfm?id&#61;2930663

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics.

Jean H French. 2017. "Image-Based Memes as Sentiment Predictors.". doi:10.23919/i-Society.2017.8354676.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9: 1735-1780. doi:10.1162/neco.1997.9.8.1735.

Yoon Kim. 2014. *Convolutional Neural Networks for Sentence Classification*.

Srinivas Konda, B. Rani, Varaprasad Mangu, G. Madhukar, and B. Ramana. 2019. "Convolution Neural Networks for Binary Classification." *Journal of Computational and Theoretical Nanoscience* 16: 4877-4882. doi:10.1166/jctn.2019.8399.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE* 86 (11): 2278-2324. doi:10.1109/5.726791.

Zhuang Liu, Wayne Lin, Ya Shi, and Jun Zhao. 2021. "A Robustly Optimized BERT Pre-Training Approach with Post-Training." In *Chinese Computational Linguistics*, 471-484. Cham: Springer International Publishing. https://library.biblioboard.com/viewer/822f5f9f-f7f4-11eb-926c-0a9b31268bf5.

Kate Manne. 2017. *Down Girl: The Logic of Misogyny*. New York: Oxford University Press. https://oxford.universitypressscholarship.com/10.1093/oso/9780190604981.001.0001/oso-9780190604981.

Nicola Messina, Fabrizio Falchi, Claudio Gennaro, and Giuseppe Amato. 2021. "AIMH at SemEval-2021 Task 6: Multimodal Classification using an Ensemble of Transformer Models. "Association for Computational Linguistics https://aclanthology.org/2021.semeval-1.140.

F. Pozzi, E. Fersini, E. Messina, and B. Liu. 2016. *Sentiment Analysis in Social Networks* Elsevier Science. https://books.google.es/books?id=aS2lCgAAQBAJ.

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i-Nieto. 2019. "Hate Speech in Pixels: Detection of Offensive Memes Towards Automatic Moderation." *arXiv Preprint arXiv:1910.02334*.

Niloofar Safi Samghabadi, Parth Patwa, Srinivas PYKL, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. "Aggression and Misogyny Detection using BERT: A Multi-

Task Approach." European Language Resources Association, may. https://aclanthology.org/2020.trac-1.20.

Serhad Sarica and Luo,Jianxi. 2021. "Stopwords in Technical Language Processing." *Plos One* 16 (8): 1-13. https://doi.org/10.1371/journal.pone.0254937.

Mike Schuster and Kuldip Paliwal. 1997. "Bidirectional Recurrent Neural Networks." *Signal Processing, IEEE Transactions On* 45: 2673. doi:10.1109/78.650093.

Peng Shi and Jimmy Lin. 2019a. *Simple BERT Models for Relation Extraction and Semantic Role Labeling* https://arxiv.org/abs/1904.05255.

———. 2019b. "Simple BERT Models for Relation Extraction and Semantic Role Labeling." http://arxiv.org/abs/1904.05255.

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

Bencheng Wei, Jason Li, Ajay Gupta, Hafiza Umair, Atsu Vovor, and Natalie Durzynski. 2021. "Offensive Language and Hate Speech Detection with Deep Learning and Transfer Learning." . https://arxiv.org/abs/2108.03305.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. *Text Classification Improved by Integrating Bidirectional LSTM with Two-Dimensional Max Pooling*.

INF-UFRGS at SemEval-2022 Task 5: analyzing the performance of multimodal models

Gustavo A. Lorentz
Inst. of Informatics / UFRGS – Brazil
galorentz@inf.ufrgs.br

Viviane P. Moreira
Inst. of Informatics / UFRGS – Brazil
viviane@inf.ufrgs.br

Abstract

This paper describes INF-UFRGS submission for SemEval-2022 Task 5 Multimodal Automatic Misogyny Identification (MAMI). Unprecedented levels of harassment came with the ever-growing internet usage as a means of worldwide communication. The goal of MAMI is to improve the quality of existing methods for misogyny identification, many of which require dedicated personnel, hence the need for automation. We experimented with five existing models, including ViLBERT and VisualBERT - both uni and multimodally pretrained - and MMBT. The datasets consist of memes with captions in English. The results show that all models achieved Macro-F1 scores above 0.64. ViLBERT was the best performer with a score of 0.698.

1 Introduction

Social media and anonymity enable the spread of hateful speech, which explains why misogyny is prevalent and abundant on the internet. Not only is it present, but also increasingly so, as confirmed by Farrell et al. (2019). The platforms that contribute to the sharing of hateful content dedicate a considerable amount of human effort in detecting, analyzing, and eventually removing these contents. The task is demanding due to the nature of the posts, which are frequently not straightforward – they often contain irony and slang. Additionally, the textual information needed to automatically classify a post as misogynistic might be part of an image, in the form of a meme. That prevents sexist posts from being immediately detected by algorithms that rely solely on textual input.

In this paper, we describe the training and usage of five different multimodal models applied to detecting misogynistic memes in the scope of SemEval-2022 Task 5 Multimodal Automatic Misogyny Identification (MAMI) (Fersini et al., 2022). We explain the distinction between the models and compare their performances in light of differences in pretraining (unimodal or multimodal).

Among the five models, the one which achieved the highest score was ViLBERT, reaching the 32nd position on the leaderboard (out of 83 participants), with a score of 0.698. The one which performed the worst was MMBT-Grid, with a score of 0.649.

The remainder of this paper is organized as follows: Section 2 covers background and related work. Section 3 presents an overview of our system. The experimental setup is described in Section 4. Section 5 presents our results. Then, Section 6 concludes the paper.

2 Background and Related Work

One crucial aspect of this task is the multimodality of inputs. Most of the time, a meme requires both textual and visual information to be correctly understood. Not only because the punch line usually comes in written form, but also because texts and images often contradict each other for humouristic purposes. Take for example Figure 1. The text alone indicates a positive feeling towards an object that makes sandwiches. The image, if one would remove the caption, would show a woman. But when taken into consideration simultaneously, it is a sexist meme implying that women exist to make men sandwiches.

We took part only in Subtask A, in which the goal of the model is to take a meme such as the one in Figure 1 as input and indicate whether it is misogynistic or not.

The Hateful Memes Challenge (Kiela et al., 2021) is similar to MAMI since both address hateful multimodal contents. Participants in the Hateful Memes Challenge received a dataset of memes with visual as well as textual inputs and had to predict whether the memes were hateful.

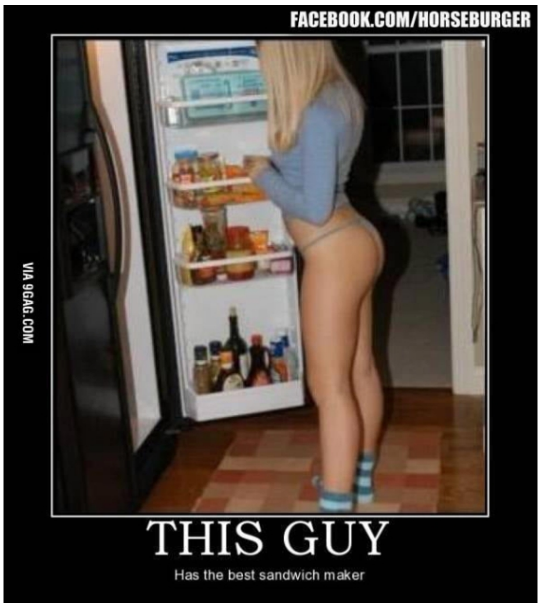


Figure 1: Example of a meme from the MAMI dataset

MMF (Singh et al., 2020) is a multimodal framework from Facebook AI Research and it implements state-of-the-art visual and language models, such as VisualBERT (Li et al., 2019), ViLBERT (Lu et al., 2019), M4C (Hu et al., 2020), and Pythia (Jiang et al., 2018), among others. MMF provides code and model implementations for The Hateful Memes Challenge. Their work served as the primary inspiration for our experiments, in which we apply many of the same models to the Multimedia Automatic Misogyny Identification (MAMI) dataset.

3 System Overview

We used MMF (Singh et al., 2020) to train five models on the MAMI dataset. These models can be briefly described as follows.

1. **MMBT-Grid** is a supervised multimodal bitransformer that jointly finetunes unimodally pretrained text and image encoders by projecting image embeddings to text token space. Its inputs are the concatenation of textual embeddings and the final activations of a ResNet after pooling – the downsampling of dimensions – and positional and segment encodings. The final activations are transformed so that they fit the dimensions of the transformers' hidden layers.

2. **ViLBERT and ViLBERT CC** ViLBERT (Lu et al., 2019) consist of two parallel models, one that operates over visual inputs, and another that operates over textual inputs. Both models operate similarly to BERT, *i.e.*, they are a series of transformer blocks. The difference lies in the *Co-attentional Transformer Layers* introduced by the researchers. During the attention calculation, they compute the usual Q, K, and V matrices. However, the textual K and V are passed to the visual multi-headed attention block, and the visual K and V are passed to the textual multi-headed attention block. The rest of the transformer operations proceed normally, causing multi-modal features since each modality pays attention to the other.

3. **VisualBERT and VisualBERT COCO** - VisualBERT extends BERT by modifying the input it processes. Making use of features extracted from Object Proposals – a set of image regions likely to contain objects – the model can capture the interaction between text and image. The model does that by treating these features as usual BERT input tokens, appending them to the textual tokens. That is, VisualBERT uses the self-attention mechanism to align textual and visual elements implicitly.

Two versions of ViLBERT and VisualBERT were used. The distinction between these two versions lies not in the architecture, but rather in how they were pretrained. The multimodally pretrained versions, ViLBERT CC and VisualBERT COCO, are the official ones published by Lu et al. (2019) and Li et al. (2019), respectively. The unimodally pretrained versions are, as explained by Kiela et al. (2021), *multimodal models that were unimodally pretrained (where, for example, a pretrained BERT model and a pretrained ResNet model are combined in some way).*

4 Experimental Setup

The dataset used to train all models was the one provided by the organization team. The data has not been augmented or modified in any way. Training data consists of 10,000 memes, trial data has 100 memes, and validation data has 1,000 memes.

The MMF framework comes with implementations of state-of-the-art models, preconfigured with hyperparameters. In our experiments, the default configurations of the models were used. All models share the same values for the main configurations, such as 1e-5 for learning rate, 22,000 for maximum number of steps, 128 tokens at most for text processing and, due to memory limitations, one hyperparameter was set to a fixed value, that is, models used batch sizes of 16 for training. Hyperparameter optimization could, therefore, be applied to the models to obtain better results. The configuration files are open to inspection and change, given that they are publicly available at Facebook Research's Github repository[1]. The specific commit that was used for this work is available here[2]. The training process of all models, excluding MMBT-Grid, uses image features, which were not included with the dataset supplied by the organization team. We relied on a script included in MMF to extract these features, using ResNet-152.

The main evaluation metric used in the task and during training is macro-F1. Here we also report True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) Rates.

5 Results

In this section, we report on our experimental results organized around four questions.

5.1 What are the best and worst models?

The results obtained by each model can be seen in Table 1. The best and worst-performing models were, respectively, ViLBERT and MMBT-Grid, with macro-F1 scores of 0.698 and 0.649. With this score, ViLBERT ranked 32nd on Subtask A. It is worth pointing out that they had very similar values for TP-rate. MMBT-Grid achieved a value of 0.866, despite being the worst-ranked among all five models. That means it had a good performance in identifying misogynistic memes. The problem is evidenced by the TN and FP rates. MMBT-Grid was the worst at classifying memes that are not misogynistic, with a TN-rate of 0.463, the lowest of all. It also has the highest FP rate of 0.537. Analyzing ViLBERT's metrics, we can see that what guaranteed it the first place among the five models was the TP and FN rate, which were, respectively, the highest and the lowest. VisualBERT COCO was the best at correctly classifying the negative class (TN rate = 0.581), but it also had, by far, the highest FN rate (0.21).

The differences in performance can not be explained by the usage of uni or multimodal pretraining. This is evidenced by the similarity between scores obtained by unimodally pretrained models (VisualBERT and ViLBERT) and that by multimodally pretrained models (VisualBERT COCO and ViLBERT CC). Additionally, the mentioned models share the same architecture (ViLBERT with ViLBERT CC and VisualBERT with VisualBERT COCO), and so it can not be the explanation for the differences in performance. However, what seems to have impacted scores the most is the use of image features during training, since MMBT-Grid performs the worst.

5.2 Do multimodally pretrained models perform better?

It is interesting to notice that there was no great difference in performance between unimodally and multimodally pretrained models, such as VisualBERT vs. VisualBERT COCO and ViLBERT vs. ViLBERT CC. This finding is in line with Kiela et al. (2021), who worked on the Hateful Memes dataset. Nevertheless, while multimodally pretrained models were slightly better on Hateful Memes, here the unimodally pretrained version of ViLBERT yielded slightly better results, but the difference was not statistically significant (according to a Wilcoxon signed-rank test).

5.3 Can combining classifiers improve classification performance?

To answer this question we analyzed the predictions of the five models for each instance on the evaluation dataset. Our results have shown that:

- 86.89% of the instances were correctly predicted by at least one model;
- 77.58% of the instances were correctly predicted by at least two models;
- 69.67% of the instances were correctly predicted by at least three models;
- 61.76% of the instances were correctly predicted by at least four models;
- 47.95% of the instances were correctly predicted by all models.

[1]https://github.com/facebookresearch/mmf

[2]https://github.com/facebookresearch/mmf/tree/d31f8776f3bee53e7be722cb6d6c7ecf0827cc30/mmf/configs

Model	Macro-F1	TP	TN	FP	FN
MMBT-Grid	0.649	0.866	0.463	0.537	0.134
VisualBERT	0.666	0.874	0.483	0.517	0.126
VisualBERT COCO	0.679	0.786	0.581	0.419	0.214
ViLBERT CC	0.697	0.836	0.571	0.429	0.164
ViLBERT	0.698	0.874	0.541	0.459	0.126

Table 1: Macro-F1 scores, true positive and negative rates, and false positives and negative rates for our models

This analysis suggests that, if we were to use a simple majority voting system to determine the predicted label for images, the obtained accuracy score would be 69.67%, which does not surpass the score achieved by ViLBERT alone. Additionally, we tried combining the predictions of the classifiers by averaging their output probabilities. Similar to what we found with majority voting, there were no performance improvements in relation to ViLBERT on its own.

5.4 How correlated are the models?

Table 2 shows the Pearson correlation coefficient calculated for all pairs of models to measure their level of agreement, *i.e.,* how many images they classified with the same label. We can see that ViLBERT and ViLBERT CC have the highest correlation coefficient, 0.78. We initially supposed that the reason for their high similarity was that they share the same architecture, but further analysis showed that VisualBERT and VisualBERT COCO, the other models that also share architectures, have low similarities. Therefore, the initial hypothesis was wrong and we can assert that the reason for the difference in similarity resides in the pretraining modality, since that is the only distinction between the models. We see that MMBT-Grid and ViLBERT have a correlation score of 0.61, while the lowest score is between MMBT-Grid and VisualBERT, 0.56. The fact that all correlation scores can be classified between strong and moderate explains why there were no gains in combining the models in an ensemble.

5.5 Is there any pattern in memes that were erroneously classified?

We analyzed images that were wrongly classified by all five models. They were, in total, 131 images. Through visual inspection, we were able to identify a pattern in the captions. We noticed that most false positives contained words like "girl", "girls", "woman", and "women", while false negatives did not present these words. To confirm this, we examined the frequency of these words in training and test datasets. The term "girl" appeared in approximately 4.57% of not misogynous memes in the training dataset, and in 6.37% of misogynous memes, that is, 1.39 times more often. This proportion, however, is almost reversed in the test dataset, in which the term appears in 11.1% of *not* misogynous memes, and only in 7.1% of misogynous memes, that is, 1.56 times *less* frequently. This might explain the high number of wrong classifications for memes that contain this word. For the term "women", training dataset analysis shows that 8.27% of misogynous memes had this word, while appearing in only 2% of not misogynous memes, about 4.13 times less often, while in the test dataset, 5.8% of misogynous memes had it, and 2.4% of not misogynous memes, that is, 2 times less. The change in word frequency for this term might also have contributed to misclassification.

6 Conclusion

In this paper, we described our submission to SemEval-2022 Task 5. Using Hateful Memes and MMF as inspiration, we wanted to replicate their methods in a similar context. Although hateful and misogynistic memes share some overlap, there are important distinctions between them, regarding different vocabulary, context, and targets (*i.e.,* hate can be directed towards anyone, while misogyny cannot).

In our experiments, we trained five models and confirmed that they reach similar performances in this dataset as they do in Hateful Memes. Our best model, ViLBERT, reached a F1 score of 0.698 and ranked 32nd out of 83 on the leaderboard. We showed that using a majority voting system with all models would not be beneficial. The models could be further improved by hyper-parameter

	ViLBERT CC	VisualBERT	VisualBERT COCO	MMBT-Grid	ViLBERT
ViLBERT CC	1.00	0.66	0.58	0.61	0.78
VisualBERT		1.00	0.57	0.56	0.69
VisualBERT COCO			1.00	0.58	0.59
MMBT-Grid				1.00	0.61
ViLBERT					1.00

Table 2: Pearson correlation for each pair of models

tuning. We could also have experimented with late/early fusion, which, as suggested by Hateful Memes (Kiela et al., 2021), has an impact on performance, and we leave this as future work.

Acknowledgments. This work was partially supported by CAPES Finance Code 001.

References

Tracie Farrell, Miriam Fernandez, Jakub Novotny, and Harith Alani. 2019. Exploring misogyny across the manosphere in reddit. In *Proceedings of the 10th ACM Conference on Web Science*, WebSci '19, page 87–96, New York, NY, USA. Association for Computing Machinery.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. 2020. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. 2018. Pythia v0.1: the winning entry to the vqa challenge 2018.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2021. The hateful memes challenge: Detecting hate speech in multimodal memes.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

Amanpreet Singh, Vedanuj Goswami, Vivek Natarajan, Yu Jiang, Xinlei Chen, Meet Shah, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. 2020. Mmf: A multimodal framework for vision and language research. `https://github.com/facebookresearch/mmf`.

MMVAE at SemEval-2022 Task 5: A Multi-modal Multi-task VAE on Misogynous Meme Detection

Yimeng Gu
Queen Mary
University of London
United Kingdom
yimeng.gu@qmul.ac.uk

Ignacio Castro
Queen Mary
University of London
United Kingdom
i.castro@qmul.ac.uk

Gareth Tyson
Queen Mary
University of London
United Kingdom
g.tyson@qmul.ac.uk

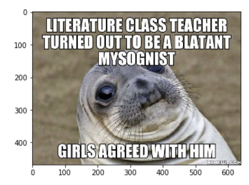
Abstract

Memes have become quite common in day-to-day communications on social media platforms. They often appear to be amusing, evoking and attractive to audiences. However, some memes containing malicious content can be harmful to targeted groups. In this paper, we study misogynous meme detection, a shared task in SemEval 2022 - Multimedia Automatic Misogyny Identification (MAMI). The challenge of misogynous meme detection is to co-represent multi-modal features. To tackle with this challenge, we propose a Multi-modal Multi-task Variational AutoEncoder (MMVAE) to learn an effective co-representation of visual and textual features in the latent space. Our goal is to automatically determine if a meme contains misogynous information and then identify its fine-grained category. Our model achieves F_1 scores of 0.723 on the MAMI sub-task A and 0.634 on sub-task B. We carry out comprehensive experiments on our model's architecture and show that our approach significantly outperforms several strong uni-modal and multi-modal approaches. Our code is released on github[1].

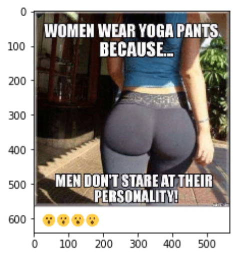
1 Introduction

With the rapid development of social media, the use of image-based memes has been growing. People use memes for various purposes, such as to express humor (Velioglu and Rose, 2020), or to attract greater attention. Enabling this, simple websites that allow people to easily create new memes have further seen their proliferation.

However, this easy composition has allowed people to easily embed harmful messages within memes, often circumventing more traditional text-based moderation tools (Malmasi and Zampieri, 2017). This paper particularly focuses on misogynous memes (*i.e.* hatred towards women) — see Figure 1 for two examples. As some platforms may choose to limit the sharing of such material, we argue it is vital to build tools that can automatically identify misogynous memes at scale.


LITERATURE CLASS TEACHER TURNED OUT TO BE A BLATANT MYSOGNIST
GIRLS AGREED WITH HIM

(a) A misogynous meme with an uncorrelated image.


WOMEN WEAR YOGA PANTS BECAUSE...
MEN DON'T STARE AT THEIR PERSONALITY!

(b) A misogynous meme with a correlated image.

Figure 1: Examples of misogynous memes. In (a), the embedded text is misogynous. In (b), both the image and the embedded text are misogynous.

Many prior works have sought to analyze the content of memes. Some have looked into meme emotion analysis (Sharma et al., 2020; Smitha et al., 2018), meme ecosystem measurements (Zannettou et al., 2018) and meme auto-generation (Vyalla and Udandarao, 2020). Past efforts have also shed light on hateful meme detection (Kiela et al., 2020) or offensive meme detection (Sabat et al., 2019) more generally.

This paper builds on these prior works to automatically detect *misogynous* meme content (Fersini et al., 2022). This comes with several key challenges. First, a meme usually comes with both a visual and a textual part. Sometimes the standalone image or text is not necessarily hateful or toxic, but when combined together, the semantic meaning becomes harmful. To effectively understand the semantic meaning of a meme, information encoded in both modalities should be considered. Second, different memes frequently have the same image, but are embedded with different text (and thus have different and even opposite meanings). This makes reliance on image-hash lists ineffective. It is also common that the image and text of a meme are unrelated, as in Figure 1a. In this case, finding an

[1]https://github.com/MMVAE-project/MMVAE

accurate co-representation of both visual and textual features is vital. Finally, malicious information contained in a meme can have granular labels and even belong to multiple categories. Thus, it is often necessary to devise a more nuanced taxonomy.

With the above challenges in mind, we propose MMVAE: a pipeline to determine if a meme contains misogynous information, and to identify its fine-grained hateful labels accordingly. Specifically, our contributions are as follows:

1. We propose a Multi-modal Multi-task Variational AutoEncoder (MMVAE) that effectively co-represents the textual and visual features of the meme. We use this to predict if a meme is misogynous/non-misogynous. We further expand our model to predict more fine-grained labels, *e.g.* shaming, violence, objectification.

2. We evaluate our model on SemEval competition Task 5: Multimedia Automatic Misogyny Identification (Fersini et al., 2022). We achieve an F_1 score of 0.723 on sub-task A and 0.634 on sub-task B.

3. We analyze the strengths and weaknesses of our multi-modal approach by presenting a text-based error analysis and case study. Our model is better at identifying misogynous memes with a high precision, yet less effective in determining the correct label for non-misogynous memes.

2 Background

Our approach leverages multi-modal learning, pre-trained model, variational autoencoder and multi-task learning. Below, we provide a brief overview of these concepts.

2.1 Multi-modal Feature Representation

We leverage multi-modal learning to co-represent features from both image and text. Common multi-modal features co-representation techniques (Zeng et al., 2021) include fusion mechanism (early at feature level (Su et al., 2020), or late at decision/scoring level (Poria et al., 2016)), tensor factorization (Zadeh et al., 2017; Mai et al., 2019) and complex attention mechanisms which can be further classified as dot-product attention (Yu et al., 2021), multi-head attention (Cao et al., 2021; Wu et al., 2021), hierarchical attention (Pramanick et al., 2021), attention on attention (Liu et al., 2021) among others. These techniques have proven to be effective in encoding multi-modal features. However, leveraging multi-modal features doesn't always enhance the task performance. Hence, another topic that has raised concern in multi-modal learning is the uni-modal contribution analysis. A straightforward approach (Hessel and Lee, 2020; Frank et al., 2021) is to ablate cross-modal inputs or interactions in order to evaluate the model's performance on uni-modal data. Zeng et al. (2021) demonstrated that multi-modal models might not achieve optimal performance because there are noises contained in each modality.

2.2 Pre-trained Model

We adopt pre-trained models to embed the text and image of a meme.

Usually trained on large dataset corpus, pre-trained models have achieved state-of-the-art (SOTA) performances on various Natural Language Processing (NLP) and Computer Vision (CV) benchmark tasks. Thus, they can be used as a powerful embedding tool or be easily fine-tuned on downstream tasks. Popular language pre-trained models such as BERT (Devlin et al., 2019), LASER (Artetxe and Schwenk, 2019) and LaBSE (Feng et al., 2020) have constantly updated the SOTA performance on downstream NLP tasks. Similarly, image pre-trained models including ResNet (He et al., 2016), VGG (Simonyan and Zisserman, 2015), and Inception (Szegedy et al., 2016) have proved their effectiveness in multiple tasks. Recently, there has been increasing interest in pre-training multi-modal models. Lu et al. (2019) proposed ViLBert, which extends BERT to multi-modal two-stream models that interacts through co-attentional transformer layers and can be easily transferred into performing multiple visual-and-language tasks. Similarly, LXMERT (Tan and Bansal, 2019) further included a cross-modality encoder that captures cross-modality relationships. Instead of implementing two stream models, Su et al. (2020) input the caption and image regions all together to the modified BERT model named VL-BERT. Radford et al. (2021) jointly trained an image encoder and a text encoder known as CLIP to match the image and its corresponding caption by contrastive learning.

2.3 VAE Overview

We later use Variational AutoEncoder (VAE) to fuse multi-modal features. Thus, here we introduce the basic structure and loss calculation of a VAE. A VAE is an unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data (Li et al., 2019a). The aim of finding a representation in the latent space is to capture meaningful factors of variation in the data (Li et al., 2019a). Typically, a VAE consists of an encoder and a decoder, which look "symmetrical" in the model's architecture. The encoder learns a latent variable space where a latent variable z is sampled from and input to the decoder for original data reconstruction. Figure 2 shows the structure of a simple VAE.

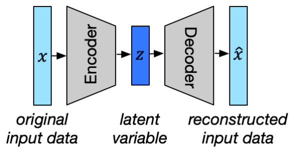


Figure 2: A simple VAE example.

Encoder. The encoder network of a VAE can have various structures. For textual input, the common structure is the Recurrent Neural Networks (RNN), *e.g.* Bi-LSTM (Cheng et al., 2020). For image input, the common encoder structure is the Convolutional Neural Networks (CNN). Mathematically, the encoder can be described as $q_\phi(\mathbf{z}|\mathbf{x})$, where ϕ is the parameters of the encoder network. $q_\phi(\mathbf{z}|\mathbf{x})$ stands for the probability distribution of latent variable z given x.

Decoder. Generally, the decoder network of a VAE is symmetrical to the encoder. It reconstructs the input by sampling from the learned latent variable space subject to a Gaussian distribution. The decoder can be described as $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, where $\boldsymbol{\theta}$ is the parameters of the decoder network. $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ stands for the probability distribution of reconstructed x given z.

Loss. In order to compute the loss of VAE in our model, we first introduce how to compute the loss of a general VAE. Since VAE models the probabilistic generation of data $\{x^{(i)}\}$, the goal is to maximize the (log) data likelihood (Kingma and Welling, 2014):

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})) \\ +\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)})\end{aligned} \tag{1}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are parameters of the decoder and encoder network respectively. The first KL divergence term on the right hand side can't be computed explicitly but it is non-negative. The second term is called the *lower bound* on the marginal likelihood of datapoint i and can be rewritten as (Kingma and Welling, 2014):

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) \\ +\mathbf{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z})\right]\end{aligned} \tag{2}$$

Hence, to maximize the log likelihood, we only need to maximize $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)})$. To achieve that, the KL divergence term in eq. 2 should be minimized and the expected log likelihood of datapoint i should be maximized (equivalent to minimizing the expected reconstruction error).

2.4 Multi-task Learning

Our work leverages the concept of multi-task learning to learn fine-grained category labels of misogynistic memes. According to Ruder (2017), as long as a model is optimized by more than one loss functions, it is doing multi-task learning. When the tasks are relevant and the knowledge learned from one task could benefit the learning of other tasks, applying multi-task learning will have promising performances (Caruana, 1997). Multi-task learning has been successfully applied in many NLP tasks such as text classification (Cheng et al., 2020; Khattar et al., 2019), sentiment analysis (Majumder et al., 2019), neural machine translation (Niehues and Cho, 2017), etc. However, the most common issue of multi-task learning is the negative transfer when the performance on single task is undermined. Lee et al. (2016) avoids negative transfer by allowing asymmetric transfer between tasks. Wu et al. (2020) observes that misalignment between tasks can cause negative transfer. Wu et al. (2019) proposed a method to filter and select shared feature to prevent adverse features being integrated into certain tasks.

3 Problem Statement

The paper strives to build a classifier that can distinguish misogynous *vs.* non-misogynous memes. We break this down into two sub-tasks:

- Sub-task A: Given a meme's image I and its corresponding text transcription T, we must predict its binary (0/1) label on *misogyny* (Fersini et al., 2022).
- Sub-task B: Given a meme's image I and its corresponding text transcription T, we must

predict its binary (0/1) label on the following classes: *shaming*, *stereotype*, *objectification* and *violence* (Fersini et al., 2022), which are types of misogyny.

Note, the meme's image I has the embedded text on it, and T is transcribed from the meme's embedded text.

4 Our Approach

As illustrated in Figure 3, our MMVAE model consists of 3 components: (*i*) the Image/Text embedding Module, (*ii*) the Variational AutoEncoder Module and (*iii*) the Multi-Task Learning Module. The embedding module turns the inputs into unimodal vector representations. After that, the VAE module fuses multi-modal representations and generates a co-representation. Finally, the multi-task learning module gives the label prediction. In this section, we first introduce each part of our proposed model, and then demonstrate how to put them all together and jointly train the model.

4.1 Image/Text Embedding Module

In our pipeline, we first adopt two pre-trained models to embed the meme's text and image (illustrated in Figure 3 left part). We directly input the provided text transcription and meme's image to the pre-trained models to get embeddings as module output. However, when using ResNet-50 and BERT as pre-trained models, we apply data preprocessing beforehand.

Image Embeddings. To obtain the meme's image embedding, we have experimented with 2 different pre-trained models: ResNet-50 (He et al., 2016) pre-trained on ImageNet (Deng et al., 2009) and the multi-lingual version of OpenAI CLIP-ViT-B32 (Radford et al., 2021). When using ResNet-50 to embed images, we transform the input images by resizing it to 224×224, applying random rotation, random horizontal flip, random crop and normalization for data augmentation. We use the last fully connected layer's output as our image embedding. When using CLIP to embed images, although we have experimented with data transformation techniques, our optimal performance is achieved by directly embedding the raw image.

Text Embedding. To obtain the meme's text embeddings, we experiment with 4 different pre-trained models: BERT for sentence classification (Devlin et al., 2019), LASER (Artetxe and Schwenk, 2019), LaBSE (Feng et al., 2020) and the multi-lingual version of OpenAI CLIP-ViT-B32. We set the maximum input sentence length to 512 tokens when using BERT while we directly input the meme's text to other pre-trained models.

4.2 Variational AutoEncoder (VAE) Module

The next module in our pipeline takes the embeddings as an input and finds a multi-modal co-representation. We assume that there is an effective multi-modal co-representation in the latent space which can better capture the inter-relationship between text and image data.

In our pipeline, we leverage VAE to learn the multi-modal co-representation (illustrated in Figure 3 middle part). The input to this module are the embeddings generated from the pre-trained models, and the output is the reconstructed embeddings. Yet what we need for later meme detection classifiers is the latent variable generated from the VAE encoders.

Encoders. We build a text encoder and an image encoder to first learn the latent variables of the text embedding and image embedding separately. In our case, since the input embeddings are already semantically meaningful, there is less need to further extract semantic information using complex and deep layers. Therefore, we decide to use 1 fully connected layer for each modality as the encoder structure. And then we concatenate the learned latent variables to form a multi-modal co-representation z in the latent space.

Decoders. Accordingly, we build a text decoder and an image decoder to reconstruct the text embedding and image embedding from the learned multi-modal co-representation z. Our decoders still consists of 1 fully connected layer, with the number of output channels equal to the number of input channels in the encoder network.

Losses. We calculate the reconstructed image embedding loss $\mathcal{L}_{img}$, the reconstructed text embedding loss $\mathcal{L}_{txt}$ (corresponding to the 2nd part of eq. 2) and the KL loss KLD (corresponding to the 1st part of eq. 2) from the sampled latent variable z for gradient descent. The reconstruction loss is calculated by L_2 loss function (squared error). The KL loss is calculated using the formula in Appendix B of Kingma and Welling (2014).

4.3 Multi-task Learning

We leverage multi-tasking learning because we expect that learning the fine-grained hateful class labels will benefit the misogyny meme detection and

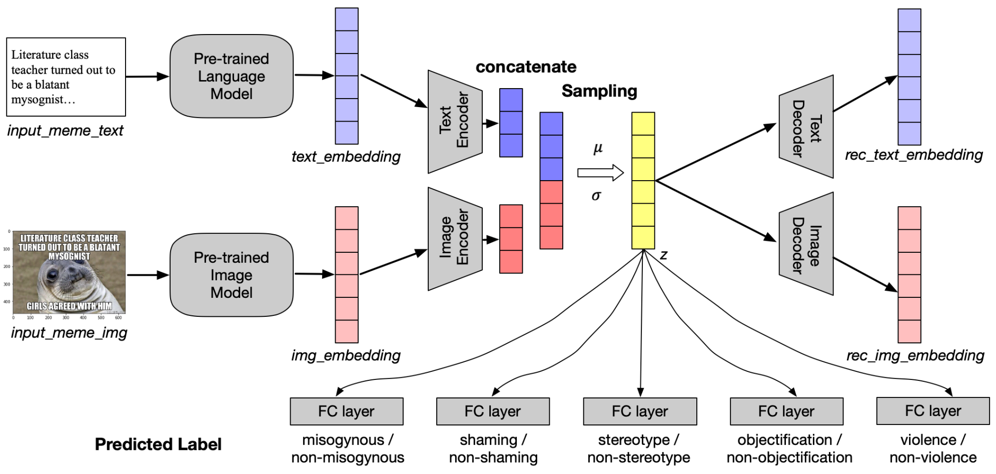


Figure 3: The architecture of our proposed Multi-modal Multi-task VAE (MMVAE) model. The upper part illustrates the image/text embedding module and the VAE module from left to right. The lower part shows the multi-task learning module.

vice versa. The input to this module is the latent variable z (colored yellow in Figure 3) learned in Section 4.2 and the output is label predictions on each class.

Our model learns 5 tasks at the same time: misogyny detection, shaming detection, stereotype detection, objectification detection and violence detection. The latter 4 classes are more specific types of misogyny. Note, if a meme is misogynous, it could fall into more than one specific misogyny classes.

The architectures are the same across all the sub-networks in our model as shown in Figure 3: 1 fully connected layer followed by the softmax binary classifier. Yet the sub-networks are independent - the parameters are not shared.

Here the cross entropy loss function is used to calculate the loss for each task:

$$\mathcal{L}_t = -\mathbf{E}_{y \sim Y_t}\left[y \log y_t + (1-y)\log(1-y_t)\right] \tag{3}$$

where y is the ground-truth label of data point x in t detection (e.g. misogyny detection) and y_t is the predicted probability of x belonging to class t (e.g. misogyny).

4.4 Putting It All Together

We have introduced each module in our MMVAE and how to calculate their standalone losses. Next, we introduce how to jointly train the model by putting three modules together.

The total loss used to compute gradient descent is composed of the reconstructed image error L_{img}, reconstructed text error L_{txt}, KL divergence KLD, and multi-task cross entropy losses L_t, where $t \in \{$*misogyny*, *shaming*, *stereotype*, *objectification*, *violence*$\}$. This can be expressed as:

$$\mathcal{L}_{total} = \lambda_i \mathcal{L}_{img} + \lambda_t \mathcal{L}_{txt} + \lambda_{kl} KLD + \sum_t \lambda_t \mathcal{L}_t \tag{4}$$

where λs are used to adjust the learning focus, so that we can direct the focus of learning to the task we care more about. We calculate gradient descent on $\mathcal{L}_{total}$ and back propagate it to update model's parameters.

5 Experimental Setup

In this section, we first introduce the Multimedia Automatic Misogyny Identification (MAMI) dataset released by the task organizer (Fersini et al., 2022) and then discuss our experimental settings.

5.1 Dataset

The MAMI dataset contains 10,000 memes in the training set and 1,000 memes in the test set. Each meme has an associated text transcription in English and labels on 5 classes. The label distributions of given classes are not balanced and are summarized in Table 1. Only in the misogyny class, the labels are totally balanced. Detailed task descriptions can be found in Section 3.

5.2 Experimental Settings

We have experimented with different latent variable (z) sizes of 256, 512, and 1024. 512 has the best

Class	# Positive	# Negative
Misogyny	5,000	5,000
Shaming	1,274	8,726
Stereotype	2,810	7,190
Objectification	2,202	7,798
Violence	953	9,047

Table 1: Number of positive labels and negative labels in each class.

performance on the validation set. Hence, we set the latent variable size to 512 during the evaluation phase.

Different pre-trained models generate embeddings with different dimensions, D. In both encoders, we map the input embeddings' dimension from D to half of z's dimension.

For the optimizer, we use the Adam optimizer with weight decay set to 1e-5. The batch size is set to 64. In addition, the initial learning rate is 1e-4 and divided by 4 every 8 training epochs. The model is trained for 30 epochs in total with early stopping.

We split the training set into the training and validation set with a ratio of 9:1. We train our model on 9,000 memes from the original training set and test it on the rest of 1,000 memes to evaluate our model's performance.

5.3 Baselines

BERT (Devlin et al., 2019). As the text-only baseline, we use BERT for sequence classification. Here, only the text transcription is used for misogyny detection. BERT has achieved SOTA performances on most NLP benchmark tasks. Therefore, we consider it as a robust textual baseline model for this task.

ResNet-50 (He et al., 2016). As the image-only baseline, we use ResNet-50. Here, only the image is used for misogyny detection. ResNet-50 has achieved SOTA performances on ImageNet, a benchmark task in CV. Therefore, we consider it as a robust image baseline model for this task.

CNN-Based VAE. We build an image-only VAE model whose encoder and decoder are both composed of 5 CNN layers. Here, only the meme image is used for misogyny detection. This image-only input model is used to compare with our MMVAE: both of them are constructed based on VAE.

Note, we only experiment with BERT and ResNet-50 with sub-task A because we directly use the pre-trained model's architecture without incorporating multi-task learning into them.

6 Results and Analysis

In this section, we present our model's performances on both tasks, and give further analysis on its strengths and weaknesses.

6.1 Evaluation Metrics

The main evaluation metric for both tasks is F_1 score (macro-F_1 for sub-task B), which calculates the average of F_1 for both labels:

$$F_1 = \frac{pos\_F_1 + neg\_F_1}{2} \tag{5}$$

We also extend it to include *precision* and *recall* by calculating the average score similarly.

6.2 Results

Table 2 summarizes the performances of our model and the baselines. The first group of models are unimodal. The second group of models share the same architecture, MMVAE, but with different embedding methods. Furthermore, MMVAEs used in the third group come from the best performed model of the second group, *i.e.* MMVAE$_{\text{LASER+CLIP}}$. In the third group, we have tried a number of techniques to mitigate the overfitting problem: adding different dropout rates, concatenating the word embeddings generated by LASER and LaBSE, adding one more liner layer to the text VAE encoder, and introducing image transforms.

For sub-task A, the optimal F_1 performance is 0.723, which is achieved by applying batch normalization layers and set dropout = 0.2 after each linear layer in the encoders and decoders of the VAE. Among all the tweaks we apply to reduce overfitting, introducing dropout is the most effective by which we get our top two F_1 scores (0.723 and 0.714). Instead of dropping more parameters, keeping 80% of them produces a better result (0.723).This might because the number of parameters in our model is not large, so excluding more will harm the learning capability. We also see a performance improvement to 0.712 by concatenating text embeddings from LASER and LaBSE, which introduces more information, thereby reducing overfitting. The other two attempts fail to improve the performance, the reason might be that adding more layers doesn't make the model more generalizable and image transforms is not effective when applying pre-trained models to embed.

Model	Sub-task A			Sub-task B		
	Precision	Recall	F1	Precision	Recall	macro-F1
BERT	0.608	0.632	0.589	-	-	-
ResNet-50	0.635	0.656	0.622	-	-	-
CNN-VAE	0.526	0.550	0.462	0.514	0.545	0.469
MMVAE$_{\text{BERT+ResNet}}$	0.640	0.653	0.632	0.543	0.590	0.532
MMVAE$_{\text{BERT+CLIP}}$	0.707	0.752	0.693	0.586	0.633	0.589
MMVAE$_{\text{LASER+CLIP}}$	0.721	0.756	**0.711**	0.594	0.648	0.600
MMVAE$_{\text{LaBSE+CLIP}}$	0.707	0.751	0.694	0.578	0.686	0.575
MMVAE$_{\text{CLIP+CLIP}}$	0.712	0.760	0.698	0.587	0.658	0.592
MMVAE$_{\text{+dropout=0.5}}$	0.724	0.759	0.714	0.606	0.656	0.616
MMVAE$_{\text{+dropout=0.2}}$	0.730	0.756	**0.723**	0.613	0.647	0.622
MMVAE$_{\text{+concat}}$	0.721	0.751	0.712	0.602	0.657	0.609
MMVAE$_{\text{+more layers}}$	0.710	0.750	0.698	0.631	0.649	**0.634**
MMVAE$_{\text{+img transform}}$	0.710	0.756	0.696	0.605	0.651	0.615

Table 2: Performance of our MMVAE model and variants on the test set.

For sub-task B, the optimal performance in F_1 is 0.634, which is achieved by applying batch normalization layers after each linear layer in the encoders and decoders, and adding one more linear layer to the text encoder in the VAE.

Our MMVAE's best performance on sub-task A is significantly higher than that of the uni-modal baselines: 22.8% higher than the text-only BERT baseline, and 16.2% higher than the image-only ResNet-50 baseline. This confirms that our proposed model has effectively learned from both textual and visual features.

Our constructed image-only CNN-based VAE produces the least F_1 score, probably because CNN layers are less effective in capturing complex semantic information contained in meme images.

6.3 Text-Based Error Analysis

We observe that on our randomly selected validation set (10% of memes from the training set), MMVAE obtains nearly 0.87 as F_1 score on subtask A. But on the test set, we see a 16.7% performance drop. Thus, there is a large gap between our model's performance on the validation set and the test set. We speculate that multiple factors could cause the misclassifications, for instance, similarities between images that are associated with different text. For simplicity, here we only investigate the impact of hateful text on the classification results. We delay further investigations to future work.

To start, we compute the confusion matrix of our best model: MMVAE$_{\text{+dropout=0.2}}$, which is displayed in Figure 4. There are 500 misogynous memes and 500 non-misogynous memes in the test set. As a result, the number of false negative (56) is much lower compared to the false positives (214). Our model correctly classifies 88.8% of misogynous memes yet only 57.2% of non-misogynous memes.

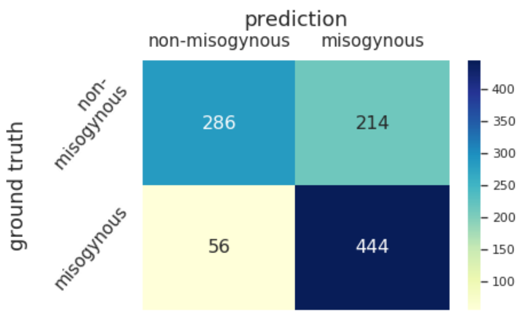


Figure 4: Confusion matrix of MMVAE$_{\text{+dropout=0.2}}$'s prediction.

We conjecture that issues with performance may be driven by the nature of hateful text. To analyze this, we calculate the hateful scores on the memes' text using the sentiment analysis toolkit in (Pérez et al., 2021). Figure 5 presents boxplots showing the scores. We observe that for true positives (TP) and false negatives (FN), the hateful score distributions are similar, although the former's 2nd quartile is much higher. In contrast, we find that true negatives (TN) contain less hateful text in memes compared to false positives (FP), and the mean score value is significantly different. Apparently, non-misogynous memes tend to have lower hateful scores for their text. Based on our analysis, we infer that our model is more confident in assigning the correct non-misogynous label to a meme with less hateful text content. Yet, when assigning misogynous labels (although FNs have comparably lower hateful scores), our model is less accurate. As such, we cannot tell the decision boundary between FN and TP by simply looking at the hateful messages contained in meme's text.

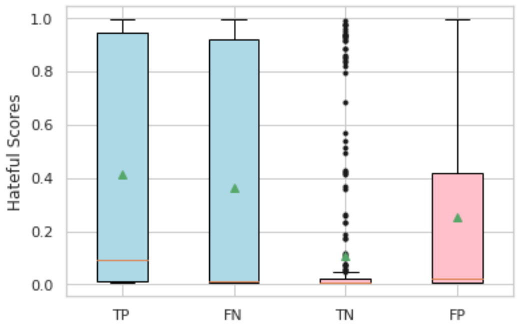
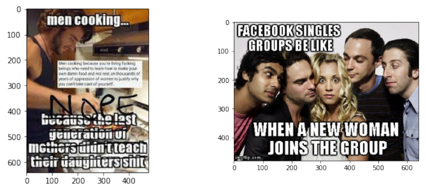


Figure 5: Boxplots of hateful scores on true positives (TP), false negatives (FN), true negatives (TN) and false positives (FP). Green triangle indicates the mean value of the given class. Blue and pink respectively refer to misogynous and non-misogynous memes in ground-truth.

Manual inspection on FPs suggests that the misclassification is potentially driven by several different types of meme. The misogynous text is crossed out in the left meme of Figure 6, but it's still included in the text transcription, which can be confusing for the classifier. Similarly, although the other meme in Figure 6 is not misogynous, it is related to women. We suspect that the presence of certain words associated with femininity e.g. woman, girl, is a another determinate in the prediction. We will test this hypothesis in our future work.


men cooking...
NOPE
because the last generation of mothers didn't teach their daughters shit
FACEBOOK SINGLES GROUPS BE LIKE
WHEN A NEW WOMAN JOINS THE GROUP

(a) Hateful score: 0.017 (b) Hateful score: 0.086

Figure 6: Two examples of FPs, *i.e.* the ground-truth labels are non-misogynous while the predicted labels are misogynous.

7 Related Work

The misogynous meme detection task is novel, but has similarities to other more general hateful meme detection tasks. Related datasets has been released on a number of shared hateful meme detection/classification tasks (Kiela et al., 2020; Mostafazadeh Davani et al., 2021). Most of the prize-winning models adopted visual-and-linguistic pre-trained models. Velioglu and Rose (2020) utilized pre-trained VisualBERT (Li et al., 2019b) to encode the meme image regions and caption all together. Sharif et al. (2021) and Zia et al. (2021) leveraged visual and textual pre-trained models to encode the meme image and the embedded text respectively, and then learned multi-modal co-representation through vector concatenation. Instead of concatenating the vectors, Pramanick et al. (2021) applied self-attention to learn intra-modality semantic alignments. Lippe et al. (2021) used an ensemble of existing multi-modal pre-trained models based on UNITER (Chen et al., 2020). Zhu (2020) showed that directly applying state-of-the-art multi-modal models on hateful meme classification won't get the optimal performance. They used various data pre-processing approaches to get sufficient features, *e.g.* entity tags, as additional inputs to the pre-trained models. Note, there have also been studies of misogyny in uni-modal platforms (Guest et al., 2021; Zeinert et al., 2021; Jiang et al., 2022).

Our work differs from the above. Many of these previous works directly leverage uni-modal embeddings produced by pre-trained models. They then build a relatively simple model afterwards, of which the learnt multi-modal features are not integrated. In contrast, we strive to overcome this limitation via the co-representation of both textual and image modalities. Moreover, we differ in that we are focusing on misogynous meme detection, rather than the broader topic of hateful meme detection.

8 Conclusion

The spread of hateful memes targeting certain groups has become an important problem on social media platforms. To mitigate the negative consequences brought by misogynous memes, we propose a Multi-modal Multi-task Variational AutoEncoder (VAE) to identify them and assign them more fine-grained labels. Our model consists of three main components: the Image/Text Embedding Module, the Variational AutoEncoder Module, and the Multi-Task Learning Module. Our model's performance outperforms the state-of-the-art unimodal baselines by 22.8% and 16.2%. It effectively learns the co-representation of visual and textual features, and is jointly trained on multiple downstream classification tasks. In our future work, we plan to integrate attention mechanism into our model and carry out more comprehensive statistical analysis on model's results.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Meng Cao, Long Chen, Mike Zheng Shou, Can Zhang, and Yuexian Zou. 2021. On pursuit of designing multi-modal transformer for video grounding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9810–9823.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Mingxi Cheng, Shahin Nazarian, and Paul Bogdan. 2020. Vroc: Variational autoencoder-aided multi-task rumor classifier based on text. In *Proceedings of The Web Conference 2020*, pages 2892–2898.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

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

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021. Vision-and-language or vision-for-language? on cross-modal influence in multimodal transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9847–9857.

Ella Guest, Bertie Vidgen, Alexandros Mittos, Nishanth Sastry, Gareth Tyson, and Helen Margetts. 2021. An expert annotated dataset for the detection of online misogyny. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1336–1350.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Jack Hessel and Lillian Lee. 2020. Does my multimodal model learn cross-modal interactions? it's harder to tell than you might think! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 861–877.

Aiqi Jiang, Xiaohan Yang, Yang Liu, and Arkaitz Zubiaga. 2022. Swsr: A chinese dataset and lexicon for online sexism detection. *Online Social Networks and Media*, 27:100182.

Dhruv Khattar, Jaipal Singh Goud, Manish Gupta, and Vasudeva Varma. 2019. Mvae: Multimodal variational autoencoder for fake news detection. In *The world wide web conference*, pages 2915–2921.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Giwoong Lee, Eunho Yang, and Sung Hwang. 2016. Asymmetric multi-task learning based on task relatedness and loss. In *International conference on machine learning*, pages 230–238. PMLR.

Fei-Fei Li, Justin Johnson, and Serena Yeung. 2019a. Lecture notes of cs231n.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019b. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, and Helen Yannakoudakis. 2021. A multimodal framework for the detection of hateful memes. *PMLR*, 133:344–360.

Fuxiao Liu, Yinghan Wang, Tianlu Wang, and Vicente Ordonez. 2021. Visual news: Benchmark and challenges in news image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6761–6771.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Sijie Mai, Haifeng Hu, and Songlong Xing. 2019. Divide, conquer and combine: Hierarchical feature fusion network with local and global perspectives for multimodal affective computing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 481–492.

Navonil Majumder, Soujanya Poria, Haiyun Peng, Niyati Chhaya, Erik Cambria, and Alexander Gelbukh. 2019. Sentiment and sarcasm classification with multitask learning. *IEEE Intelligent Systems*, 34(3):38–43.

Shervin Malmasi and Marcos Zampieri. 2017. Detecting hate speech in social media. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 467–472.

Aida Mostafazadeh Davani, Douwe Kiela, Mathias Lambert, Bertie Vidgen, Vinodkumar Prabhakaran, and Zeerak Waseem, editors. 2021. *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. In *Proceedings of the Second Conference on Machine Translation*, pages 80–89.

Soujanya Poria, Iti Chaturvedi, Erik Cambria, and Amir Hussain. 2016. Convolutional mkl based multimodal emotion recognition and sentiment analysis. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 439–448. IEEE.

Shraman Pramanick, Shivam Sharma, Dimitar Dimitrov, Md. Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021. MOMENTA: A multimodal framework for detecting harmful memes and their targets. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4439–4455.

Juan Manuel Pérez, Juan Carlos Giudici, and Franco Luque. 2021. pysentimiento: A python toolkit for sentiment analysis and socialnlp tasks.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International conference on machine learning*, volume 139. PMLR.

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

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. In *AI for Social Good workshop at NeurIPS*.

Omar Sharif, Eftekhar Hossain, and Mohammed Moshiul Hoque. 2021. NLP-CUET@DravidianLangTech-EACL2021: Offensive language detection from multilingual code-mixed text using transformers. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, pages 255–261.

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Bjorn Gamback. 2020. Semeval-2020 task 8: Memotion analysis–the visuo-lingual metaphor! In *Proceedings of the 14th International Workshop on Semantic Evaluation*, pages 759–773.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015*.

ES Smitha, Selvaraju Sendhilkumar, and GS Mahalaksmi. 2018. Meme classification using textual and visual features. In *Computational Vision and Bio Inspired Computing*, pages 1015–1031. Springer.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111.

Riza Velioglu and Jewgeni Rose. 2020. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge. In *The Hateful Memes Challenge at NeurIPS*.

Suryatej Reddy Vyalla and Vishaal Udandarao. 2020. Memeify: A large-scale meme generation system. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, pages 307–311.

Lianwei Wu, Yuan Rao, Haolin Jin, Ambreen Nazir, and Ling Sun. 2019. Different absorption from the same sharing: Sifted multi-task learning for fake news detection. *CoRR*, abs/1909.01720.

Sen Wu, Hongyang R. Zhang, and Christopher Ré. 2020. Understanding and improving information transfer in multi-task learning. In *International Conference on Learning Representations*.

Yang Wu, Pengwei Zhan, Yunjian Zhang, Liming Wang, and Zhen Xu. 2021. Multimodal fusion with co-attention networks for fake news detection. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2560–2569.

Tiezheng Yu, Wenliang Dai, Zihan Liu, and Pascale Fung. 2021. Vision guided generative pre-trained language models for multimodal abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3995–4007.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1103–1114.

Savvas Zannettou, Tristan Caulfield, Jeremy Blackburn, Emiliano De Cristofaro, Michael Sirivianos, Gianluca Stringhini, and Guillermo Suarez-Tangil. 2018. On the origins of memes by means of fringe web communities. In *Proceedings of the Internet Measurement Conference 2018*, pages 188–202.

Philine Zeinert, Nanna Inie, and Leon Derczynski. 2021. Annotating online misogyny. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3181–3197.

Ying Zeng, Sijie Mai, and Haifeng Hu. 2021. Which is making the contribution: Modulating unimodal and cross-modal dynamics for multimodal sentiment analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1262–1274.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

Haris Bin Zia, Ignacio Castro, and Gareth Tyson. 2021. Racist or sexist meme? classifying memes beyond hateful. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 215–219.

AMS_ADRN at SemEval-2022 Task 5: A Suitable Image-text Multimodal Joint Modeling Method for Multi-task Misogyny Identification

Da Li
Tencent. Inc
Beijing, China
danielsli@tencent.com

Ming Yi
Tencent. Inc
Beijing, China
mingyyi@tencent.com

Yukai He
Tencent. Inc
Beijing, China
yukaihe@tencent.com

Abstract

Women are influential online, especially in image-based social media such as Twitter and Instagram. However, many in the network environment contain gender discrimination and aggressive information, which magnify gender stereotypes and gender inequality. Therefore, the filtering of illegal content such as gender discrimination is essential to maintain a healthy social network environment. In this paper, we describe the system developed by our team for SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. More specifically, we introduce two novel systems to analyze these posts: a multimodal multi-task learning architecture that combines Bertweet for text encoding with ResNet-18 (He et al., 2016) for image representation, and a single-flow transformer structure which combines text embeddings from BERT-Embeddings and image embeddings from several different modules such as EfficientNet (Tan and Le, 2019) and ResNet (He et al., 2016). In this manner, we show that the information behind them can be properly revealed. Our approach achieves good performance on each of the two subtasks of the current competition, ranking 12th for Subtask A (0.746 macro F1-score), 10th for Subtask B (0.706 macro F1-score) while exceeding the official baseline results by high margins.

1 Introduction

Women are influential online, especially in image-based social media such as Twitter and Instagram : 78 percent of women use social media multiple times a day, compared with 65 percent of men.(Elisabetta Fersini) However, although the Internet has opened new opportunities for women, systematic offline inequality and discrimination are replicated in cyberspace in the form of offensive content against them. The popular communication tool in social media platforms is MEME. Memetics is essentially an image, which is characterized by the content of the image and the poster cover text introduced by human beings. Its main goal is fun or irony. Although most of them were created for a joke, in a very short period of time, people began to use them as a form of hatred against women, and then magnified gender stereotypes and gender inequality. Women are exposed to gender discrimination and aggressive information in the network environment. Therefore, the filtering of illegal content such as gender discrimination is essential to maintain a healthy social network environment.

In this paper, we describe the system developed by our team for SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. More specifically, we introduce two novel systems to analyze these posts: a multimodal multi-task learning architecture that combines Bertweet (Nguyen et al., 2020) for text encoding with ResNet-18 (He et al., 2016) for image representation, and a single-flow transformer structure which combines text embeddings from BERT-Embeddings and image embeddings from several different modules such as EfficientNet (Tan and Le, 2019) and ResNet (He et al., 2016).

2 Related Work

Language Model Pre-training. The MLM-based pre-training method used in BERT (Devlin et al., 2019) opens up the pre-training paradigm of language model based on Transformer structure. RoBERTa (Liu et al., 2020) carefully measured the influence of key hyperparameters and training data size, and further enhanced the effect. SpanBERT (Joshi et al., 2020) extends BERT by masking continuous random spans rather than random markers, and trains the span boundary representation to predict the whole content of the shielded span without relying on a single marker representation. MacBERT (Cui et al., 2020) improved RoBERTa in several aspects, especially using MLM as the correction masking strategy (Mac). BERTweet (Nguyen et al., 2020) is the first large-scale language model for English Tweets and produced better performance results than the previous state-of-the-art models on three Tweet NLP tasks: Part-of-speech tagging, Named-entity recognition and text classification.

Visual and Language Multimodal Learning. Some studies explored cross-modal transmission between images and text. Some ideas are related to cross-model representation learning (Aytar et al., 2017; Ngiam et al., 2011), aiming to generate representations that effectively correlate different sensor modes. Recently, several ideas have been proposed to focus on the transformer backbone by improving the pre-training strategy rather than optimizing the backbone structure. VideoBERT (Sun et al., 2019) learns the bidirectional joint distribution of visual and linguistic tag sequences, which are respectively from the vector quantization of frame data and the ready-made speech recognition output. VisualBERT (Li et al., 2019) uses MLM to pre-train specific tasks, which uses the same type ID for language & visual input. VL-BERT (Su et al., 2020) uses a simple Transformer model as the backbone, and uses a mixture of region of interest (ROI) and words in the image as input.

In addition, some ideas have become focused on improving the transformer backbone network. Unicoder-VL (Li et al., 2020) uses three pre-training tasks, including mask language modeling (MLM), mask object classification (MOC) and visual language matching (VLM), to learn the joint representation of vision and language. ViLBERT (Lu et al., 2019) extends the BERT architecture to a multi-modal dual-flow model and processes it in separate flows interacting through the common attention converter layer

3 Task Description

The proposed task, i.e. Multimedia Automatic Misogyny Identification (MAMI) consists in the identification of misogynous memes, taking advantage of both text and images available as source of information. The task will be organized around two main sub-tasks:

- Sub-task A: a basic task about misogynous meme identification, where a meme should be categorized either as misogynous or not misogynous;
- Sub-task B: an advanced task, where the type of misogyny should be recognized among potential overlapping categories such as stereotype, shaming, objectification and violence.

4 Methodology

For this task, we have tried a variety of modeling and optimization methods, which are described in detail as follows.

4.1 Model design

We designed two kinds of multimodal model schemes with different architectures : transformer single-flow structure and double-tower structure.

Transformer Single-flow Structure. The model of this structure only contains a set of longitudinal stacks of transformer modules. For image and text features, after their respective processing and stitching, together as the input of the model, so called single-stream transformer model. The advantage of this model is that it can express image and text cross-modal features in the same vector space, which is equivalent to mapping multi-modal features to single-modal vector representation space. The prediction effect of single-flow model is very dependent on pre-training. This work uses VisualBERT (Li et al., 2019) pretraining weights on COCO-VQA tasks. The text features directly use BERT-base-uncased (Devlin et al., 2019) tokenizer for word segmentation and post-vectorization. The image features are vectorized by ResNet-152 (He et al., 2016), EfficientNet-b2, EfficientNet-b4 and EfficientNet-b7 models (Tan and Le, 2019) to improve the richness of image feature expression. Finally, the image and the text vector are spliced together as the input of the model.

Double-tower Structure. Different from single-flow models, the double-tower model is actually two parallel models. For images and text, each builds a model structure and adds MLP to the output side to parallel them for joint training. Although the twin-tower structure cannot retain a large amount of prior knowledge through pre-training as the single-flow model does, nor can it fully excavate the cross-modal interaction characteristics as the twin-flow model does, the twin-tower structure is less dependent on the amount of training sample data, and is more flexible. Compared with the single-flow and double-flow transformer structures, it is more suitable for this task. Due to less training data, in order to avoid overfitting, we choose a relatively simple network structure, image side feature representation model using

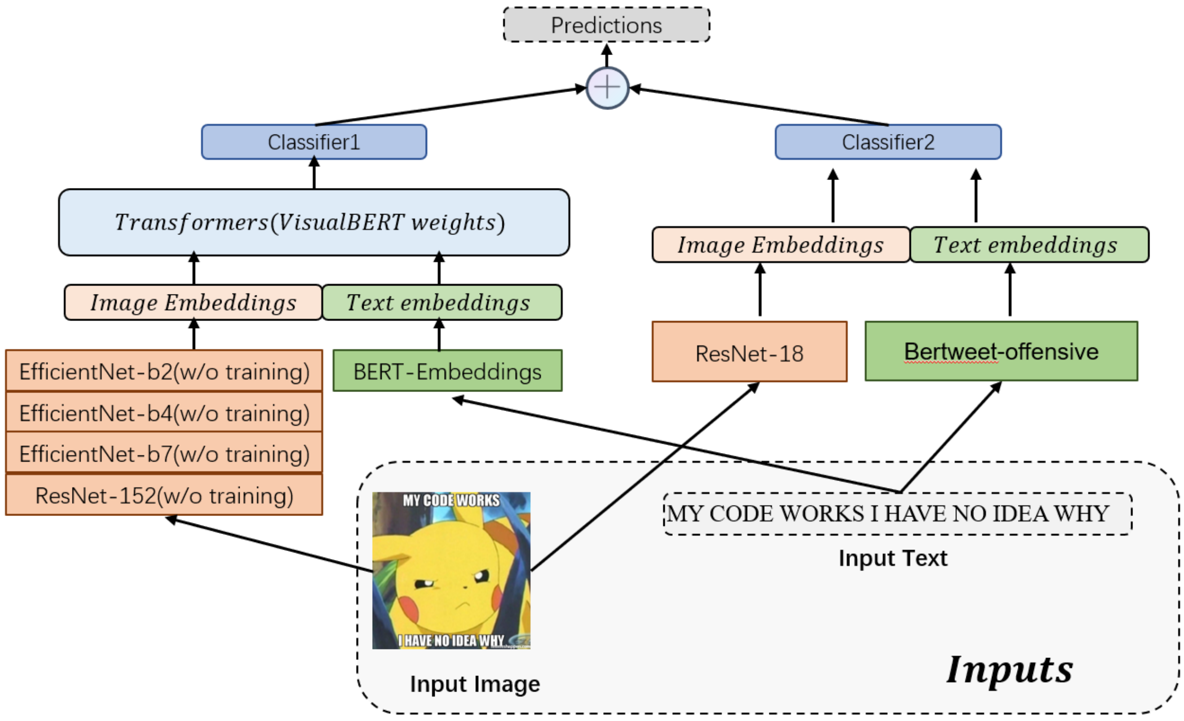


Figure 1: An overview of the two model structures. Transformer single-flow structure on the left and double-tower structure on the right.

ResNet-18 (He et al., 2016), text side feature representation model using Bertweet.

Model Ensemble. In order to make full use of the differences between models to achieve the best prediction effect, we use the weighted average method to obtain the final prediction results. The method is as follows:

$$Y_{pred} = \alpha \cdot Y_1 + (1 - \alpha) \cdot Y_2 \tag{1}$$

Where Y_{pred} is the final prediction probability, Y_1 is the single flow model prediction, Y_2 is the double tower model prediction, $\alpha = 0.1$.

4.2 Training Framework

As shown in Fig 2, we use a three-stage training framework consists of multi-task binary classification for sub-task B, single-task binary classification for sub-task A and post-processing.

Stage1: Multi-task binary classification. Sub-task A is a single-task binary classification problem, and sub-Task B is a multi-task binary classification problem. We tried to define sub-Task B as a single-task binary classification, so that the amount of data becomes five times that of multi-task binary classification to expand the training samples. However, the experimental results show that the effect is basically the same as that of multi-task binary classification. Therefore, we first define a multi-task binary task and obtain the respective prediction probabilities of the model for misogynous, stereotype, shaming, objectification and violence.

Stage2: Single-task binary classification. For sub-task A, we use external data from the memotion task (Sharma et al., 2020; Mem) for the misogynous binary classification. Although these data predict different targets, they are homologous to the task data. Through the analysis, it is found that there is almost no gender discrimination in the samples of 'Not Offensive' and 'Slight Offensive'. Therefore, this part of the external data is used to expand the training set, which greatly improves the prediction results of sub-task A. The reason why external data is not used in multi-task stage is that the proportion of positive samples in 'shaming', 'stereotype', 'objectification' and 'violence' tasks is very low, and increasing the number of negative samples will increase the prediction difficulty of these fields, so only external data is used in this stage.

Stage3: Post-processing. Through analysis, it is found that when 'shaming', 'stereotype', 'objectification' and 'violence' are positive samples, 'misogynous' must be positive samples. Due to the higher F1 value predicted by 'misogynous' in the

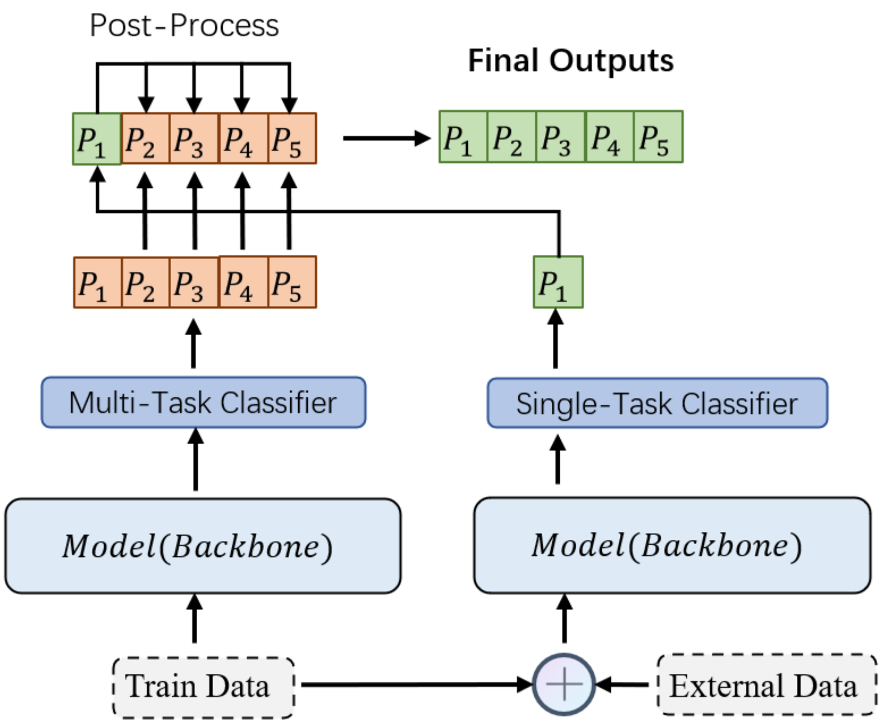


Figure 2: An overview of the proposed three-stage training framework. Firstly, we only use the training data to train the multi-task model and obtain the prediction probability of each task. Then, we use the external data to expand the training data to train the single task model and obtain the probability of 'misogynous'. Finally, we use the probability of 'misogynous' to post-process other tasks and obtain the final prediction results.

second stage, the prediction results of 'shaming', 'stereotype', 'objectification' and 'violence' are corrected by the prediction results of 'misogynous' in the second stage. We modify the prediction results of samples 'shaming', 'stereotype', 'objectification' and 'violence' with 'misogynous' prediction 0 to 0, which will improve the prediction results.

5 Experiments

This section is organized as follows. First we introduce the MAMI dataset (Elisabetta Fersini), then we introduce the experimental setup in detail. Finally, we show the effectiveness of our proposed method on MAMI datasets.

5.1 Datasets Analysis

The number of samples in the dataset is shown in table 1. Sub-task A uses external data, so the training data contains 10000 original samples and 14820 external data samples, and the predicted data contains 1000 samples; sub-task B does not use external data, so the training samples contain only 10000 original items and the predicted data also contains 1000 samples.

In the original 10000 training samples, there are five prediction labels, 'misogynous', 'shaming', 'stereotype', 'objectification' and 'violence'. Their respective number and proportion of positive samples are shown in table 2.

Dataset	Sub-task A	Sub-task B
train	10000	10000
external-train	14820	0
test	1000	1000

Table 1: Datasets statistics.

Category	Samples	Percentage
misogynous	5000	50.0%
shaming	1274	12.7%
stereotype	2810	28.1%
objectification	2202	22.0%
violence	953	9.5%

Table 2: Datasets analysis.

5.2 Experiment Settings

We implement our model using Pytorch. Using a workstation with an Intel Xeon processor, 64GB of RAM and a Nvidia P40 GPU for training. We apply AdamW as an optimization algorithm with

10% steps of warmup. For the hyperparameters, we set epochs=10, batch size=64. We also set the earlystop patience epoch as 3. We resize all images to 256 $\times$ 256 RGB pixels, and set the maximum cut length of text to 64. For the single-flow model, learning rate is set to 5×10^{-5}. For the double-tower model, we use the sub-regional learning rate strategy. The learning rate of text module is 5×10^{-5}, the learning rate of image module is 1×10^{-4}, and the learning rate of combined multi-layer percetron (MLP) module is 1×10^{-3}. We used multi-label stratified k-fold method to split training data into 5 folds.

Augmentation method used in image preprocessing. In the field of computer vision, because the image data have high dimensional characteristics, it is necessary to expand the training data, and usually do so. We use the following procedures to expand. In the training phase, we use (i) random resizing and cropping, (ii) random horizontal flipping, and (iii) random vertical flipping. In the inference phase, we use “five-crop inference” for robust prediction. This is essentially an average ensemble of the predictions on augmented images.

5.3 Experimental Results

Table 3 contains the results obtained from running the experiments. It can be seen that the prediction effect of single-flow model is worse than that of double-tower model, because the prediction effect of single-flow model is very dependent on pre-training. For the pre-training of single-stream model, most of the current mainstream models such as VisualBERT, UNITER (Chen et al., 2019) and so on are based on ImageNet (Deng et al., 2009), COCO (Lin et al., 2014) and other data sets for graph-text matching pre-training. There are some differences between these data sets and the scene of this task : (i) Twitter images contain text, and it is essential for emotional tendency expression, while ImageNet, COCO and other data sets contain almost no text ; (ii) The text of ImageNet, COCO and other data sets is the semantic expression of the image, and the text does not appear directly in the image, but the text information in the twitter sample is the text extracted from the image by OCR, which can be said to be a subset of image features from the feature information. The pre-training method of image-text matching is not very suitable. These reasons limit the prediction effect of the single-flow transformer structure model to a certain extent. Although compared with baseline, the potential of the model is still quite large. Sufficient training data and appropriate pre-training task design can further significantly improve the prediction results.In the single model, the twin-tower model has the best prediction results. Precisely because a particular module of the model contains only prior knowledge of a particular modality, it is more adaptable to different scenes. The cross-modal features are only learned through the current task, and do not depend on the prior knowledge. When the training data are not sufficient for pre-training, this modeling method can achieve the best results.

Method	**Sub-task A**	**Sub-task B**
organizers baseline	0.650	0.621
transformer single-flow	0.658	0.685
double-tower	0.684	0.706
model ensemble	0.682	0.707
post-processing	**0.746**	**0.708**

Table 3: Results on MAMI dataset.

Benefit from the huge difference in the feature representation of the two models, the difference of prediction results between single-flow model and double-tower model is very large, and better prediction results can be obtained by simple weighted average method. Finally, the model obtained by subtask A is used to post-process other prediction results, and the final best result is obtained.

5.4 Error Analysis

As shown in Table 4 and Fig 3, we select some typical samples that can reflect the performance of the methods for analysis. In Table 4, the first two columns are the file name and the text recognized in the image. SF, DT, ME and PP represent the prediction results of transformer single-flow, double-tower, model ensemble and post-processing methods in Table 2 on the misogyny task, respectively.

We can see that the transformer single-flow method is more dependent on text information than image. For some non-misogyny images, due to some gender-biased words like "she", "cute girl" and "sex", the transformer single-stream method mispredicts misogyny images, and the double-tower method can correctly distinguish them. For some misogyny images, the single-flow method fails to predict correctly because there is no misogyny tendency in the text.

file name	Text Transcription	SF	DT	ME	PP
Misidentified as misogyny under the influence of gender-biased words					
16233.jpg	SHE WONT CATCH YOU CHEATING ingtip.com IF YOU DONT CHEAT Opening Men Tut-Thur Fri-Sal Sunday	1	0	0	0
15888.jpg	when you ask a really cute girl out and she says yes	1	0	0	0
15458.jpg	Me letting my girlfriend know how incredibly beautiful she is You're breathtaking!	1	0	0	0
15567.jpg	Girls on the stairs Boys on the stairs	1	0	0	0
15402.jpg	GETS INVITED TO FIRST SEX PARTY 36 REALIZES IT FOR A BABY GENDER REVEAL meme memegenerator.net	1	0	0	0
Misogyny samples that were not identified due to the absence of misogyny words in the text					
16164.jpg	TSBURGH Bat TEELERS DOTBALL $65 A WILD SNORLAX APPEARS!! THE ULTRA BALLS Throw Them	0	1	1	1
15759.jpg	IWILL... SCRATCH YOU WITH EVERYTHING I'VE GOT !!	0	1	1	1
15729.jpg	WHEN IT'S 100 DEGREES @MAKEUPLOLZ BUT YOU STILL TRYNA APPLY FOUNDATION	0	1	1	1
15036.jpg	FRIEND ZONE LEVEL: INFINITY Virgin lvl over 9000	0	1	1	1
15963.jpg	ONE TEQUILA, TWO TEQUILA, THREE TEQUILA... FLOOR	0	1	1	1
Indistinguishable misogyny samples that are difficult to identify only based on image or text					
17013.jpg	1st year 3rd year 2nd year Final year	0	1	0	1
16085.jpg	When you leave her house after 2 hours of just kissing When you leave her house after 2 hours of just kissing banana in ...	0	0	0	1
15178.jpg	When you've been kissing for a half hour... Via MehsilyPre h.com After 30 mins of nonstop make out session	0	0	0	1
15999.jpg	When ur cleaning dishes and a chunk of food touches your hand @MARVY	1	0	0	1
15528.jpg	CABE THIS IS WHAT ALL MEN NEED TO SURVIVE Meme Center	0	1	0	1
Misidentified as a misogyny sample because of humor					
15952.jpg	HEY BABE CAN YOU MAKE ME A SANDWICH? Hey babe can you make me a sandwich? I should have bought the boat...	1	1	1	0
15502.jpg	JUST BOUGHT A NEW GUITAR THEN SHE SAID: ARE YOU GOING TO SELL THE OLD ONE? hengenerator.net	1	1	1	0
15795.jpg	GIRLFRIEND OFFERS TO WATCH FOOTBALL WITH YOU COMMENTS ON THE TEAM'S UGLY UNIFORMS	1	1	1	0
15460.jpg	IT'S A BIT COLD OUT BETTER PUT A HAT ON	1	1	1	0
16146.jpg	FOR MY NEXT TRICK ONEED YOUR FAVORITE SLIPPERS	0	1	1	0

Table 4: Error analysis of typical samples.

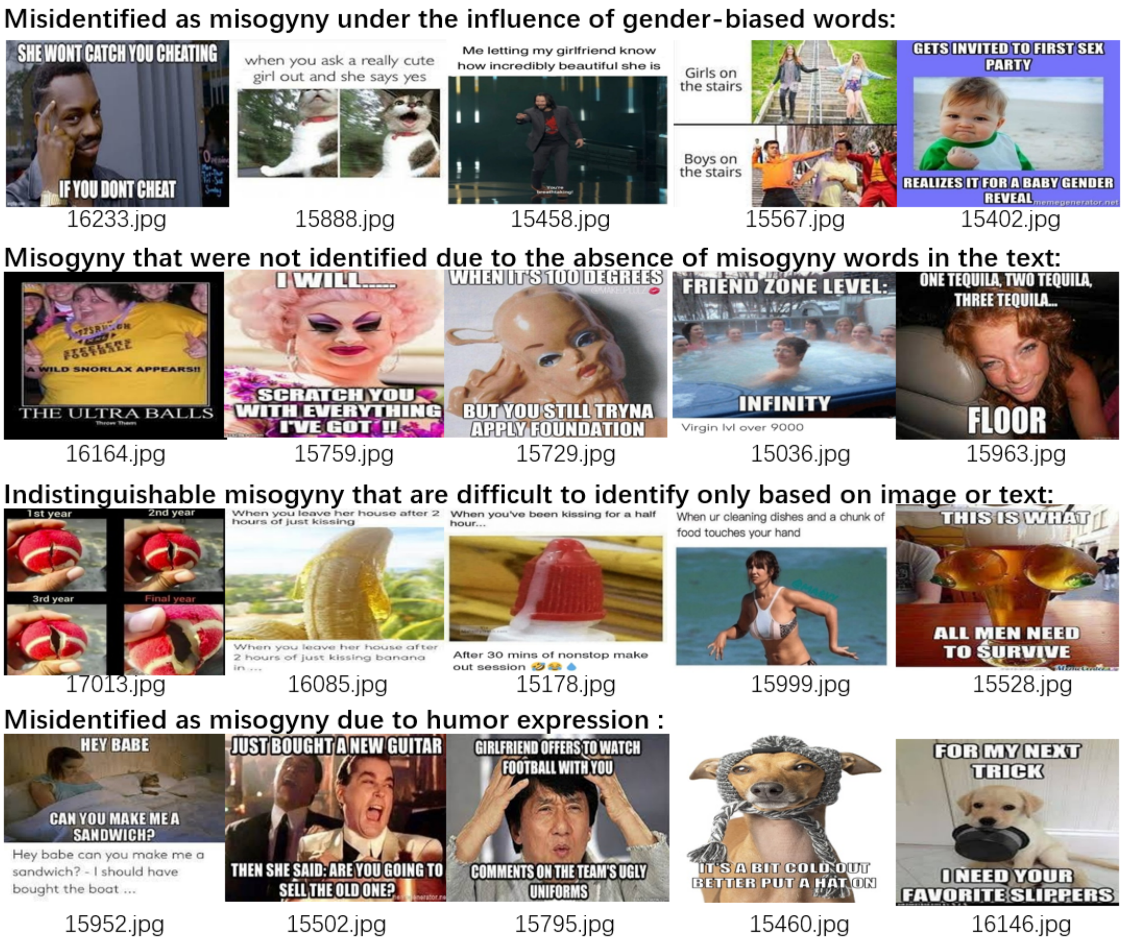

Misidentified as misogyny under the influence of gender-biased words:
16233.jpg
15888.jpg
15458.jpg
15567.jpg
15402.jpg
Misogyny that were not identified due to the absence of misogyny words in the text:
16164.jpg
15759.jpg
15729.jpg
15036.jpg
15963.jpg
Indistinguishable misogyny that are difficult to identify only based on image or text:
17013.jpg
16085.jpg
15178.jpg
15999.jpg
15528.jpg
Misidentified as misogyny due to humor expression :
15952.jpg
15502.jpg
15795.jpg
15460.jpg
16146.jpg

Figure 3: Error analysis of typical samples.

Many samples of misogyny are very obscure and do not tend to discriminate either from text or from images alone, but there is a combination of discriminatory hints that post-processing method is more capable of detecting such samples. At the same time, the post-processing method also calibrated some samples that were mistakenly identified as misogyny due to humor expression.

6 Conclusions and Future Works

In this paper, we describe the system developed by our team for SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. More specifically, we introduce two novel systems to analyze these posts: a multimodal multi-task learning architecture that combines Bertweet for text encoding with ResNet-18 (He et al., 2016) for image representation, and a single-flow transformer structure which combines text embeddings from BERT-Embeddings and image embeddings from several different modules such as EfficientNet (Tan and Le, 2019) and ResNet (He et al., 2016).In addition, we also use the model fusion and the post-processing method of prediction results, and improve the prediction results.

In the future, we will try to get more unlabeled sample data to pre-train the transformer single-flow method more fully and look forward to getting better multimodal methods in Twitter scenarios.

References

https://competitions.codalab.org/competitions/35688. In *AAAI-21 Workshop*.

Yusuf Aytar, Lluis Castrejon, Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. 2017. Cross-modal scene networks. *TPAMI*, 40(10):2303–2314.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. In *EMNLP*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Giulia Rizzi Aurora Saibene Berta Chulvi Paolo Rosso Alyssa Lees Jeffrey Sorensen Elisabetta Fersini, Francesca Gasparini. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *TACL*, 8:64–77.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *AAAI*, volume 34, pages 11336–11344.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized bert pretraining approach. *ICLR*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NIPS*, pages 13–23.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *ICML*.

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

Chhavi Sharma, Deepesh Bhageria, William Paka, Scott, Srinivas P Y K L, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Björn Gambäck. 2020. SemEval-2020 Task 8: Memotion Analysis-The Visuo-Lingual Metaphor! In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain. Association for Computational Linguistics.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. *ICLR*.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. In *ICCV*, pages 7464–7473.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR.

University of Hildesheim at SemEval-2022 task 5: Combining Deep Text and Image Models for Multimedia Misogyny Detection

Milan Kalkenings
University of Hildesheim
Hildeheim, Germany
kalkenin@uni-hildesheim.de

Thomas Mandl
University of Hildesheim
Hildesheim, Germany
mandl@uni-hildesheim.de

Abstract

This paper describes the participation of the University of Hildesheim at the SemEval task 5. The task deals with Multimedia Automatic Misogyny Identification (MAMI). Hateful memes need to be detected within a data collection. For this task, we implemented six models for text and image analysis and tested the effectiveness of their combinations. A fusion system implements a multi-modal transformer to integrate the embeddings of these models. The best performing models included BERT for the text of the meme, manually derived associations for words in the memes and a Faster R-CNN network for the image. We evaluated the performance of our approach also with the data of the Facebook Hateful Memes challenge in order to analyze the generalisation capabilities of the approach.

1 Introduction

Hate in Social Media continues to be a societal problem. The identification of problematic content based on text has made progress, but the performance is still not satisfying (MacAvaney et al., 2019; Modha et al., 2020b). Visual content and multi-modal construction on semantics is a reality in social media today (Dancygier and Vandelanotte, 2017). Systems for realistic scenarios in social media platforms (e.g. (Modha et al., 2020a) require image processing (Sai et al., 2022).

The Multimedia Automatic Misogyny Identification (MAMI) Challenge (SemEval-2022 task 5) is addressing this problem (Fersini et al., 2022). MAMI provides a testbed for algorithms which are capable of processing text and image of memes in one system. The experiments described in this paper measure the effectiveness of different models and their combination into a fusion system. We implemented a basic text classifier based on BERT and an image processing system based on the Faster R-CNN network. In addition, the generalization capabilities between collections are tested. We conducted the experiments with the MAMI dataset as well as with the dataset provided by the Facebook Memes Challenge (Kiela et al., 2021).

2 Previous Work

The detection of Hate Speech can be considered part of Natural Language Processing. Current research is driven by benchmark data and deep learning algorithms have shown to provide best performance.

Data sets such as Offensive Language Detection in Spanish Variants (MeOffendEs@IberLEF 2021) (Plaza-del Arco et al., 2021) and DEtection of TOXicity in comments In Spanish (DETOXIS) (Gonzalo et al., 2021) focus on general concepts of offensive content while other data sets are dedicated to more specific topics than general offensive content. The SemEval 2019 Task-5 (Basile et al., 2019) focused on the detection of hate speech against immigrants and women in Spanish and English messages extracted from Twitter. Besides the main binary task to detect hate speech, there was a fine grained task to further classify into aggressive attitude and the target harassed, to distinguish whether a message contains incitement against an individual rather than a group. The best performing system (Indurthi et al., 2019) trained a SVM model with a RBF kernel using Google's Universal Sentence Encoder (Cer et al., 2018) as features.

The shared task HASOC (Modha et al., 2019) created a large multilingual dataset for hate Speech identification. The first HASOC track focused on the identification of Hate Speech in Indo-European languages (Hindi, English and German). HASOC introduced a binary classification into problematic content and other content.

While most data sets include English data, several recent shared tasks have created new collections for other languages such as Greek (Pitenis et al., 2020) and Turkish (Çöltekin, 2020). Specific forms of Hate Speech based on the targeted groups have also been analyzed automatically. For this work, the detection of misogyny is especially relevant. In the Automatic Misogyny Identification (AMI) task at Evalita, a Twitter collection of misogynous messages was assembled. Overall, 10.000 tweets were available and classified into misogynous and non-misogynous tweets. They were also further analyzed into more fine-grained classes(Fersini et al., 2018). The second edition of the Automatic Misogyny Identification (AMI) task in 2020 followed up on binary classification. It also included the prediction of aggressiveness as a binary concept for the misogynous tweets and provided a subtask for the analysis of bias in the models (Fersini et al., 2020). Multi-modal processing of text and image simultaneously has made great progress recently. There are approaches for late fusion which first analyze image and text and combine the representations. Early fusion systems process both text and image in parallel in order to benefit from the dependencies. Systems like ImageBERT (Qi et al., 2020) and Uniter (Chen et al., 2020) have achieved promising results. Uniter relates text and image parts to one another and tries to capture their interaction.

3 Multimedia Datasets

The data for the SemEval-2022 task 5 (Multimedia Automatic Misogyny Identification, MAMI) is described in the overview paper (Fersini et al., 2022). The system presented here is aiming at a binary classification. It did not use the fine-grained classification on kinds of misogyny. In addition and in order to observe how well the multimodal system generalizes over similar datasets from similar tasks, we also processed the Facebook Hateful Meme Challenge (HM) dataset (Kiela et al., 2021). This dataset provides examples for hateful memes in general and includes also other kinds of problematic content than misogyny. However, because the tasks are related a system might also work across these two datasets. Table 1 gives an overview over the two sets. Another multimodal dataset is available for English. Some 700 memes related to the presidential election in the USA in 2016 were collected and annotated (Suryawanshi et al., 2020).

4 System Description

Our system includes six single models. They were all tested as classifiers and we explored several

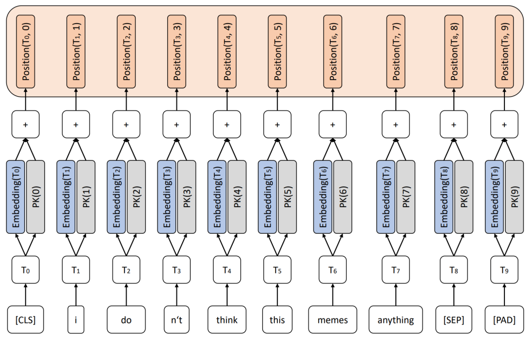


Figure 1: Transformer encoding of meme texts

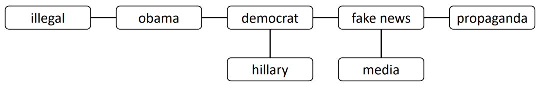


Figure 2: Examples for associations from the knowledge graph

combinations in the experiments.

4.1 Text classification with BERT

The first system is processing the text sequence associated with the meme (Devlin et al., 2019). We used the the model *bert-base-uncased* [1]. It creates a transformer based presentation of the text. The principle of BERT is illustrated in Figure 1.

4.2 Associations from Text

The association system includes semantic knowledge to enrich the representation. The assumption is that memes might often use words in another meaning than the obvious one. Looking for associations could help to enrich the representation. The associations might also be misleading. The associations were manually assembled into a knowledge graph. They were extracted after looking at many of the examples from the dataset and observing the intended meaning of many tweets.

For all words in the text sequence, the system looks for associated words in the knowledge graph. For example, the token *Obama* is related to *democrat* and to *illegal* (see figure 2). These relations represent world knowledge and prejudice which can be helpful for understanding the memes. A similar approach to incorporate knowledge graphs in classification systems has been taken by Liu et. al. (Liu et al., 2020). The approach is illustrated in Figure 3.

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

Feature	HM	MAMI
Number of memes	9.000	10.000
Number of hateful memes	3.300	5.000
Characters per text meme	62	101
Recognized objects per meme	2	2
Memes with recognized objects	7856	7777
Recognized associations per meme	2	2
Memes with recognized associations	3968	5419

Table 1: Dataset Statistics

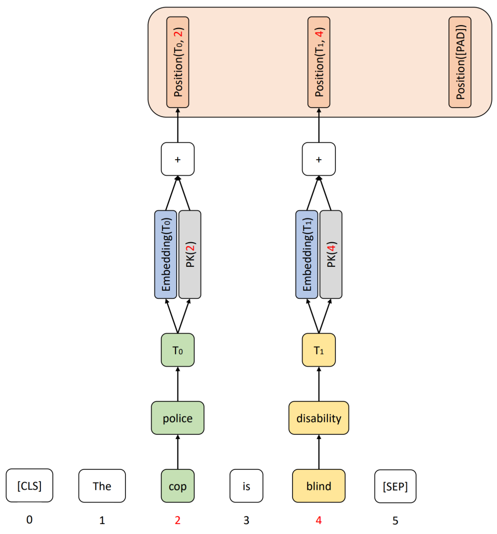


Figure 3: Graph encoding

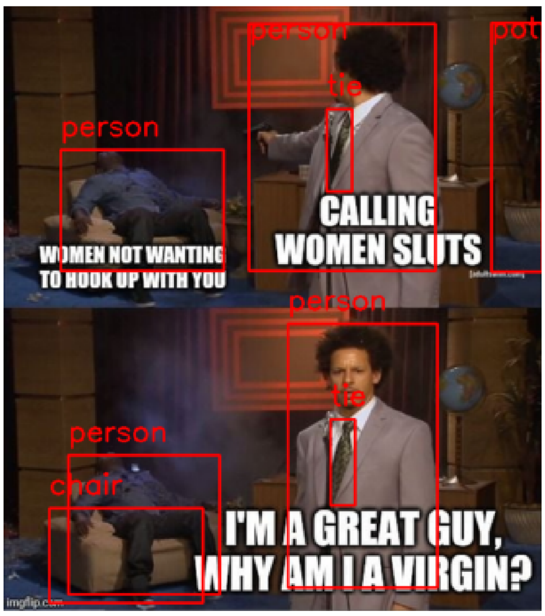

person
person
pot
tie
WOMEN NOT WANTING TO HOOK UP WITH YOU
CALLING WOMEN SLUTS
person
tie
person
chair
I'M A GREAT GUY, WHY AM I A VIRGIN?
imgflip.com

Figure 4: Object Detection for one meme

The tokens for the associations are extracted from the BERT model and given the position encoding of the word from the original text. All associations found are used. If more associations are found than the sequence length, then the last ones are cut off.

4.3 Sentiment Analysis in Text

In the approach presented here, sentiment analysis is used on the text of the meme. The rationale is that highly emotional texts could indicate a tendency toward hatefulness. Overall, six values are collected. The system VADER suggested by Hutto [2] is used to obtain the first two values for the overall sentiment and intensity (Hutto and Gilbert, 2014). The third and fourth value record the maximum and minimum values for sentiment for all tokens. The method of Loria [3] was used to obtain a measure of subjectivity and of sentiment.

[2]https://github.com/cjhutto/vaderSentiment

[3]https://textblob.readthedocs.io/

4.4 Faster R-CNN-Network

The first visual feature classifier is built with an object recognition system. It uses a Faster R-CNN-Network (Ren et al., 2017) as available online [4]. This system identifies interesting regions which contain much information. The visual features of these regions and their location embedding are fed into a ResNet system. Only the N most likely objects are used. Layer normalization is applied to obtain a final embedding. An example for a result of the object recognition for the dataset is given in figure 4.

4.5 Tile Approach for Image Analysis

A tile approach is splitting the image into 196 rectangular tiles of equal size. These are tiles are analyzed by CNNs and processed as suggested by (Dosovitskiy et al., 2021) and (Lin et al., 2021). The resulting feature vector is associated with a

[4]https://pytorch.org/vision/stable/models.html,

number indicating the position of the tile.

The ResNet architecture is used to obtain an embedding of the entire image. The output embedding is split in two parts of the same size which are used for further processing (Huang et al., 2020).

5 Experiments

For combining the single classifiers, they are fed into a fusion system. For that processing step, a transformer is used. After a layer normalization (Ba et al., 2016), all embedding values are concatenated and fed into a transformer. A sigmoid function is used for the final prediction. First, the models were tested individually. Then, combinations were tested. For finding the optimal fusion of the classifiers above, we applied Sequential Forward Selection (SFS) and Sequential Backward Elimination (SBE). For all experiments, the models were fully trained. Learning rates were adapted for each underlying model so that models converging faster did not overfit. Models with larger embeddings were assigned a higher dropout in the fusion system.

For SFS, all systems were tested individually first and the best system is used as the first component of the combined system. Afterwards, SFS iteratively adds further components. In each iteration, the current version of the combined system is extended by each individual system that is not part of the combined system yet. The combination that leads to the best improvement is taken as the new best combination. SFS converges, when no further improvement can be accomplished. SBE works similar to SFS but starts with a combination of all systems and iteratively detaches individual systems from the combined system. The submitted result was assigned the team name milan_kalkenings.

6 Results and Discussion

The main results refer to the training on the two datasets. Furthermore, we used the two datasets for training and testing respectively. These cross-dataset experiments are reported in the subsequent section.

6.1 Experiments within Datasets

First, the classification by each system individually was tested. As Table 2 shows, the best performance was given by the text classification system. It was followed by the associations system which is another system based on text analysis.

System	AUC-ROC
Text	0.6617
Sentiment	0.5706
Associations	0.6588
Image	0.5958
Tiles	0.5633
Object detection	0.5607

Table 2: Results for each system

The SFS selection method for the SemEval task 5 (MAMI) led to the following optimal combination: text, objects and associations (0.8509 AUC-ROC score). The SFS selection method for HM led to the following optimal combination: text, sentiment, associations and tiles (0.7136 AUC-ROC score). SBE led to the best performance for MAMI with the following combination: text, sentiment, associations and tiles (0.7136 AUC-ROC score). For HM, SBE gave best performance for this set of featues: text, associations and objects (0.8556 AUC-ROC score). The fusion led to improved scores as compared to processing one single modality. It is obvious, that text based metrics are more often in the optimal set.

Selection method	Systems	AUC-ROC
SFS	text, object detection and associations	0.8509
SBE	text, sentiment, associations and tiles	0.7136

Table 3: Results for fusion systems on the MAMI dataset

After the optimal fusion of single systems was determined, we obtained the performance on the test data. Table 4 reports the experiments for the optimal combination as well as for late fusion.

The HM dataset includes many benign confounders which either modify the text of a meme to

Experiment	AUC-ROC	Recall	Precision
HM	0.7146	0.6134	0.5334
HMlate	0.7069	0.3131	0.6712
MAMI	0.8421	0.8217	0.7261
MAMIlate	0.846	0.8529	0.7321

Table 4: Results of experiments with the two datasets

change its class or use the text of a hateful meme in another image. These were introduced to make the task more challenging (Kiela et al., 2021). Leaving out these duplicates changes the performance drastically. Leaving out the memes with identical text increases the performance by some 10%. On the contrary, leaving out the memes with identical images decreases the performance by some 20%. This shows again the impact of the text for this task.

6.2 Experiments Across Datasets

Across datasets, we first trained a classifier for distinguishing between the two datasets. That turned out to be fairly easy for the system (0.94 AUC-ROC). It seems that there are inherent features inserted during data creation which make that distinction easy for systems. Pretraining with the other dataset does not lead to a better performance overall. Only for the MAMI dataset, a performance close to the best overall performance was achieved.

7 Conclusion

The experiments have shown that the identification of hateful memes is still a challenging problem. In our experiments, text features are the most beneficial ones for the system. The influence of associations in particular needs to be further analyzed. First analysis seems to suggest that the number of found associations has a correlation with the performance for the problematic class. The performance across the two datasets is not optimal. Further datasets are needed to analyze the generalization across different collections. In future work, we intend to analyze the impact of each system for the fusion in more detail. We also plan to experiment whether training with a misogyny text dataset can be beneficial for a multimodal system.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proc.13th Intl. Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for english. In *Proc. Conf. on Empirical Methods in Natural Language Processing, EMNLP: Brussels, Oct. 31 - Nov. 4*, pages 169–174.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: universal image-text representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Çagri Çöltekin. 2020. A corpus of Turkish offensive language on social media. In *Proc. 12th Language Resources and Evaluation Conference, LREC Marseille, France, May 11-16*, pages 6174–6184. European Language Resources Association.

Barbara Dancygier and Lieven Vandelanotte. 2017. Viewpoint phenomena in multimodal communication. *Cognitive Linguistics*, 28(3):371–380.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop onSemantic Evaluation (SemEval)*. Association for Computational Linguistics.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018. Overview of the Evalita 2018 task on automatic misogyny identification (AMI). In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. AMI @ EVALITA2020: automatic misogyny identification. In *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), Online event, December 17th, 2020*, volume 2765 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Julio Gonzalo, Manuel Montes-y-Gómez, and Paolo Rosso. 2021. Iberlef 2021 overview: Natural language processing for Iberian languages. In *Proc. Iberian Languages Evaluation Forum (IberLEF) co-located with Conference of the Spanish Society for Natural Language Processing (SEPLN), XXXVII*, volume 2943 of *CEUR Workshop Proceedings*, pages 1–15. CEUR-WS.org.

Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. 2020. Pixel-BERT: Aligning image pixels with text by deep multi-modal transformers. *CoRR*, abs/2004.00849.

Clayton J. Hutto and Eric Gilbert. 2014. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014*. The AAAI Press.

Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. 2019. FERMI at SemEval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in Twitter. In *Proc. 13th Intl. Workshop on Semantic Evaluation*, pages 70–74, Minneapolis, Minnesota, USA. ACL.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Casey A Fitzpatrick, Peter Bull, Greg Lipstein, Tony Nelli, Ron Zhu, et al. 2021. The hateful memes challenge: competition report. In *NeurIPS 2020 Competition and Demonstration Track*, pages 344–360. PMLR.

Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, Jie Zhang, Jianwei Zhang, Xu Zou, Zhikang Li, Xiaodong Deng, Jie Liu, Jinbao Xue, Huiling Zhou, Jianxin Ma, Jin Yu, Yong Li, Wei Lin, Jingren Zhou, Jie Tang, and Hongxia Yang. 2021. M6: A Chinese multimodal pretrainer. *CoRR*, abs/2103.00823.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI, Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI, New York, NY, USA, Febr. 7-12*, pages 2901–2908. AAAI Press.

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. 2019. Hate speech detection: Challenges and solutions. *PLOS ONE*, 14(8):1–16.

Sandip Modha, Prasenjit Majumder, Thomas Mandl, and Chintak Mandalia. 2020a. Detecting and visualizing hate speech in social media: A cyber watchdog for surveillance. *Expert Syst. Appl.*, 161:113725.

Sandip Modha, Thomas Mandl, Prasenjit Majumder, and Daksh Patel. 2019. Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in Indo-European Languages. In *Working Notes of FIRE 2019 - Forum for Information Retrieval Evaluation, Kolkata, India, Dec. 12-15*, volume 2517, pages 167–190. CEUR-WS.org.

Sandip Modha, Thomas Mandl, Prasenjit Majumder, and Daksh Patel. 2020b. Tracking hate in social media: Evaluation, challenges and approaches. *SN Comput. Sci.*, 1(2):105.

Zesis Pitenis, Marcos Zampieri, and Tharindu Ranasinghe. 2020. Offensive language identification in Greek. In *Proc. 12th Language Resources and Evaluation Conference*, pages 5113–5119, Marseille, France. European Language Resources Association.

Flor Miriam Plaza-del Arco, Marco Casavantes, Hugo Jair Escalante, M Teresa Martín-Valdivia, Arturo Montejo-Ráez, Manuel Montes, Horacio Jarquín-Vásquez, Luis Villaseñor-Pineda, et al. 2021. Overview of MeOffendEs at IberLEF 2021: Offensive language detection in Spanish variants. *Procesamiento del Lenguaje Natural*, 67:183–194.

Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti, and Arun Sacheti. 2020. ImageBERT: Cross-modal pre-training with large-scale weak-supervised image-text data. *CoRR*, abs/2001.07966.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149.

Siva Sai, Naman Deep Srivastava, and Yashvardhan Sharma. 2022. Explorative application of fusion techniques for multimodal hate speech detection. *SN Comput. Sci.*, 3(2):122.

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, TRAC@LREC 2020, Marseille, May 2020*, pages 32–41. European Language Resources Association (ELRA).

Mitra Behzadi at SemEval-2022 Task 5 : Multimedia Automatic Misogyny Identification method based on CLIP

Mitra Behzadi and **Ali Derakhshan** and **Ian G. Harris**
Department of Computer Science
University of California, Irvine
Irvine, California, USA
mbehzadi@uci.edu , aderakh1@uci.edu , harris@ics.uci.edu

Abstract

Everyday more users are using memes on social media platforms to convey a message with text and image combined. Although there are many fun and harmless memes being created and posted, there are also ones that are hateful and offensive to particular groups of people. In this article present a novel approach based on the CLIP (Radford et al., 2021) network to detect misogynous memes and find out the types of misogyny in that meme. We participated in Task A and Task B of the Multimedia Automatic Misogyny Identification (MaMi) challenge (Fersini et al., 2022) and our best scores are 0.694 and 0.681 respectively.

1 Introduction

In the past few years more and more people have been using memes on social media platforms to express their thoughts and sometimes their beliefs. Although there are countless memes that are humorous and fun without expressing hate towards any certain group of people, there are also memes that aim to attack people.

Misogynistic memes use a combination visual and textual content to put down women and some of them are so violent that can be triggering to previous sexual abuse victims. That is why it would be so useful if these instances could be identified automatically.

One of the main challenges of this problem is that the images of these memes come in various forms. Also, the text on the memes adds occlusion to the objects in the images which makes the image understanding part of the problem even more challenging.

To address these challenges, we present a novel multi-modal classification approach based on CLIP (Radford et al., 2021) to identify misogynistic memes and also determine which 4 subcategories of shaming, stereotype, objectification or violence they belong to. CLIP is a multi-modal network trained for object detection. We use a multi-label classification method and detect misogynistic memes and all the subcategories using one single pipeline.

2 Related Work

There have been numerous researches conducted to solve hateful speech detection online in the past decade. Many of these approaches such as (Samghabadi et al., 2020) and (Cao et al., 2020) focus on only one modality which is text. More recently, researchers such as (Gomez et al., 2020) gathered multi-modal data-sets to be able to detect those instances of hate that go undetected by using only textual context.

Memes are also a form of multi-modal data that are widely used to indirectly communicate some meaning online. For the past few years, researchers have tried to solve the problem of hateful meme detection in various ways. In (Suryawanshi et al., 2020) a small data-set was gathered and an approach based on VGG16 (Simonyan and Zisserman, 2014) neural network was proposed. In 2020 Facebook gathered a the Hateful Meme data-set with 10,000 instances which was part of a Hateful Meme Detection challenge (Kiela et al., 2020). The winner of that challenge (Zhu, 2020) used an ensemble of visual-linguistic models such as Visual-Bert (Li et al., 2019) and Ernie-Vil (Yu et al., 2020) and fine-tuned them to solve the problem.

3 Proposed Method

3.1 Data Description

The Multimedia Automatic Misogyny Identification (MaMi) data-set which was gathered for SemEval Task5 challenge (Fersini et al., 2022), consists of 10,000 instances for training phase and 1000 instances for test. An example meme from this dataset is shown in Figure 1. Each instance has 5 binary labels depicting if it is misogyny, shaming, stereotype, objectification and violence.

It is important to note that these subcategories are not mutually exclusive, that is more than one label can be 1 for each instance. So we are dealing with a multi-label classification problem.

Label	Training Data	Test Data
Non-Misogynous	5000	500
Misogynous	5000	500
Shaming	1274	146
Stereotype	2810	350
Objectification	2202	348
Violence	958	153

Table 1: Training and Test Data-set Distributions

The challenge (Fersini et al., 2022) has two parts, in Task A the goal in to detect the misogynous memes. In the second part, Task B, the goal is to determine the type of misogyny that was present. We participated in both parts of the challenge using a multi-label classification scheme.

3.2 Classification Pipeline

In our proposed model, the pre-trained multi-modal object detection network CLIP has an important role. CLIP was initially introduced as multi-modal way of object detection. It was trained on 400,000 million pairs of images and text. It has proven to be much more efficient than many other state-of-the-art object detection techniques.(Radford et al., 2021)

We use CLIP to encode the image and text separately and concatenate the features as can be see in Figure 2 .The main idea is that there is a non-linear function $f(X_I, X_T)$ between the image feature space X_I and text feature space X_T that will help determine if an instance belongs to each one of the 5 categories or not. To find out the parameters of this non-linear function we create a feed-forward neural network and feed the concatenated features to that. The network has 5 output nodes, each for one of the labels.

Before feeding the image input to the CLIP image encoder we had to resize the image to 224x244 with 3 channels to match the input shape requirements. The text was also truncated to a sequence of length 77 to be seamlessly used with the CLIP tokenizer.

Additionally, we use Sigmoid layer as the activation function because in contrast to softmax, the probabilities of each instance belonging to a class do not have to sum up to 1 and so they can be independent of each other. Therefore, it is more suitable for multi-label classification. After getting the output of the pipeline, we determine the binary value for the predictions based on a threshold of 0.5.

3.3 Training Process

As no validation data-set was provided, we randomly split the 10,000 instances into 9000 for training and 1000 for validation purposes. We used binary cross-entropy loss function and used Adam optimizer for the process. The optimal hyper-parameters were found empirically with learning rate of 0.001, batch size of 128 and the training was done for 10 epochs.

At the end of each training epoch, the evaluation metrics including precision, recall and F1-score with macro averaging was calculated on the validation data-set. If a higher F1-score was found then the state of the model was saved as the best state.

As CLIP comes with different options for image feature extractions, we made sure to try two different ones, Visual Transformer (ViT) (Dosovitskiy et al., 2020) with 32x32 patches and Residual Network(He et al., 2016) with 101 layers to see if they have an impact on the classification results.

4 Evaluation and Results

4.1 Evaluation Metrics

All the submissions to the challenge were automatically evaluated by a script that was programmed by the organizers. After the challenge was over, the script was released and we investigated how the F1-scores for each task was calculated.

First the confusion matrix was calculated resulting in $M = \left(\begin{smallmatrix} tp & fp \\ fn & tn \end{smallmatrix}\right)$. Then, as shown in Equations 1 - 7, positive precision P^+, positive recall R^+, negative precision P^- and negative recall R^- was calculated separately and the final F1-score is the average of positive F1-score and negative F1-score.

$$P^+ = \frac{tp}{tp + fp} \tag{1}$$

$$R^+ = \frac{tp}{tp + fn} \tag{2}$$

$$F1Score^+ = \frac{2 \times (P^+ \times R^+)}{P^+ + R^+} \tag{3}$$

$$P^- = \frac{tn}{tn + fn} \tag{4}$$

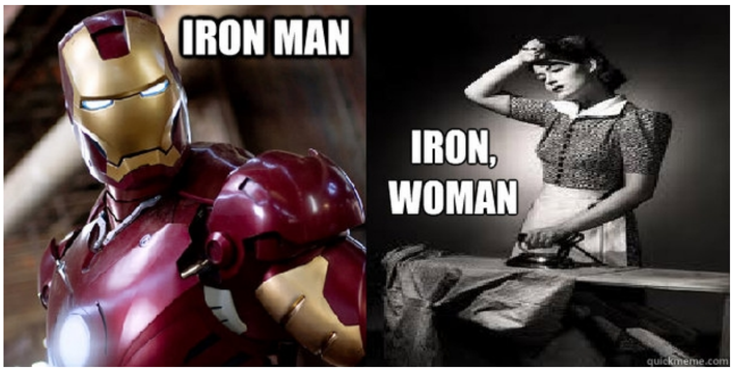


Figure 1: Sample number 152 of Training data which is Misogynous and Stereotype

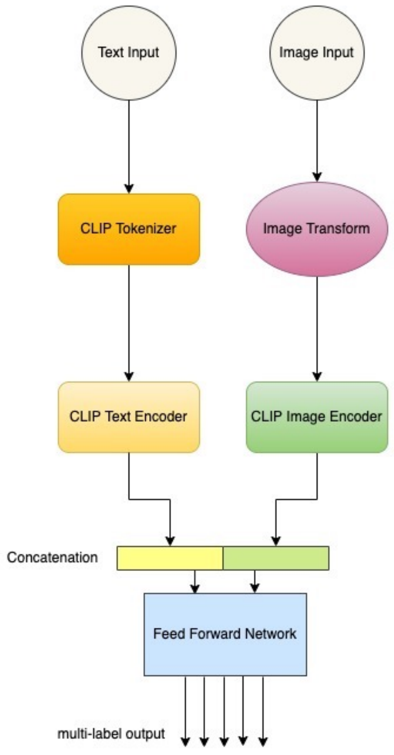


Figure 2: Proposed Classification Pipeline

$$R^- = \frac{tn}{tn + fp} \tag{5}$$

$$F1Score^- = \frac{2 \times (P^- \times R^-)}{P^- + R^-} \tag{6}$$

$$F1Score = \frac{F1Score^+ + F1Score^-}{2} \tag{7}$$

4.2 Experiments and Results

We conducted multiple experiments in the evaluation phase of the challenge. We tried different architectures for the feed forward network of our pipeline to get the best results.

In our initial experiments we had more layers in the feed forward network and did not include dropout layer and batch normalization. As the challenge progressed we realized that those models lacked in generalization, so we started using a simpler architecture and added dropout and batch normalization to get better results. Additionally, we tried switching the image encoder and discovered that using Resnet-101 based encoder results in much better scores, especially in Task B.

4.3 Error Analysis

As can be seen in the Table 2 our best result significantly outperforms all the baseline provided by the organizers. As informed by the organizers, the baseline models are grounded upon VGG-16 model for image feature extraction and USE model for textual feature extraction. Our best model achieves 0.694 score in Task A and 0.681 in Task B. It uses the pre-trained Resnet101 as the image encoder and also has 200 nodes in the hidden layer of our feed forward network.

5 Conclusion

In this paper we demonstrated how a successful model such as CLIP can be used to detect misogynous memes. We presented a novel architecture based on that multi-modal model and used multi-label training. We were able to achieve good results using this approach.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1813858. This research was also supported by a generous gift from the Herman P. & Sophia Taubman Foundation.

Our Models: Image Encoder - Feed Forward Network	**Task A F1-score**	**Task B F1-score**
ViT/32 - (1024,200,10,5)	0.678	0.497
ViT/32 - (1024,200,10,5) Dropout = 0.2	0.682	0.513
ViT/32 - (1024,200,5) Dropout = 0.2	0.681	0.629
ViT/32 - (1024,200,5) Dropout = 0.2 + Batch Norm	0.687	0.633
ViT/32 - (1024,100,5) Dropout = 0.2 + Batch Norm	0.674	0.639
ViT/32 - (1024,400,5) Dropout = 0.2 + Batch Norm	0.687	0.617
Resnet101 - (1024,200,5) Dropout = 0.2 + Batch Norm	**0.694**	**0.681**
Resnet101 - (1024,400,5) Dropout = 0.2 + Batch Norm	0.659	0.694
Resnet101 - (1024,100,5) Dropout = 0.2 + Batch Norm	0.693	0.673
Baseline Models		
Baseline Image	0.639	0.0
Baseline Text	0.640	0.0
Baseline Image-Text	0.543	0.0
Baseline Flat Multi-label	0.437	0.421
Baseline Hierarchical Multi-label	0.650	0.621

Table 2: Evaluation Results

References

Rui Cao, Roy Ka-Wei Lee, and Tuan-Anh Hoang. 2020. Deephate: Hate speech detection via multi-faceted text representations. In *12th ACM conference on web science*, pages 11–20.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1470–1478.

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

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *arXiv preprint arXiv:2005.04790*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Niloofar Safi Samghabadi, Parth Patwa, Srinivas Pykl, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and misogyny detection using bert: A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 32–41.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*, 1:12.

Ron Zhu. 2020. Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

IITR CodeBusters at SemEval-2022 Task 5: Misogyny Identification using Transformers

Gagan Sharma
Indian Insitute of Technology Roorkee
gagan_s@cs.iitr.ac.in

Gajanan Sunil Gitte*
Indian Insitute of Technology Roorkee
gajanan_s@cs.iitr.ac.in

Shlok Goyal*
Indian Insitute of Technology Roorkee
shlok_g@cs.iitr.ac.in

Raksha Sharma
Indian Insitute of Technology Roorkee
raksha.sharma@cs.iitr.ac.in

Abstract

This paper presents our submission to task 5 (Multimedia Automatic Misogyny Identification) of the SemEval 2022 competition. The purpose of the task is to identify given memes as misogynistic or not and further label the type of misogyny involved. In this paper, we present our approach based on language processing tools. We embed meme texts using GloVe embeddings and classify misogyny using BERT model. Our model obtains an F1-score of 66.24% and 63.5% in misogyny classification and misogyny labels, respectively.

1 Introduction

SemEval 2022 task 5 (Fersini et al., 2022) is a sentiment analysis task aimed at memes[1], divided into two subtasks of increasing complexity. Subtask A is misogynous meme identification, *i.e.*, whether a meme should be categorized either as misogynous or not misogynous; subtask B, on the other hand, is a multi-label classification problem, *i.e.*, classifying what kind of misogyny is involved in the meme. A meme is usually intended to convey a sarcastic message, but in a short time, people have started to use them to deliver sexist messages in an online environment. The anonymity of the internet tends to make them more aggressive. Such an environment amplifies the offline world's sexual stereotyping and gender inequality.

Meme sentiment analysis is a challenging task as often it relies on implicit themes or knowledge and trending news at the time of the creation of the meme. Meme analysis has been of growing interest for the NLP community. Our approach also relies on various NLP-based tools and the code to implement the paper is available at https://github.com/gagansh7171/IITR-CodeBusters .

*These two authors contributed equally.

[1]The term meme used in this task refers to an idea or a message conveyed via an image and embedded text.

The rest of the paper is organized as follows. In Section 2 we briefly describe the details regarding the datasets used. In Section 3 we describe some recent related work conducted in the field of meme classification. In Section 4 we describe and define the models and baselines for the task. In section 5 we describe the low-level details including scoring parameters, pre-processing, libraries and hyper-parameters used for the experimental setup. In Section 6 we describe the results obtained and some insight of why the models are performing the way they performed. In section 7 we conclude the paper.

2 Background

The dataset used for this task consists of a set of 10000 memes images whose text has already been extracted. Each meme has already been labeled as *misogynous* for subtask A and as *shaming*, *stereotype*, *objectification* ,and *violence* for subtask B. The dataset is perfectly balanced in terms of classification of misogyny, *i.e.*, precisely 50% of the memes are misogynous and the rest 50% are not. Of these 5000 misogynous memes -

- 1274 or 25.48% are labelled as shaming category.
- 2810 or 56.2% are labelled as stereotype category.
- 2202 or 44.04% are labelled as objectification category.
- 953 or 19.06% are labelled as violence category.

The data is provided as zip file of memes in image format and a .csv file where the memes are labelled according to the following structure (Fersini et al., 2022).

- **file_name:** name of the file denoting the meme

- **misogynous:** a binary value (1/0) indicating if the meme is misogynous or not. A meme is misogynous if it conveys an offensive message having as target a woman or a group of women.
- **shaming:** a binary value (1/0) indicating if the meme is denoting shaming. A shaming meme aims at belittling women because of some body characteristics.
- **stereotype:** a binary value (1/0) indicating if the meme is denoting stereotype. A stereotyping meme aims at representing a fixed idea or preconceived notion regarding women.
- **objectification:** a binary value (1/0) indicating if the meme is denoting objectification. A meme that describes objectification represents a woman like an object through over-analysis of physical or sexual appeal or comparing women to inanimate objects.
- **violence:** a binary value (1/0) indicating if the meme is denoting violence. A violent meme describes physical or verbal violence or harassment represented by textual or visual content.
- **Text Transcription:** transcription of the text reported in the meme.

3 Related Work

Memes are essentially language of the internet but such widespread use of memes had also made them a target for spreading hateful and offensive messages by fringe web communities (Zannettou et al., 2018). Many online communities accept sexism and harassment in the name of humour in the form of memes and has resulted in increased attraction of attention from academics (Drakett et al., 2018). Thus, memes had emerged as a multi-modal expression of online hate.

Research in the field of multi-modal sentiment analysis has been mostly focused on video and text or speech and text (Rao et al., 2021; Zadeh et al., 2016). Sentiment analysis of memes was conducted by French (2017) but it was based on correlation of the meme and online discussion in the comments.

Recent study for meme classification was done by Zia et al. (2021) where hateful memes were classified based on the protected category they attacked which were *race*, *sex*, *religion*, *nationality*, *disability*. The study included usage of state-of-the-art visual and textual representations to produce respective embedding of the memes which were then concatenated to train a logistic regression classifier model. Facebook recently launched The Hateful Memes Challenge to accelerate development in this field (Facebook, 2020).

We follow a more humble approach of using GloVe to embed text in the memes but tried models more sophisticated than Logistic Regression Classifier for the classification. The study conducted by Zia et al. (2021) reflects that better results emerge when both text and image are considered. But we do not consider the images for the challenge and consider this challenge an opportunity to learn about NLP first-hand.

4 System Overview

BERT[2] model is a state-of-the-art model developed by the AI team at Google (Vaswani et al., 2017). Traditional NLP models are unidirectional, *i.e.*, they read the text from left-to-right or right-to-left. On the other hand, BERT is bidirectional, understanding the correlation of a word with words on both sides by reading the entire sequence of the words at once. Google showed that this scheme helps better understand the statement's sentiment, making this model the best fit for the task.

RoBERTa[2] model is built on top of the BERT model. It has the same architecture but differs in terms of tokenizer and pre-training scheme, *i.e.*, much larger mini-batches and learning rates are used. The architecture similarity with the BERT model makes this model suitable for this task.

Baseline scores are made available by the organizers and are mentioned in Table 1. We provide additional baseline scores obtained by traditional classification models as well for comparison with the BERT and RoBERTa model. The different baseline models used are Logistic Regression (LR)[3], K Nearest Neighbours (KNN)[3], Random Forest (RF)[3] and Multilayer Perceptron (MP)[3].

5 Experimental Setup

The scoring parameters, pre-processing, language, libraries and hyper-parameters used are mentioned in this section for ease in reproduction.

[2]We use transformers 4.16.2 implementation of the model.
[3]We use Scikit-learn 1.0.2 implementation of the model.

Model	A	B
Baseline_Image_Text	54.3%	0%
Baseline_Text	64%	0%
Baseline_Image	63.9%	0%
Baseline_Flat_Multilabel	43.7%	42.1%
Baseline_Hierarchical_M	65%	62%

Table 1: Baseline Scores obtained by SemEval Organizers.

5.1 Scoring parameters

Macro F1 score is used as a measure of performance for the models for subtask A and weighted average of F1 scores for each prediction category for subtask B.

5.2 Pre-processing

The text is converted to lowercase before being used for training. Each meme is vectorised using GloVe embedding (Pennington et al., 2014).

Glove consists of word to vectors mapping and these vectors come in various dimensions. We are using 200-d vectors. For embedding one meme we find out word-vectors for every word and calculate an average of these vectors to finally calculate a vector representation of a meme. Glove vectors map words to a point in n-dimensional space where words with similar meaning are closer to each other. So taking average of these vectors should give us a vector in this space which represent an average gist of the message.

5.3 Language and libraries used

The experiment is conducted at Google Colab platform using Python 3.7.2 as the programming language. The packages are listed in Table 2

Package	Version
Pandas	1.3.5
Jupyter	1.0.0
Keras	2.7.0
Tensorflow	2.7.0
Tensorflow-hub	0.12.0
Numpy	1.19.5
Transformers	4.16.2
Scikit-learn	1.0.2

Table 2: List of Python Packages used for the Experiment.

5.4 Hyper-parameters of the models

This section details hyper-parameters for each model.

K Nearest Neighbours 5 CPU jobs, other values are default from Scikit-learn (refer KNN).

Logistic Regression random seed of 42, solver is liblinear, maximum iterations of 1000, 5 CPU jobs, f1_macro scoring is used, refit is set to True, other values are default from Scikit-learn (refer LogisticRegressionCV).

Multilayer Perceptron maximum iteration is set to 200, other values are default from Scikit-learn (refer MLPClassifier).

Random Forest 5 CPU jobs, bootstrap samples are used while building trees, out-of-bag samples are used to estimate the generalization score, 10 trees are used in the forest, all features are considered while looking for best split, other values are default from Scikit-learn (refer RandomForestClassifier).

BERT transformers based implementation of BERT is used. Pretrained bert-base-uncased model and tokenizer are used. Adam optimizer with 3e-6 learning rate, 1e-08 epsilon and 1.0 clipnorm as parameters is used as the optimizer. Sparse Categorical Crossentropy with from_logits set to True is used as loss function. Early stopping is done with a patience value of 4 and minimum increment of validation accuracy as 0.005. The model which gave best result for validation set is restored before training is complete. Other values are default from Hugging-Face implementation (refer BERT).

RoBERTa transformers based implementation of RoBERTa is used. Pretrained roberta-base model and tokenizer are used. Rest of the parameters are same as used in BERT model. Other values are default from Hugging-Face implementation (refer RoBERTa).

6 Results

Models are trained with 80% of data as training data and 20% of data as validation data. The results obtained for subtask A and B are mentioned in Table 3 and 4 respectively. The models' hyper-parameters were tuned for subtask A and were reused for subtask B, hence training and validation scores are omitted in Table 4, instead score for each classification category[4] and final scores are listed. Baseline results obtained by the SemEval

[4]These scores are obtained in post-evaluation phase after the labels for test data were released by the organizers.

Model	Training Score	Validation Score	Subtask A
KNN	78.05%	64.27%	57.54%
LR	73.95%	71.03%	58.69%
MP	89.97%	71.32%	58.05%
RF	98.56%	61.97%	58.81%
BERT	**87.26%**	**82.10%**	**66.24%**
RoBERTa	88.63%	80.60%	60.80%

Table 3: Scores obtained for subtask A.

Model	Misogynous	Shaming	Stereotype	Objectification	Violence	Subtask B
KNN	57.54%	53.62%	58.24%	54.51%	56.38%	56.50%
LR	58.69%	53.35%	55.63%	51.52%	52.65%	55.17%
MP	58.06%	52.30%	54.50%	60.66%	62.66%	57.74%
RF	58.88%	48.01%	54.19%	45.04%	47.16%	52.30%
BERT	**66.24%**	**64.06%**	**61.75%**	**65.20%**	**66.51%**	**64.76%**
RoBERTa	60.80%	46.06%	62.10%	66.11%	53.36%	60.14%

Table 4: Scores obtained for subtask B.

Organizers are listed in Table 1 for comparison.

From the results we can see that Random Forest model has high level of over-fitting as the score difference of training and validation is highest in this model.

The least difference in training and validation score appears in Logistic Regression model but this is mostly due to high level of under-fitting as the least training score is obtained in this model. BERT model provides the best results. The model obtains a high training score and score difference in training and validation is the least for this model. The trend maintains in subtask B as well with the model obtaining highest score.

Ranking phase - We submit our scores obtained from BERT model for the leader board. The scores obtained are 66.24% and 63.5%[5] for subtask A and B respectively. The ranks obtained for subtask A and B are **47** and **34** respectively.

7 Conclusion

The results show that BERT model is the best among those we tried for the given problem statement. However, these results are obtained using purely NLP based tools and techniques, the image component of the memes is not considered. As discussed in the Background section, there is a scope of improvement if we consider the images of the memes as well for the classification.

[5]The score of 64.76% mentioned in the table is obtained using the same model used in making the submission for evaluation phase. This score is obtained post-evaluation with test-labels being made public by the organizers.

References

Jessica Drakett, Bridgette Rickett, Katy Day, and Kate Milnes. 2018. Old jokes, new media – online sexism and constructions of gender in internet memes. *Feminism & Psychology*, 28(1):109–127.

Facebook. 2020. Hateful memes challenge and dataset for research on harmful multimodal content.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jean H. French. 2017. Image-based memes as sentiment predictors. In *2017 International Conference on Information Society (i-Society)*, pages 80–85.

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

Ashwini Rao, Akriti Ahuja, Shyam Kansara, and Vrunda Patel. 2021. Sentiment analysis on user-generated video, audio and text. In *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pages 24–28.

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

Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. 2016. Multimodal sentiment intensity analysis in videos: Facial gestures and verbal messages. *IEEE Intelligent Systems*, 31(6):82–88.

Savvas Zannettou, Tristan Caulfield, Jeremy Blackburn, Emiliano De Cristofaro, Michael Sirivianos, Gianluca Stringhini, and Guillermo Suarez-Tangil. 2018. On the origins of memes by means of fringe web communities. In *Proceedings of the Internet Measurement Conference 2018*, pages 188–202, New York, NY, USA.

Haris Zia, Ignacio Castro, and Gareth Tyson. 2021. Racist or sexist meme? classifying memes beyond hateful. pages 215–219.

IIT DHANBAD CODECHAMPS at SemEval-2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification

Shubham Kumar Barnwal
Department of Computer Science and Engineering , Indian Institute of Technology (Indian School of Mines), Dhanbad, India
shubham.developer02@gmail.com

Ritesh Kumar
Department of Computer Science and Engineering, National Institute of Technology Jamshedpur, India
ritesh.cse@nitjsr.ac.in

Rajendra Pamula
Department of Computer Science and Engineering , Indian Institute of Technology (Indian School of Mines), Dhanbad, India
rajendrapamula@gmail.com

Abstract

With the growth of the internet, the use of social media based on images has drastically increased like Twitter, Instagram, etc. In these social media, women have a very high contribution as of 75% women use social media multiple times compared to men which is only 65% of men uses social media multiple times a day. However, with this much contribution, it also increases systematic inequality and discrimination offline is replicated in online spaces in the form of MEMEs. A meme is essentially an image characterized by pictorial content with an overlaying text a posteriori introduced by humans, with the main goal of being funny and/or ironic. Although most of them are created with the intent of making funny jokes, in a short time people started to use them as a form of hate and prejudice against women, landing to sexist and aggressive messages in online environments that subsequently amplify the sexual stereotyping and gender inequality of the offline world. This leads to the need for automatic detection of Misogyny MEMEs. Specifically, I described the model submitted for the shared task on Multimedia Automatic Misogyny Identification (MAMI (Fersini et al., 2022)) and my team name is IIT DHANBAD CODECHAMPS.

1 Introduction

With the growth of the internet, social media becomes a crucial part of everyone's life. As every coin has two side positive and negative, social media also comes with a number of problems. The challenge of identifying misogyny (Srivastava et al., 2017) in different social media specially in forms of meme which contains both image and text is very complicated. Misogyny meme highly affected the life of women's as its spread hate and prejudice behaviour against women's. Social media like twitter, Instagram, etc have handled by their own ways. However, detecting such memes is highly challenging. Due to this challenge, it attracts the researcher's attention. According to one social media Instagram, more than 1 million users shared memes daily. So, with this huge amount of data in social medias and internet it is impossible to detect every misogyny meme by man power. So, we need machine learning, deep learning and artificial intelligence techniques to detect automatically misogyny memes in social media. In this paper, we have explored various Machine Learning (ML) and Deep Learning (DL) algorithms for misogyny identification in shared task MAMI (Fersini et al., 2022) challenge and my team's name is IIT DHANBAD CODECHAMPS. As per requirement of MAMI, I have submitted 4 runs for Subtask-A. My best run in Subtask-A has achieved Macro-F1 score of 0.656.

2 Related Works

Many works related to automatic detection of misogyny, hate, sexism on social media and web have been proposed.
Abir Rahali (Rahali et al., 2021) proposed a approach for automatic misogyny detection in social media using attention based bidirectional LSTM.
Endang Wahyu Pamungkas (Pamungkas et al., 2020) proposed a method for Automatic Identification of Misogyny in English and Italian Tweets at EVALITA 2018 with a Multilingual Hate Lexicon.
Mario Anzovino, Elisabetta Fersini (Anzovino et al., 2018) proposed a method for Automatic Identification and Classification of Misogynistic Language on Twitter. The main contribution of this paper is two-fold: (1) a corpus of misogynous tweets, labelled from different perspective and (2) an exploratory investigation on NLP features and ML models for detecting and classifying misogynistic language.
Rachael Fulper (Fulper et al., 2014) proposed a relation between misogynistic language in twitter and sexual Violence. In their paper they consider all 50 states in Washington DC.

Lakes Goenaga, Aitziber (Goenaga et al., 2018) Atutxa proposed a Automatic misogyny identification using neural networks. In this paper they focus on recurrent neural network (RNN) approach using a Bidirectional Long Short Term Memory (Bi-LSTM).

3 Task and Dataset Description

Here we have described the dataset and task provided by Multimedia Automatic Misogyny Identification (MAMI (Fersini et al., 2022)) challenge.
Multimedia Automatic Misogyny Identification (MAMI) task is divided into two sub task. Sub-task A: a basic task about misogynous meme identification, where a meme should be categorized either as misogynous or not misogynous (shown in Table 1).

Sub-task B: an advanced task, where the type of misogyny should be recognized among potential overlapping categories such as stereotype, shaming, objectification and violence. *e.g.*

1026.jpg 10101 POV: You’re my wife made with mematic

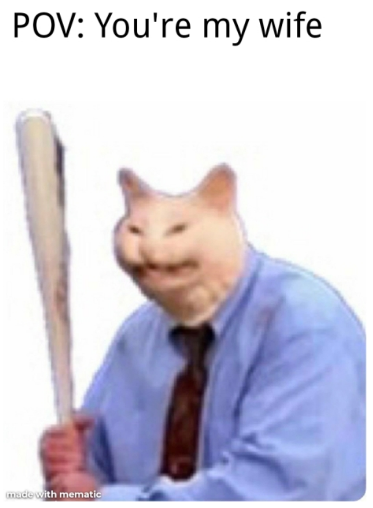


Figure 1: 1026.jpg

Here, 1026.jpg represent meme file name . Next column contains 5 numbers of zeros and ones . First numbers represent wheather meme is misogynous . Second numbers represent wheather meme is shaming . Third numbers represent wheather meme is stereotype .Fourth numbers represent wheather meme is objectification .Fifth numbers represent wheather meme is violence . Next column represent Text Transcription of the meme.

4 Methodology

4.1 Text Preprocessing

First, we removed all the punctuations, numbers, links and stop words. We have used lemmatization for grouping together the different forms of a word into a single word. NLTK wordnet (Loper and Bird, 2002) is used for lemmatization.

4.2 Feature Extraction

TfidfVectorizer (Kumar and Subba, 2020) is used for converting the text into numerical features. Pipeline [1] is used for doing TfidfVectorizer and classification in pipelined manner. Tokenizer by keras library is used for LSTM and Bert. For Logistic regression and SVM we have used TfidfVectorizer from scikit-learn library.

Models Proposed

For Subtask-A, we have submitted 4 runs based on four different algorithms, namely- Logistic Regression (Sammut and Webb, 2010), SVM (Noble, 2006), LSTM (Hochreiter and Schmidhuber, 1997), Bert (Devlin et al., 2018) with different parameters like batch size, epochs, number of perceptron etc. We have used the scikit-learn library for logistic regression based models and SVM (support vector machines) models. Keras is used for LSTM and BERT. We scored maximum F1 score 0.656 using BERT. We have used the following value of parameters :-
1.For TfidfVectorizer, we have used mindf=20, maxfeatures=2000 and maxdf=0.6 .
2. For LSTM and BERT, we have used batch size = 2, epochs = 3 and number of layers = 2 .

5 Result and Discussions

The results of Subtask-A are represented in terms of Macro-F1 (shown in Table 2). The best score as Macro-F1 for Subtask-A we get is 0.656. Table 2 shows the score of our submissions based on different algorithms on MAMI challenge official ranking.
For Subtask-A BERT performs better than all other models with the parameters batch size = 2 , epochs = 3 , number of hidden layers = 2 and number of perceptron’s is 128 in first layer and 64 in second layer.

[1] `shorturl.at/eqrFW`

file_name	misogynous	Text transcription
1.jpg	0	Milk Milk.zip
10000.jpg	0	MAN SEEKING WOMAN Ignad 18 O
1026.jpg	1	POV: You're my wife made with mematic
10014.jpg	0	HOOKER MY DICK VIRGIN ME

Table 1: Categories of MEMEs with Examples for Subtask-A

Model	F1 Score
BERT	**0.656**
Logistic	0.631
SVM	0.584
LSTM	0.651

Table 2: Result of Subtask-A based on different Models

6 Conclusions and Future Work

We have completed the task using various classification algorithms and evaluated the performance of different classification algorithms for Multimedia Automatic Misogyny Identification (MAMI) shared task. Our overall score is 0.656 for subtask-A which were average as compared to other submissions obtained in the Multimedia Automatic Misogyny Identification (MAMI) shared task. We look forward to experimenting with different advance algorithm or neural network models. Also, till now our algorithms works only with text classification. We are looking forward to work in text and image simultaneously for better accuracy and classification. Also, fine tuning the parameters of the algorithm can help in improvement of the overall performance. We shall be exploring these tasks in the coming days.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Rachael Fulper, Giovanni Luca Ciampaglia, Emilio Ferrara, Y Ahn, Alessandro Flammini, Filippo Menczer, Bryce Lewis, and Kehontas Rowe. 2014. Misogynistic language on twitter and sexual violence. In *Proceedings of the ACM Web Science Workshop on Computational Approaches to Social Modeling (ChASM)*, pages 57–64.

Iakes Goenaga, Aitziber Atutxa, Koldo Gojenola, Arantza Casillas, Arantza Díaz de Ilarraza, Nerea Ezeiza, Maite Oronoz, Alicia Pérez, and Olatz Perez de Viñaspre. 2018. Automatic misogyny identification using neural networks. In *IberEval@SEPLN.*

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

Vipin Kumar and Basant Subba. 2020. A tfidfvectorizer and svm based sentiment analysis framework for text data corpus. In *2020 National Conference on Communications (NCC)*, pages 1–6.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *arXiv preprint cs/0205028.*

William S Noble. 2006. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

Abir Rahali, Moulay A Akhloufi, Anne-Marie Therien-Daniel, and Eloi Brassard-Gourdeau. 2021. Automatic misogyny detection in social media platforms using attention-based bidirectional-lstm. pages 2706–2711.

Claude Sammut and Geoffrey I. Webb, editors. 2010. *Logistic Regression*, pages 631–631. Springer US, Boston, MA.

Kalpana Srivastava, Suprakash Chaudhury, PS Bhat, and Samiksha Sahu. 2017. Misogyny, feminism, and sexual harassment. *Industrial psychiatry journal*, 26(2):111.

QiNiAn at SemEval-2022 Task 5: Multi-Modal Misogyny Detection and Classification

Qin Gu, Nino Meisinger, Anna-Katharina Dick
University of Tuebingen
{firstName.lastName}@student.uni-tuebingen.de

Abstract

In this paper, we describe our submission to the misogyny classification challenge at SemEval-2022. We propose two models for the two subtasks of the challenge: The first uses joint image and text classification to classify memes as either misogynistic or not. This model uses a majority voting ensemble structure built on traditional classifiers and additional image information such as age, gender and nudity estimations. The second model uses a RoBERTa classifier on the text transcriptions to additionally identify the type of problematic ideas the memes perpetuate. Our submissions perform above all organizer submitted baselines. For binary misogyny classification, our system achieved the fifth place on the leaderboard, with a macro F1-score of 0.665. For multi-label classification identifying the type of misogyny, our model achieved place 19 on the leaderboard, with a weighted F1-score of 0.637.

1 Introduction

Even though women are as much present online as men, some online spaces, such as microblogging websites, are still male dominated. Misogynistic jokes and memes are inevitably somewhat common in certain parts of the Internet and have the potential to perpetuate harmful ideas about gender or instill false ideas and expectations. Notably, the way they are spread is often through various modalities, most commonly visual and textual. Therefore, it would be useful to have a system that could automatically detect if certain combinations of texts and images are misogynistic or not. This is not a trivial task however, since misogyny can manifest in many different forms such as stereotyping, objectifying or threatening violence against women. A crucial difficulty in this multi-modal classification task is also the interplay between text and image. Some memes may appear harmless if only either the image or text are viewed separately. The Multimedia Automatic Misogyny Identification (MAMI) challenge at SemEval-2022 (Fersini et al., 2022) seeks to find solutions to solve this task.

We propose two models that automatically detect English misogynistic memes and classify the type of problematic ideas they perpetuate.[1] For simple binary misogyny detection, we created an ensemble model that makes predictions based on majority voting on two text-based and two image-based classifiers using partly hand-crafted features such as age, gender and nudity classification. For task B we relied only on text information and used a transformer based approach, creating a RoBERTa model (Liu et al., 2019) that classifies the type of misogyny that is perpetuated in the memes.

2 Background

Automatically identifying misogynistic texts has been explored in the past. In 2020, this task was proposed as an EVALITA shared task, using Italian tweets (Fersini et al., 2022). So far, the research in the area of misogyny detection has mostly focused on pure text data from social media, specifically Twitter (Anzovino et al., 2018; Frenda et al., 2019). In the current multi-modal task however, competitors were given the opportunity to explore classification using both visual and textual data given. Singh et al. (2020) used a multi-modal multi-task learning system with BERT (Devlin et al., 2018) features to extract textual information and ResNet features to handle image classification. A similar approach proved to be useful in another related shared task, in which Tamil memes should be identified as trolling or not (Suryawanshi and Chakravarthi, 2021).

[1]The code is made available at https://github.com/cicl-iscl/SemEval-2022_Multimedia_Automatic_Misogyny_Identification.

shaming	THE FACE YOU MAKE WHEN TRUMP HAS THE CLASSY FOREIGN CHICK AND YOU'RE MARRIED TO KERCHAK imgrip.com
stereotype	creator made native women beautiful, to hide all that crazy
objectification	When my girlfriend is trying to have a serious conversation with me TITTY
violence	ROSES ARE RED, VIOLETS ARE BLUE IF YOU DON'T SAY YES, I'LL JUST RAPE YOU quickmeme.com
shaming + stereotype	CAN'T TELL IF THIS IS AN UGLY HIPPY CHICK OR REALLY PRETTY HIPPY BOY quickmeme.com
stereotype + objectification	Keeping your dishwasher clean will make it last longer Take care of your appliances
objectification + violence	inglip.com IS RAPING A PROSTITUTE A THEFT?

Table 1: Examples of text transcriptions that fit the four categories of misogyny and combinations of multiple labels.

The provided data set includes both images and text transcriptions of memes. The two subtasks we participated in were structured as follows: In subtask A, the memes should simply be identified as misogynous or not misogynous. Subtask B posed a more advanced challenge, as memes should additionally be classified as being part of four overlapping categories. These categories specify the type of misogyny expressed in the meme: stereotypes, shaming, objectification and violence. Shaming memes will insult women's appearance, stereotypes perpetuate harmful ideas about (groups of) women, objectification reduces women to their sexuality or body and violence downplays or advocates for violence against women. Examples of text transcriptions of these categories can be found in Table 1. The dataset for training includes 10000 memes with all of these labels, with exactly half of the memes being classified as misogynous and half as harmless. 2810 memes in the training set perpetuate stereotypes, 2202 objectification, 1274 shaming, and 953 violence. Many of the misogynistic memes have therefore overlapping categories, being classified as two or even three or all types of misogynous.

3 Subtask A: Binary misogyny classification

3.1 System Overview

For the binary classification of memes we experimented with a number of classification algorithms for both the image and the text transcriptions, as well as with a variety of different features. We decided to examine the benefits of an ensemble model that uses more traditional forms of text classification, as optimized ensemble models have been shown to perform well on similar tasks such as hate speech detection (Van Thin et al., 2019). The resulting model consists of four estimators, as illustrated in Figure 1.

The first set of classifiers are a multinomial naive Bayes classifier and a separate Gradient boosting classifier with tf-idf transformed vectors of the text transcriptions found in the data set. Originally we considered training the models on n-gram features, as previous research has shown that they can be useful to classify short texts (Buda and Bolonyai, 2020), but tf-idf vectors consistently performed better. Similarly, we experimented with adding more classifiers trained on tf-idf vectors, however, performance stayed consistent or decreased, thus we stayed with two.

Secondly, we introduce a Random forest classifier trained on a variety of features pertaining to the images themselves, rather than the text. The features used are Hu moment invariants (Hu, 1962), Haralick textures (Haralick et al., 1973) and image histograms. All three features are expected to help in gaining general insights into the image structure of the meme: Hu moment invariants are used to characterize the shape of an object in an image, Haralick textures should provide information about regions of interest, and the image histogram are employed to gain information about the color distribution, as the former two features require images to be converted into grayscale. To make all images equivalent, they were re-scaled to 500x500 pixels.

Lastly, with this being a multi-modal classifica-

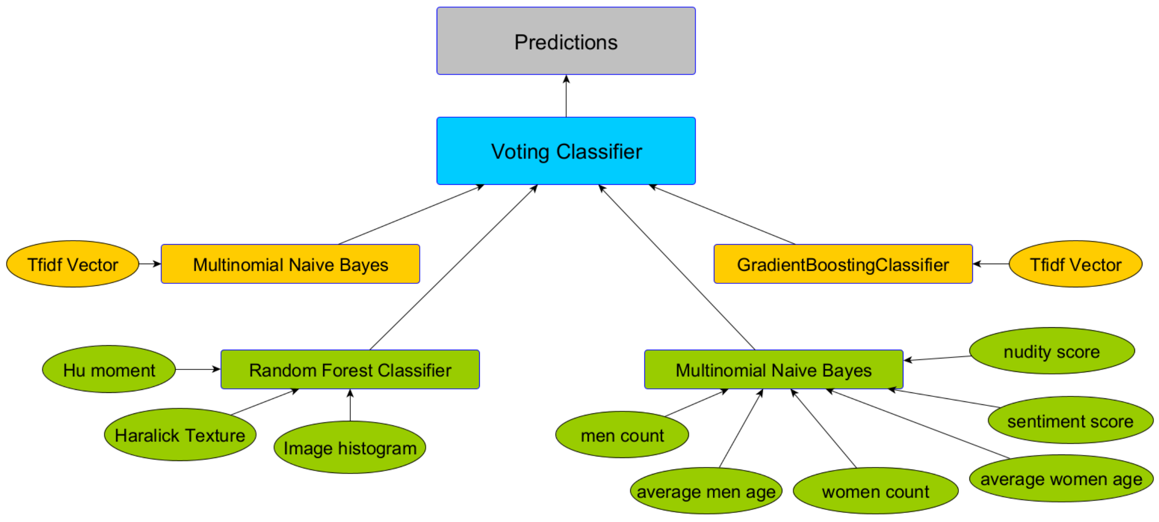


Figure 1: Ensemble Model used for Subtask A.

tion task, we were interested in whether we can extract additional information from the images/text transcriptions that might be relevant to decide if a meme is misogynistic or not. Thus, we enhanced the data set with the following features:

- The number of men/women depicted in the meme, as there might be a gender imbalance relevant for the classification process in combination with other features. Examining the text transcriptions, 'women' and 'woman' were the first and third most used word in misogynistic memes. It is not unreasonable to assume that women are also more likely to be featured in the corresponding images.
- For the same reason as above, we included the average estimated age of all men/women depicted in the meme.
- A binary nudity score, indicating whether an image is sexually explicit or not. In the training data, 15.75% of the misogynistic and only 2.86% of the non-misogynistic memes were sexually explicit.
- A sentiment score ranging from [-1, 1], indicating the polarity of any given text. A score of 1 indicates a positive statement, a score of -1 a negative sentiment. Misogynistic memes are usually hateful, so it can be assumed that the are more likel to use negative language in their texts.

All numerical scores retrieved from these features were transformed into a vector representation and used to train another multinomial naive Bayes classifier.

Finally, we built an ensemble model, using a majority voting rule, with all estimators created so far. Said model was hyperparameter tuned using a grid search, as well as cross-validated using five folds.

3.2 Experimental Setup

To extract information about the number of men/women and their estimated age, we made use of the Facial Recognition API provided by Face++.[2] The nudity score was obtained from the images using the NudeClassifier from NudeNet.[3] The classifier returns probabilities whether an image is sexually explicit or not. These probabilities were then transformed into a binary label. The sentiment was obtained using the TextBlob[4] library and the polarity scores added to the data set. The image features for the Random forest classifier were calculated through the opencv-python library,[5] whereas the tf-idf vectors, as well as the model itself, was built with the scikit-sklearn library (Pedregosa et al., 2011).[6]

The model was hypertuned using a grid search and a 5-fold cross-validation. The parameters for the grid search can be found in Table 2. The tf-idf vectors were tuned separately for the Multinomi-

[2]https://www.faceplusplus.com/
[3]https://github.com/notAI-tech/NudeNet
[4]https://github.com/sloria/textblob
[5]https://github.com/opencv/opencv-python
[6]https://scikit-learn.org/stable/index.html

nalNB classifier and the GradientBoostingClassifier. Similarly, the two MultinominalNB classifier were tuned separately, one in combination with the tf-idf vectors, and on in combination with the gender, age, nudity, and sentiment features.

Grid search parameter settings	
TfidfVectorizer	
ngram_range	(1,1), ***(1,2)***, (1,3), (2,2), (2,3)
analyzer	char, ***word***
max_features	***None***, 5000, 10000
MultinominalNB	
alpha	0.5, ***1.0***, 3.0
fit_prior	True, ***False***
RandomForestClassifier	
n_estimators	100, 1000, 5000, ***7000***
GradientBoostingClassifier	
n_estimators	100, 1000, 5000, ***7000***

Table 2: Grid search parameters for ensemble model. The final parameters are highlighted.

3.3 Results

Model	**Macro F1**
(1) MultinominalNB (tf-idf)	0.626
(2) GradientBoosting + (1)	0.645 *(+0.019)*
(3) MultinomialNB (gender) + (2)	0.649 *(+0.004)*
(4) RandomForest + (3)	**0.665** *(+0.016)*

Table 3: Gradual built-up of the ensemble model.

Our system for Subtask A achieved place 5 on the leaderboard, with a macro F1 score of 0.665. To make sure that each estimator of our ensemble model actually improves the classification of misogynistic memes, we gradually built it up and run it against the test data. As can be seen from the results in Table 3, both the GradientBoostingClassifier as well as the RandomForestClassifier provide substantial performance gains compared to the handcrafted gender, age, nudity, and sentiment features.

While the nudity score seems to offer strong support on whether a meme is misogynistic or not, the same cannot be said for the other handcrafted features. Given the sentiment score, for example, 1119 out of 5000 misogynistic memes received a negative sentiment score. Compared to 1103 non-misogynistic memes that also received a negative sentiment score, it seems likely that the score provides very little information for the classification process. Similarly, a direct correlation cannot be derived from the number of men/women or their estimated average age. Because of that, it is likely that they provide little information gain as well. The ensemble model from Subtask A was able to correctly classify instances of more explicit forms of misogyny. Memes that include the word "rape", mention "cooking" or "cleaning", show (exposed) female body parts in the image or explicitly mention them in the text are correctly identified as sexist. When only one of these features is present, the model produces false positives, for example harmless memes that feature women doing housework. The memes the model was not able to identify as misogynistic often do not feature human faces or only represent them by minimalistic drawings, are low quality, or have more subtle references to sexuality that the nudity detection cannot identify (such as handprints on breasts).

4 Subtask B: Multi-label classification of misogyny types

Although the ensemble model performed reasonably well for Subtask A, performance was significantly worse when using it for the extended misogynistic labels introduced in Subtask B. Because of this, we built and trained a different model infrastructure using a deep learning approach (which performed worse than our ensemble model, when applied to Subtask A).

4.1 System Overview

Subtask B is a multi-label classification task, so each meme can be assigned to one or more categories. To capture the features which differentiate those memes among different subtypes, we made use of a multi-label model from the Simple Transformers library,[7] using text data only.

In this task, we have five distinct binary labels for each data entry in the training set: misogyny, shaming, stereotype, objectification, and violence. The transformer-based model consists of a transformer model plus a classification layer on top of it. The main difference between this model and the binary classification model is that, in this model, the classification layer has five output neurons, corresponding to each out of the five labels in the training set. For the transformer model, we chose a pretrained RoBERTa model, which is imported by the Simple

[7] https://simpletransformers.ai/

	Misogyny (model A)		Shaming		Stereotype		Objectification		Violence	
	1 pred	0 pred	1 pred	0 pred	1 pred	0 pred	1 pred	0 pred	1 pred	0 pred
1 true	**333**	167	54	92	160	**190**	148	**200**	58	**95**
0 true	168	**332**	**98**	**756**	137	**513**	122	**530**	32	**815**

Table 4: Confusion matrices for all classification categories. Analysis was performed on the test set.

Transformer library from the Transformer library (Wolf et al., 2020) that was developed by HuggingFace. It is based on the RoBERTa model proposed by Liu et al. (2019).

RoBERTa removed the next sentence prediction task from BERT (Liu et al., 2019) which is one of the reasons why we chose RoBERTa over BERT, as the majority of the text transcriptions of memes in this task are consisting of either one or two sentences. The other reason being, that RoBERTa was trained on a larger dataset than BERT, thus more likely to result in better predictions.

4.2 Experimental Setup

The SimpleTransformers library is designed with the purpose of easily setting up a transformer model. Transforming text into a suitable vector representation is done automatically, and various hyperparameters can be tuned to improve performance. we set the threshold in our implementation to 0.8. The model was trained for 20 epochs using a GPU provided by the Kaggle notebook environment.[8]

4.3 Results

Our system achieved place 19 on the leaderboard, with a macro F1 score of 0.637 using the above-mentioned transformer-based model. We have also experimented with other approaches like the FastText library[9] which did not show better performance on this multi-label classification task than the transformer approach.

Compared to the true and false negatives in the ensemble model's prediction from Subtask A, which are very balanced, the model for Subtask B produces a lot more false negatives for the test set(see Table 4). True and false positives and negatives are very balanced in the ensemble model's predictions from Subtask A, as shown in Table 4. The model for Subtask B on the other hand produces more false negatives than positives for all categories except shaming. For this task, it would be favorable if the model was stricter and produced more false positives instead. The classifier performed reasonably well, even without information about the images. Still, some memes, especially in the stereotype and objectification category, cannot be understood to belong in that category without this information. Only about a third of the pictures (379 of 1000) are completely correctly identified when it comes to the four categorization labels, but 731 have at least 3 labels matching. Only 9 samples in the test set were classified in a way that no label matched the gold standard and 67 matched less than 2.

5 Conclusion

In our experiments, we explored and compared different models for multi-modal analysis of misogynistic memes. Surprisingly, ngram-models on both word and character levels did not perform as well as expected for this task. We discovered that ensemble models using text and image information can work well even if the text classifier uses simple features such as tf-idf vectors. BERT based text analysis performs better than the baselines, even if image features are not included. We found that an ensemble model can be improved through gradient boosting and adding information about nudity, age, gender of depicted humans, and text sentiment. In the future, it would be worth exploring how well a similar ensemble model with an (additional) BERT-classifier or other more powerful text classifier performs. For Subtask B, a RoBERTa-based multi-label classification model showed its power with purely text information. It would be interesting to train it with a class of weights and different threshold values. However, we were not able to create a well-performing multi-modal model that uses image information, which might be another interesting direction for further studies.

[8] https://www.kaggle.com/docs/notebooks
[9] https://fasttext.cc//

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on Twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Jakab Buda and Flóra Bolonyai. 2020. An ensemble model using n-grams and statistical features to identify fake news spreaders on Twitter Notebook for PAN at CLEF 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Simona Frenda, Bilal Ghanem, Manuel Montes-y Gómez, and Paolo Rosso. 2019. Online hate speech against women: Automatic identification of misogyny and sexism on Twitter. *Journal of Intelligent & Fuzzy Systems*, 36(5):4743–4752.

Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. 1973. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621.

Ming-Kuei Hu. 1962. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187.

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

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pranaydeep Singh, Nina Bauwelinck, and Els Lefever. 2020. LT3 at SemEval-2020 task 8: Multi-modal multi-task learning for memotion analysis. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1155–1162, Barcelona (online). International Committee for Computational Linguistics.

Shardul Suryawanshi and Bharathi Raja Chakravarthi. 2021. Findings of the shared task on troll meme classification in Tamil. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, pages 126–132, Kyiv. Association for Computational Linguistics.

Dang Van Thin, Lac Si Le, and Ngan Luu-Thuy Nguyen. 2019. NLP@UIT: Exploring feature engineer and ensemble model for hate speech detection at VLSP 2019. *Training*, 5:3–51.

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

UMUTeam at SemEval-2022 Task 5: Combining image and textual embeddings for multi-modal automatic misogyny identification

José Antonio García-Díaz
Facultad de Informática,
Universidad de Murcia,
Campus de Espinardo,
30100, Spain
joseantonio.garcia8@um.es

Camilo Caparros-Laiz
Facultad de Informática,
Universidad de Murcia,
Campus de Espinardo,
30100, Spain
camilo.caparrosl@um.es

Rafael Valencia-García
Facultad de Informática,
Universidad de Murcia,
Campus de Espinardo,
30100, Spain
valencia@um.es

Abstract

In this manuscript we describe the participation of the UMUTeam on the MAMI shared task proposed at SemEval 2022. This task is concerning the identification of misogynous content from a multi-modal perspective. Our participation is grounded on the combination of different feature sets within the same neural network. Specifically, we combine linguistic features with contextual transformers based on text (BERT) and images (BEiT). Besides, we also evaluate other ensemble learning strategies and the usage of non-contextual pretrained embeddings. Although our results are limited, we outperform all the baselines proposed, achieving position 36 in the binary classification task with a macro F1-score of 0.687, and position 28 in the multi-label task of misogynous categorisation, with an macro F1-score of 0.663.

1 Introduction

This manuscript describes the participation of the UMUTeam in the Multimedia Automatic Misogyny Identification (MAMI) shared task (Fersini et al., 2022), proposed at SemEval 2022. This shared-task consists in the identification and categorisation of misogynous content from a dataset composed of memes (Dawkins and Davis, 2017). A meme is essentially a pictorial content with an overlaying text that pretends to be funny. However, some of these memes are being used as a form of hate against women, with sexist messages in online social networks that amplify misogynous traits such as sexual stereotyping or gender inequality.

MAMI shared task proposes two challenges. A binary classification task, in which each meme should be labelled as misogynous or not misogynous, and a multi-label classification task, to categorise different misogynous traits, namely shaming, stereotype, objectification, and violence.

2 Background information

From the last years, the number of shared tasks in workshops regarding hate-speech and misogyny detection are increasing. To name just but a few, in Italian there is the EVALITA 2018 dataset (Bosco et al., 2018); in Spanish, the AMI 2018 dataset (Fersini et al., 2018) and the EXIST dataset (Rodríguez-Sánchez et al., 2021). In German, the GermEval 2021 (Risch et al., 2021) dataset.

The common approaches for misogyny detection and categorisation consists in the training of an automatic machine learning classifier. For example, the authors of (Anzovino et al., 2018) compiled and labelled a corpus from Twitter focused on misogynous content, and evaluate several feature sets and machine-learning models. In Spanish, a similar approach was conducted in (García-Díaz et al., 2021), in which the authors released the Spanish MisoCorpus 2020. This dataset is organised into three splits: (1) VARW (Violence Against Relevant Women), focused on aggressive messages on Twitter to women who have gained social relevance; (2) SELA (European Spanish vs that of Latin America), focused on distinguish between misogynistic messages from Spain and Latin America; and (3) DDSS (Discredit, Dominance, Sexual harassment and Stereotype), focused on general traits related to misogyny. The Spanish MisoCorpus 2020 is balanced and contains 3841 misogynous documents, annotated by three human annotators.

It is worth noting that our research group evaluated of a set of hand-crafted linguistic and negation features along with Spanish pre-trained contextual and non-contextual embeddings for detecting hate-speech (García-Díaz et al., 2022b). These work included two datasets concerning misogyny and sexist behaviour.

3 Dataset

Table 1 depicts the dataset proposed by MAMI. For the training and validation split, the labels were balanced, with 5000 misogynist memes and 5000 safe memes that were manually annotated using crowd sourcing platforms. Our first experiments consider a balance between both labels. However, during the final evaluation phase we suspect that there was a strong imbalance among the labels. As our first results were limited, we sub sampled the dataset removing some misogynous documents with less than 8 words. We are aware that there are better techniques for handling class imbalance but, due to time constraints, we could not evaluate them.

Split	**Original dataset**	**Subsampled**
training	8000	6709
val	2000	1677
test	1000	1000
total	11000	9386

Table 1: Dataset statistics of the MAMI dataset. We show the original distribution (left) and our sub sampled distribution (right)

Table 2 depicts the label distribution per misogynous trait for the second challenge. It should be noticed that shaming and violence traits are the traits with less instances, hinder their detection.

Split	**(OBJ)**	**(SHA)**	**(STE)**	**(VIO)**
training	1762	1020	2248	763
val	440	254	562	190
total	2202	1274	2810	953

Table 2: Misogynous trait distribution: Objectification (OBJ), Shaming (SHA), Stereotype (STE), and Violence (VIO)

4 Methodology

For solving the challenges proposed in MAMI, we build a system which architecture is depicted in Figure 1. It a nutshell, our system works as follows. First, we select some of the documents of the training MAMI dataset to create a custom validation dataset. Next, we extract a subset of language-independent linguistic features (LF), non-contextual sentence (SE) and word embeddings (WE) from fastText, the contextual word embeddings from BERT (BF), and the image embeddings from BEiT (BI). Second, we train several neural network models by performing an hyperparameter tuning process. The models evaluated included one model per feature set, and models based on several feature sets together. Besides, we evaluate two ensembles based on soft voting (mode) and averaging all the probabilities (mean) of the neural networks trained with each feature set. To handle the multi-label challenge, we repeat this process per trait. That is, we evaluate the problem as a binary classification problem per trait.

Next, some insights of the feature sets involved are given. As the memes are images with overlaying text, this shared-task has a multi-modal perspective. Our proposal uses several feature sets based on texts, and one for the images. First, we use the UMUTextStats tool to obtain a set of relevant psycho-linguistic features (LF). This tool has already been used in studies related to misogyny, such as (García-Díaz et al., 2021, 2022a). The LF included low-level linguistic categories concerning phonetics and syntax's, and high-level features related to semantics and pragmatics, including features proper from figurative language (del Pilar Salas-Zárate et al., 2020). Moreover, these kinds of features have proven to be effective for performing other automatic classification tasks such as irony and satire identification (García-Díaz and Valencia-García, 2022). As some of the dictionaries of UMUTextStats are not translated to English, we select a subset of language-independent linguistic features, based on linguistic metrics, Part-of-Speech features and the usage of social media jargon. Second, we extract sentence and word embeddings using the pre-trained fastText model (Joulin et al., 2016). Third, we use contextual sentence embeddings from BERT (Devlin et al., 2018), which sentence embeddings are obtained in a similar manner as described at S-BERT (Reimers and Gurevych, 2019). Forth, we use visual embeddings from BEiT (Bao et al., 2021), which is a self-supervised model trained with ImageNet-21k with more than 21000 labels. BEiT learns the embeddings images as a sequence of fixed-size elements using relative position. This allows us to perform the classification using a mean-pooling strategy from the final hidden states of the patches instead of placing a linear layer on top of the final classification token. However, the suggested way to fine-tune the model for performing downstream tasks is to attach a new linear layer that uses the last hidden state of the

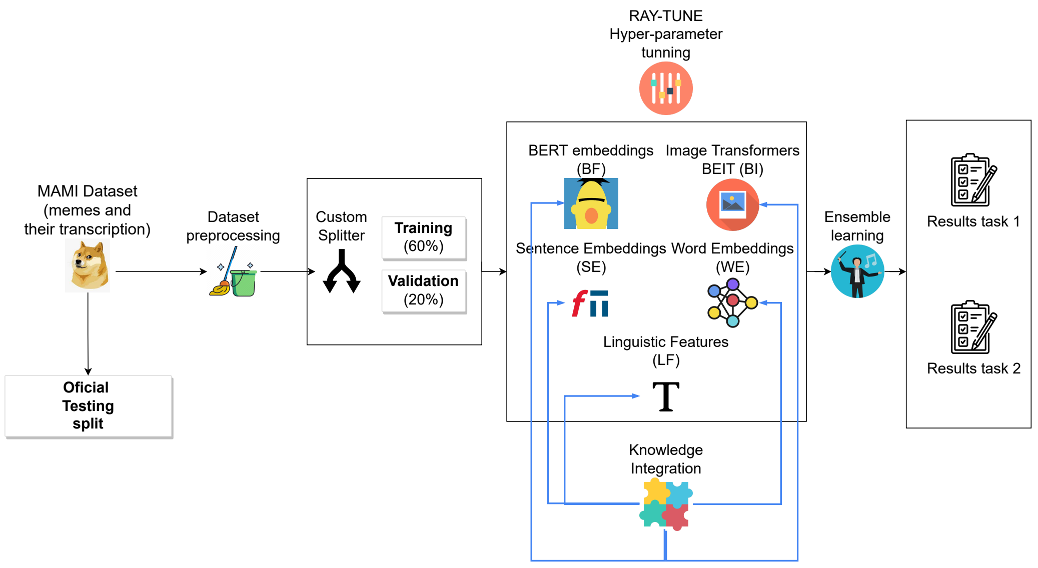


Figure 1: System architecture proposed by the UMUTeam for solving the MAMI shared task

classification token as a representation of the whole image.

To obtain the embeddings from BERT (for text) and BeIT (for images) we conduct an hyperparameter optimisation stage using RayTune (Bergstra et al., 2013) with a Tree of Parzen Estimators (TPE) to select the best combination of the hyperparameters over 10 trials. The hyperparameters evaluated and their interval range are: (1) weight decay (between 0 and .3), (2) training batch size ([8, 16]), (3) warm-up steps ([0, 250, 500, 1000]), (4) number of training epochs ([1-5]), and (5) learning rate (between 1e–5 and 5e–5).

Once all features are obtained, we train a neural network per feature set. The training of each neural network is performed with hyperparameter optimisation. Each training involved: (1) 20 shallow neural networks, that are multi-layer perceptrons (MLP) composed by one or two hidden layers with the same number of neurons per layer connected with one activation function (`linear`, `ReLU`, `sigmoid`, and `tanh`); (2) 5 deep-learning networks, that are MLP between 3 and 8 hidden layers, in which the neurons per layer are disposed for each layer in different shapes, namely brick, triangle, diamond, rhombus, and funnel, and connected with an activation function (`sigmoid`, `tanh`, `SELU` and `ELU`). The learning rate of the deep-learning models is `10e-03` or `10e-04`. Besides, on the neural networks with the pre-trained word embeddings from fastText (WE) we also evaluate 10 convolutional neural networks (CNN) and 10 bidirectional recurrent neural network layers (BiLSTM). In all experiments, we evaluate two batch sizes: `16` and `32`. These small values were selected because the training split was balanced, and a dropout mechanism (`[False, .1, .2, .3]`) for regularisation.

Apart of the neural networks trained with the feature sets separately, we evaluate different forms for combining the strengths of each feature set in the same system. The combination of the feature sets is performed using two strategies: (1) knowledge integration, in which each feature set is used as input of the same neural network. For this, we train another neural network repeating the hyperparameter optimisation stage; and (2) ensemble learning, in which the output of each neural network model trained with a feature set is combined by averaging the predictions or calculating the mode of the predictions.

5 Results and discussion

Each system was evaluated using the custom validation split (see Section 3).

The results for the first challenge are depicted in Table 3. Note that we remove some of the documents with fewer words to sub sample the dataset. For that reason, our validation split contains 677 misogynous and 1000 non-misogynous documents.

As it can be observed in Table 3, the performance of LF in isolation is limited. This fact is not surprising because not all of the features from UMUTextStats are available in English. For the rest of the textual embeddings (SE, WE, and BF), BF obtains the best results, with a macro F1-score of

Feature set	MIS	N-MIS	F1
LF	58.956	75.771	67.363
SE	73.286	80.860	77.073
WE	76.614	83.367	79.991
BF	79.690	87.306	83.498
BI	65.744	81.984	73.864
LF,BF,BI	**80.277**	**87.549**	**83.913**
LF,BF,BI (mode)	75.874	86.524	81.199
LF,BF,BI (mean)	76.853	85.183	81.018

Table 3: Results for the Task A with the custom validation split, reporting the F1-score of the misogynous (MIS) and non-documents (N-MIS) and the macro F1-score (F1).

83.498, outperforming the non-contextual sentence and word embeddings from fastText (SE and WE). Due of this, we decided to discard non-contextual embeddings and use BERT for the knowledge integration strategy. The combination of LF, BF, and BI in the same neural network outperformed the results achieved by BF, increasing slightly the F1-score of both labels and the macro f1-score. However, the ensemble learning strategy achieved lower results for the F1-score of the misogynous label, regardless of the strategy employed for combining the predictions.

For the second challenge, the results with the custom validation split are depicted in Table 4. We report the F1-score of each binary model that we train for each misogynous trait. Similar to the first challenge, the results achieved by LF are limited, achieving results below 60% in all the traits. Concerning non-contextual embeddings, the results with WE are superior to SE for all traits, but inferior compared to BF. In case of BI, it draws our attention the high macro F1-score achieved in objectification and shaming, outperforming BF. However, their results are inferior to BF for stereotype and violence. When LF is combined with BF and BI embeddings within the same neural network, the results are superior to the ones achieved separately, except in shaming. However, the results achieved with the ensemble learning strategy are quite limited, specially with the mean strategy. The result obtained with the violence trait is especially striking. We observe that the resulting model achieved a perfect recall over the violence class, which suggests that this model is always predicting all tweets as violence.

Table 5 depicts the official results for the first challenge. We achieve position 36/83 with a macro F1-score of 68.7. As it can be observed, our submission outperforms all the proposed baselines that consisted in: (1) sentence embeddings from the USE pre-trained model; (2) image features from VGG-16; (3) a combination of deep image and text representations based a shallow neural network with a single layer. In addition, two baselines focused on the second challenge were also evaluated: (4) a multi-label model, based on the concatenation of deep image and text representations, for predicting simultaneously if a meme is misogynous and the corresponding type; and (5) a hierarchical multi-label model, based on text representations, for predicting if a meme is misogynous or not and, if misogynous, the corresponding type.

The best result for the first challenge is achieved by the *SRCB_roc* team, with an F1-score of 83.4. It is worth mentioning that, although there was a restriction of the number of accounts available in Codalab per team and user, the organisers of the task are not able to control it. Nevertheless, in the official leader board the second best result was also achieved by the team *SRCB_roc*, with an F1-score of 81.1. However, as the team name is the same, we have removed this result from the table. Therefore, for the official results we ask to the reader to check the official results published in the overview of the task.

For this first challenge we send different runs and modify our strategy according to the results. We also send some basic results to obtain some baselines. For example, we achieved an macro F1-score of 52.85 with LF. This result is more limited than the ones achieved in the validation split. With non contextual embeddings, SE and WE, the results are, respectively, 61.30 and 61.96 (vs 77.073 and 79.99 with the validation split), and 64.75 for BF. Because of these results, we suspect that there are relevant different between the training and testing splits. Then, we examined carefully the testing split but we could not find relevant differences so we suspect to imbalanced as an possible explanation of this problem. To confirm this, we send a toy submission with a baseline consisting in all the predictions as non-misogyny and we observed a ratio similar to 1:3 between misogynous and non-misogynous instances. Then, we reduced the training dataset and retrained all models in order to make them stronger against class imbalance (see Section 3). It is worth noting that our methods

Feature set	Objectification	Shaming	Stereotype	Violence
LF	59.449	57.858	57.733	59.052
SE	66.983	66.479	70.923	66.730
WE	69.727	68.418	72.291	69.390
BF	72.849	68.452	73.785	67.815
BI	75.112	69.834	59.529	63.551
LF-BF-BI	**76.911**	68.957	**73.855**	**69.457**
LF,BF,BI (mode)	65.997	**70.046**	64.471	59.740
LF,BF,BI (mean)	45.465	66.330	45.577	8.680

Table 4: Results for the Task B with our custom validation split including the Macro F1-score for each misogynous trait

Rank	Team	F1-score
1	SRCB_roc	83.4
2	DD-TIG	79.4
3	Beantown	77.8
36	UMUTeam	68.7
58	Baseline 1	65.0
61	Baseline 2	64.0
61	Baseline 3	63.9
79	Baseline 4	54.3
83	Baseline 5	43.7

Table 5: Official results for the Task A

Rank	Team	F1-score
1	SRCB_roc	73.1
2	TIBVA	73.1
3	PAFC	73.1
28	UMUTeam	66.3
41	Baseline 1	62.1
48	Baseline 2	42.1

Table 6: Official results for the Task B

already consider some good practises concerning class imbalance, such as setting the initial bias, adding class weights to the model, heavily weight the few examples that are available

Finally, the official results for the second challange are depicted in Table 6. It can be observed that our best result achieved a 66.3 of F1-score, outperforming the two baselines proposed: (1) a hierarchical multi-label model (baseline 1), based on text representations, and (2) a multi-label model (baseline 2), based on the concatenation of deep image and text representations, for predicting the corresponding misogynous type.

The best result was a F1-score of 73.1, with a triple tier between teams *SRCB_roc*, *TIBVA*, and *PAFC*. Our best proposal, however, achieved position 28 in the official leader board, outperforming all baselines.

6 Conclusions

In this paper the participation of the UMUTeam in the MAMI shared task, concerning the identification and categorisation of misogynous content in memes, is described. Our approach for solving the binary and multi-label classification tasks consisted in the combination of a set of language-independent linguistic features with contextual images and textual features obtained from the documents. Our best result was achieved in the misogynous categorisation task, with an macro F1-score of 66.3, reaching position 28 in the ranking.

We consider that the weakest point of our proposal is that we have not handle class imbalance in the testing dataset. However, we have evaluated some strategies that have improve our results, as reducing the number of instances and the application of class weight. As further work, we will evaluate data augmentation techniques, both for images and for text in order to deal class imbalance.

Acknowledgements

This work is part of the research project LaTe4PSP (PID2019-107652RB-I00) funded by MCIN/ AEI/10.13039/501100011033. This work is also part of the research project PDC2021-121112-I00 funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. In addition, José Antonio García-Díaz is supported by Banco Santander and the University of Murcia through the Doctorado Industrial programme.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Hangbo Bao, Li Dong, and Furu Wei. 2021. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*.

James Bergstra, Daniel Yamins, and David Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning*, pages 115–123. PMLR.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, volume 2263, pages 1–9. CEUR.

Richard Dawkins and Nicola Davis. 2017. *The selfish gene*. Macat Library.

María del Pilar Salas-Zárate, Giner Alor-Hernández, José Luis Sánchez-Cervantes, Mario Andrés Paredes-Valverde, Jorge Luis García-Alcaraz, and Rafael Valencia-García. 2020. Review of english literature on figurative language applied to social networks. *Knowledge and Information Systems*, 62(6):2105–2137.

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

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018. Overview of the task on automatic misogyny identification at ibereval 2018. *Ibereval@ sepln*, 2150:214–228.

José Antonio García-Díaz, Mar Cánovas-García, Ricardo Colomo-Palacios, and Rafael Valencia-García. 2021. Detecting misogyny in spanish tweets. an approach based on linguistics features and word embeddings. *Future Generation Computer Systems*, 114:506–518.

José Antonio García-Díaz, Ricardo Colomo-Palacios, and Rafael Valencia-García. 2022a. Psychographic traits identification based on political ideology: An author analysis study on spanish politicians' tweets posted in 2020. *Future Generation Computer Systems*, 130:59–74.

José Antonio García-Díaz, Salud María Jiménez-Zafra, Miguel Ángel García-Cumbreras, and Rafael Valencia-García. 2022b. Evaluating feature combination strategies for hate-speech detection in spanish using linguistic features and transformers. *Complex & Intelligent Systems*.

José Antonio García-Díaz and Rafael Valencia-García. 2022. Compilation and evaluation of the spanish saticorpus 2021 for satire identification using linguistic features and transformers. *Complex & Intelligent Systems*, pages 1–14.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Julian Risch, Anke Stoll, Lena Wilms, and Michael Wiegand. 2021. Overview of the germeval 2021 shared task on the identification of toxic, engaging, and fact-claiming comments. In *Proceedings of the GermEval 2021 Shared Task on the Identification of Toxic, Engaging, and Fact-Claiming Comments*, pages 1–12.

Francisco Rodríguez-Sánchez, Jorge Carrillo-de Albornoz, Laura Plaza, Julio Gonzalo, Paolo Rosso, Miriam Comet, and Trinidad Donoso. 2021. Overview of exist 2021: sexism identification in social networks. *Procesamiento del Lenguaje Natural*, 67:195–207.

YNU-HPCC at SemEval-2022 Task 5: Multi-Modal and Multi-label Emotion Classification Based on LXMERT

Chao Han, Jin Wang and Xuejie Zhang
School of Information Science and Engineering
Yunnan University
Kunming, China
`hc_super@mail.ynu.edu.cn, {wangjin, xjzhang}@ynu.edu.cn`

Abstract

This paper describes our system used in the SemEval-2022 Task5 Multimedia Automatic Misogyny Identification (MAMI). This task is to use the provided text-image pairs to classify emotions. In this paper, We propose a multi-label emotion classification model based on pre-trained LXMERT. We use Faster-RCNN to extract visual representation and utilize LXMERT's cross-attention for multi-modal alignment. Then we use the Bilinear-interaction layer to fuse these features. Our experimental results surpass the F_1 score of baseline. For Sub-task A, our F_1 score is 0.662 and Sub-task B's F_1 score is 0.633. The code of this study is available on GitHub[1].

1 Introduction

In social networks, meme is mainly used to express the emotion of netizen. It usually consists of text and images. But at the same time, memes also convey some negative emotions, such as negative comments about women. SemEval-2022 Task5: Multimedia Automatic Misogyny Identification (MAMI) (Fersini et al., 2022) focuses on identifying whether meme conveys negative emotions towards women.

- Sub-task A: a basic task about misogynous meme identification, where a meme should be categorized either as misogynous or not misogynous;
- Sub-task B: an advanced task, where the type of misogyny should be recognized among potential overlapping categories such as stereotype, shaming, objectification, and violence.

Since the Transformer (Vaswani et al., 2017) and BERT (Devlin et al., 2019) models were proposed, researchers have begun to work on image and text multi-modality work in recent years, in addition to using one modality such as only image or text. Nowadays, for multimodal models, they can be divided into two categories, single-stream model and dual-stream model. In the single-stream model, language information and vision information are fused at the beginning and directly input into the encoder. Some representative single-stream models include ImageBERT (Qi et al., 2020), Unicoder VL (Li et al., 2020), VL-BERT (Su et al., 2020), VisualBERT (Li et al., 2019), etc. In the dual-stream model, in addition to the LXMERT, we will introduce below, there were ViLBert (Lu et al., 2019) and UNIMO (Li et al., 2021), etc.

As for emotion recognition, in previous tasks, there are also emotion classification tasks based on multi-modal graphics and text, such as Zhu et al. (2021) used text-CNN and ALBERT to Identify the persuasion skills of Meme. Peng et al. (2020) used the adversarial learning of sentiment word representations for sentiment analysis. A tree-structured regional CNN-LSTM (Wang et al., 2020) and dynamic routing in a tree-structured LSTM (Wang et al., 2019) were used for dimensional sentiment analysis. In previous SemEval competitions, Tian et al. (2021) extracted heterogeneous visual representations (i.e., face features, OCR features, and multimodal representations) and explored various multimodal fusion strategies to combine the textual and visual representations. In addition, in multimodal analysis combining images and text, Yuan et al. (2020) proposed a parallel channel ensemble model combining BERT embedding, BiLSTM, attention and CNN, and ResNet for sentiment analysis of memes.

The main difficulty of multi-modality is how to extract the two modalities' features and express the semantics more accurately, which involves the representation of multi-modality, the alignment between multi-modality, and the fusion of multi-

[1] `https://github.com/HC-super/SemEval-2022-Task-5`

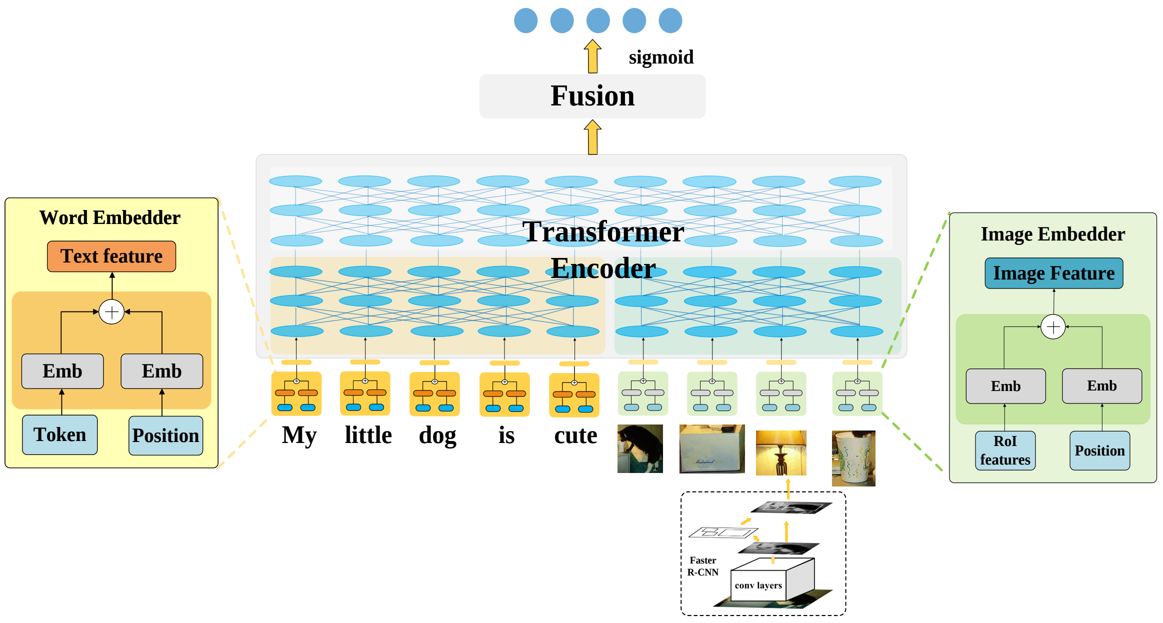


Figure 1: Overview of the proposed model specifically shows the general structure of Faster R-CNN and the details of the embedders.

modality. For the multi-modal task of text and image, the previous practice is to input the text and image into two different pre-training models for processing the text and image modalities respectively, and then concatenate the output features and predict the emotion. However, this method lacks the processing of the alignment relationship between modalities. The proposed model considers the above three problems in the multi-modality field. Inspired by LXMERT, we use it as the main framework of our model. We use Faster R-CNN (Ren et al., 2017) to extract image RoI features and their position. For texts, we use BERT to extract text embedding. Then our system uses LXMERT (Tan and Bansal, 2019) to deal with the multi-modal alignment of text and image. After when two modalities are processed by LXMERT, we use the learnable integration mechanism Bilinear-interaction layer to fuse these features.

The remainder of this paper is organized as follows. In section 2, we described LXMERT and our fusion method in detail. The experimental results are presented in section 3. Finally, a conclusion is drawn in section 4.

2 System Overview

Task A and Task B are very similar in model structure except for the output layer. Therefore, we introduce the model we proposed as a whole. This model can be divided into four parts. They are the embedding layer for image and text preprocessing, the encoder for multi-modal presentation and alignment, the feature fusion layer, and the final output layer. The proposed model is as shown in Figure 2.

2.1 Embedding

For images, LXMERT does not simply use a convolutional neural network to output feature map but uses (Anderson et al., 2018) to extract objects from images. The image processing of LXMERT is similar to text processing inspired by BERT. The specific idea is to use Faster R-CNN to select 36 RoI (region of interest) boxes with high confidence for each image and use these boxes as the features of the image. Similar to the text processing of BERT, the model also considers the position of each box and embeds the corresponding position. 36 objects are extracted by Faster R-CNN as $\{o_1, \ldots, o_{36}\}$. f_j is the 2048 dimension RoI features of o_j, and p_j is its position. As is shown in figure 2, the processing of these variables is as follows:

$$\begin{aligned} \hat{f}_j &= \text{LayerNorm}\left(W_{\text{F}} f_j + b_{\text{F}}\right) \\ \hat{p}_j &= \text{LayerNorm}\left(W_{\text{P}} p_j + b_{\text{P}}\right) \\ v_j &= \left(\hat{f}_j + \hat{p}_j\right)/2 \end{aligned} \tag{1}$$

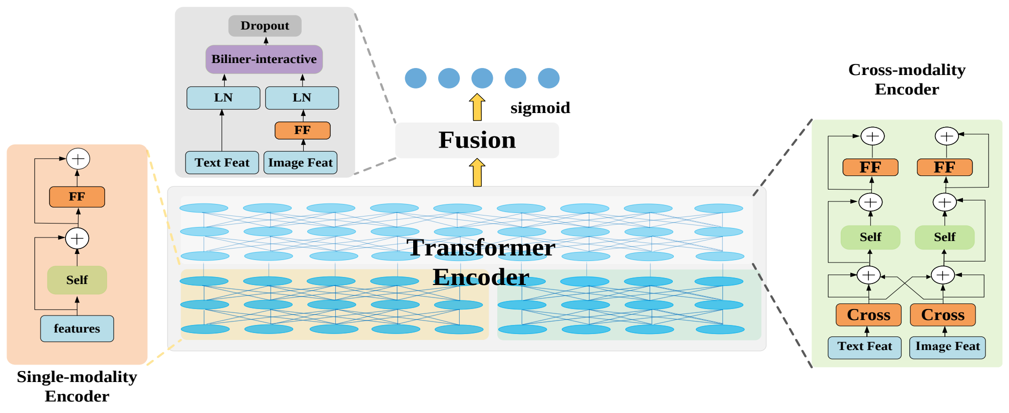


Figure 2: Overview of the proposed model, which specifically shows the details of the encoders and fusion layer. 'Self' and 'Cross' represent self-attention sub-layers and cross-attention sub-layers, respectively. 'FF' denotes a feed-forward sub-layer.

where W_F and W_P are the trainable weights of fully connected layer in matrix format. Moreover, b_F and b_P are the bias of the layer. $\hat{f}_j$ and $\hat{p}_j$ are the output of the layer-normalization.

For the text, sentences are converted into tokens whose length is equal to the length of scent according to the practice of WordPiece tokenizer (Wu et al., 2016). For instance, when the length of the sentence is n, the word tokens are $\{w_1, ..., w_n\}$. Then word w_i and its index i (the absolute position of w_i) are projected to vectors by embedding sub-layers.The specific structure of embedder is shown in Figure 1. Then added to the index-aware word embedding:

$$\begin{aligned} \hat{w}_i &= \text{WordEmbed } (w_i) \\ \hat{u}_i &= \text{IdxEmbed } (i) \\ h_i &= \text{LayerNorm } (\hat{w}_i + \hat{u}_i) \end{aligned} \tag{2}$$

The specific structure of embedder is shown in Figure 1.

2.2 Attention layer

In this subsection, we will give a brief description of the attention mechanism. The principle of the attention mechanism is to give a request vector x and its context vector y_j, then, calculate the correlation between x and each y_j, and get a correlation score. The correlation score used in LXMERT is the dot product of vector x and vector y_j. After calculating the scores of all relevant context vectors y_j for x, LXMERT uses softmax to convert each score into a probability α_j to obtain the attention distribution.

$$\begin{aligned} a_j &= \text{score}\,(x, y_j) \\ \alpha_j &= \exp\,(a_j) \,/ \sum_k \exp\,(a_k) \end{aligned} \tag{3}$$

$$\text{Att}_{\text{X}\rightarrow\text{Y}}\,(x, \{y_j\}) = \sum_j \alpha_j y_j \tag{4}$$

The output of the layer is the weighted sum of all probabilities with y_i.

The self-attention layer in LXMERT is implemented in a similar way to the attention layer, except that the query vector x in self-attention comes from the context-dependent vector y_i.

2.3 Encoder

The processing of image modality and text modality is shown in Figure 2. After embedding two modalities, LXMERT uses the two transformer single-modality encoders. One is a text encoder and another is an image encoder. Each layer in a single-modality encoder contains a self-attention ('Self') sub-layer and a feed-forward ('FF') sub-layer, where the feed-forward sub-layer is further composed of two fully-connected sub-layers. We take N_L and N_R layers in the language encoder and the object-relationship encoder, respectively. We add a residual connection and layer normalization (annotated by the '+' sign in Figure 2) after each sub-layer as in Transformer (Vaswani et al., 2017). The features processed by a single-modality encoder will be first sent to another encoder called the cross-modality layer. Its main function is to align the features of the two modalities. The bi-directional cross-attention sublayer

Emotion category	Number of label '1'	Overall proportion
misogynous	5000	50.00%
shaming	1274	12.74%
stereotype	2810	28.10%
objectification	2202	22.02%
violence	953	9.53%

Table 1: Training dataset label analysis.

contains two unidirectional cross-attention sub-layers, one from image to text and the other from text to the image. LXMERT stacks them N_x times, the input of k-th layer is the output of the previous$(k-1)$-th layer. Similarly, the query and context vectors are the outputs of the $(k-1)$-th layer. The method of processing the text features h_i^{k-1} and the image features v_j^{k-1} in unidirectional cross-attention sub-layers is as follows:

$$\begin{aligned}\hat{h}_i^k &= \text{Cross } Att_{\text{L}\rightarrow\text{R}}\left(h_i^{k-1}, \left\{v_1^{k-1}, \ldots, v_m^{k-1}\right\}\right)\\ \hat{v}_j^k &= \text{Cross } Att_{\text{R}\rightarrow\text{L}}\left(v_j^{k-1}, \left\{h_1^{k-1}, \ldots, h_n^{k-1}\right\}\right)\end{aligned} \quad (5)$$

where $\hat{h}_i^k$ and $\hat{v}_j^k$ are the output of the cross-attenton layer.

Then LXMERT further inputs the features processed by the cross-modality sublayer to the self-attention sublayer. This method aimed to further construct the internal connection of each modality after alignment. The specific treatment is:

$$\begin{aligned}\tilde{h}_i^k &= \text{Self } Att_{\text{L}\rightarrow\text{L}}\left(\hat{h}_i^k, \left\{\hat{h}_1^k, \ldots, \hat{h}_n^k\right\}\right)\\ \tilde{v}_j^k &= \text{Self } Att_{\text{R}\rightarrow\text{R}}\left(\hat{v}_j^k, \left\{\hat{v}_1^k, \ldots, \hat{v}_m^k\right\}\right)\end{aligned} \quad (6)$$

$\hat{h}_i^k$, $\hat{v}_j^k$ then processed by self-attention to $\tilde{h}_i^k$ and $\tilde{v}_j^k$, which will be further input to an 'FF' sublayer, connected through a residual, and input to the normalization to obtain the final output h_i^k, v_j^k. For each text in the data, the model will generate a Pooler output. We use the Pooler of each sentence as the output of the text modality.

2.4 Fusion

The method of this layer is inspired by Sina's paper FiBiNET by Huang et al. (2019). After LXMERT outputs two modality features, we need to further process its output. The dimension of image features is 36×768, while the dimension of text features is 768. To better integrate the two modalities, we flatten the image features and change its dimension to 768 through a feed-forward layer. Then, each modality will be normalized through layer normalization. Then, the features of each modality are sent to the Bilinear-interactive layer.

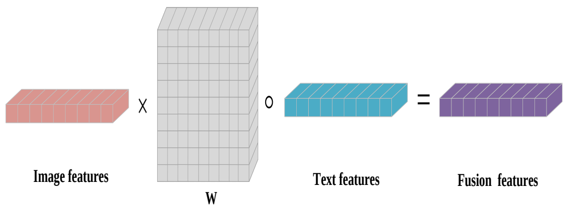


Figure 3: Bilinear-interactive layer.

The idea of the Bilinear-interactive layer is as shown in Figure 3. We establish a k-order square matrix W, which is trainable. To fuse the information of various modalities, 768-dimensional image features will first inner product with W. Then, for text features, we use Hadamard product to multiply the previous matrix. We finally use the dropout layer to improve the generalization ability of the model.

2.5 Output layer

- Sub-task A: this task is a binary classification task, so in the output layer, we use a shape of 768×1 full connection layer and use sigmoid as the activation function to process the results.

- Sub-task B: this task is a multi-label classification task. Therefore, in the output layer, we use a full connection layer whose shape is 768×5. Since each label classification is equivalent to binary classification, we use sigmoid as the activation function to process the results during output.

3 Experiments and Evaluation

3.1 Dataset

The task organizer provided 10000 pieces of data for training, including meme images with image serial numbers and text descriptions corresponding to the image. In the training dataset, there are 10000 images and an excel table to record the text corresponding to the images and supervise the learning of the corresponding labels.

When analyzing the data, we found that different labels account for different proportions in the number of their respective classifications. For the misogynous tag, both 0 and 1 categories account for 50%, so the data sample tag is more balanced for a supervision task. However, for the other four labels such as sharing, the proportion of label 1 is only 12.74%. Among 10000 samples, the label of violence accounts for only 9.53%. Table 1 shows the proportion of each label in the training dataset. As shown in Table 1, we find that the proportion of labels of different categories is very different, and there is data imbalance. This will make the model have a strong learning effect on a large classification label and easy to classify. However, for the low proportion of classification tags, it is difficult to learn and classify.

Based on this, we use Focal loss by (He et al., 2016) as the loss function of our model.

3.2 Experimental configuration

Our model is based on TensorFlow platform version 2.5.0. The main model adopts LXMERT from the Hugging Face transformers toolkit. We first use*UNC-NLP/LXMERT-base-uncased* tokenizer-Fast to process our text to embeddings, and we also use *UNC-NLP/LXMERT-base-uncased* pre-trained model as our base model LXMERT's pre-trained model. The Adam optimizer (Kingma and Ba, 2015) was used to update all trainable parameters. The Hyper-parameters configuration used in the model is shown in Table 2: We use Faster R-CNN to extract features of images, which is based on the paper by (Anderson et al., 2018). In this task, we use an open-source docker image *airsplay / bottom-up attention* and use a Faster R-CNN pre-training model based on ResNet101 to extract 36 RoI feature boxes and their corresponding position.

3.3 Evaluation Metrics

Sub-task A Systems will be evaluated using macro-average F1-score. In particular, for each class label (i.e. misogynous and not misogynous) the corresponding F1-score will be computed, and the final score will be estimated as the arithmetic mean of the two F1-score. Sub-task B Systems will be evaluated using weighted-average F1-score. In particular, the F1-score will be computed for each label and then their average will be weighted by support, i.e, the number of true instances for each label.

Adam Optimizer config	**Value**
Learning rate	5e-5
epsilon	1e-8
Focal loss config	**Value**
alpha	0.25
gamma	3
batch size	16
epoch	20

Table 2: Hyper-parameters config.

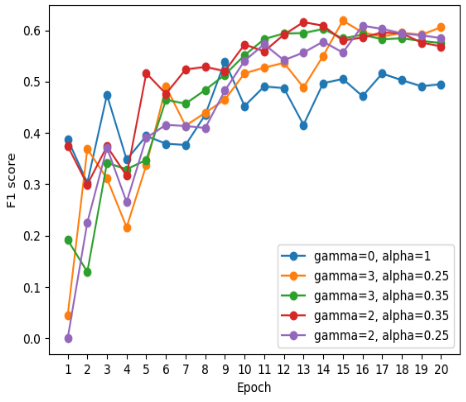


Figure 4: The ablation experiment of the Focal loss for different hyperparameter

$$\begin{aligned} \text{precision} &= \frac{TP}{TP+FP} \\ \text{recall} &= \frac{TP}{TP+FN} \end{aligned} \tag{7}$$

TP is the number of true positives classified by the model. FN is the number of false negatives classified by the model. FP is the number of false positives classified by the model.

$$\text{F}_1\text{-score} = \frac{2 \times \text{ precision } \times \text{ recall}}{\text{precision } + \text{ recall}} \tag{8}$$

F_1-score is the harmonic average of recall and precision.

3.4 Hyperparametric selection

In this section, we mainly introduce the hyperparametric selection of focal loss of the model. We adjust the two hyperparameters γ and α of Focal loss and train the model. In the training dataset, we randomly take 90 % data for the training model and the remaining 10 % as the test dataset to test

Model	Task A F1-score	Task B F1-score
Only image(Base line)	0.639	N/A
Only text(Base line)	0.640	N/A
Only ELECTRA	N/A	0.5454
Only ResNet	N/A	0.4581
ELECRTA and ResNet with concatenate	N/A	0.4816
ELECRTA and and ResNet with Fusion	N/A	0.5041
Image and Text(Base line)	0.650	0.621
LXMERT without Fusion	0.655	0.629
LXMERT with Fusion	**0.662**	**0.633**

Table 3: Performance comparison of different models. As shown in the table, our proposed model achieves the best results.

the performance of the model. The loss function focal loss is modified based on the standard cross-entropy loss.

This function can reduce the easy-to-classify samples so that the model can more focus on the samples that are difficult to classify in training. p_t in cross-entropy loss function reflects the recognition ability of the model to this sample (i.e. how well the knowledge is mastered). We define p_t is:

$$p_{\mathrm{t}} = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases} \tag{9}$$

The smaller the p_t is, the more difficult it is to classify, so contribution should be improved to the loss function when calculating the loss. Therefore, the specific method of Focal loss is to multiply a weight with p_t before the entropy loss function. α is balancing factor,$\alpha \in [0,1]$, γ is modulating factor, $\gamma \in [0,5]$. The Focal loss is as:

$$Focal\_loss(p_t) = -\alpha(1-p_t)^{\gamma} log(pt) \tag{10}$$

Thus, when $\alpha = 1$, $gamma = 0$, focal loss is similar to the cross-entropy loss function. By changing the values of γ and α, we found that when $\alpha = 0.25$ and $\gamma = 3$, for sub-task B, the weighted F_1 score of our model reached 0.662 and 0.633. See Figure 4.

3.5 Model comparison

We compare our model to a baseline and a model that combines two pre-trained models based on ELECTRA (Clark et al., 2020) and ResNet-101 (Ren et al., 2017) in this section. ELECTRA deals with text modality and ResNet is used to deal with image modality. The methods of feature fusion are compared with the Bilinear-interactive layer and concatenate layer using direct concatenate. The specific task is based on sub-task B. See Table 3 for details.

4 Conclusion

In this task, we design an image and text multi-modality model based on LXMERT for multi-modality representation and alignment, and modality fusion based on the Bilinear-interaction layer. Compared with the traditional method of stitching two pre-training models for each modality then concatenating two features to predict emotion, this model considers the representation, alignment, and fusion of multi-modality, and achieves better results than the baseline method.

At the same time, we found that after adding the Bilinear-interaction layer, the performance of the model is better than using only feature concatenate. See Table 3. Meanwhile, when analyzing the data, we found that the background of the meme graph and some characters in the graph were not used as the target input model by Faster R-CNN, which may affect the accuracy of the model. Meanwhile, the size of the meme image is too small to include multiple targets, and the target is relatively single, which may affect the performance of the model.

5 Acknowledgement

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61966038. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6077–6086. Computer Vision Foundation / IEEE Computer Society.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. Fibinet: combining feature importance and bilinear feature interaction for click-through rate prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, pages 169–177. ACM.

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

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 11336–11344. AAAI Press.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557.

Wei Li, Can Gao, Guocheng Niu, Xinyan Xiao, Hao Liu, Jiachen Liu, Hua Wu, and Haifeng Wang. 2021. UNIMO: towards unified-modal understanding and generation via cross-modal contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2592–2607. Association for Computational Linguistics.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Bo Peng, Jin Wang, and Xuejie Zhang. 2020. Adversarial learning of sentiment word representations for sentiment analysis. *Information Sciences*, 541:426–441.

Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti, and Arun Sacheti. 2020. Imagebert: Cross-modal pre-training with large-scale weak-supervised image-text data. *CoRR*, abs/2001.07966.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: pre-training of generic visual-linguistic representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hao Tan and Mohit Bansal. 2019. LXMERT: learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5099–5110. Association for Computational Linguistics.

Junfeng Tian, Min Gui, Chenliang Li, Ming Yan, and Wenming Xiao. 2021. Mind at semeval-2021 task 6: Propaganda detection using transfer learning and multimodal fusion. pages 1082–1087. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Jin Wang, Liang-Chih Yu, K. Robert Lai, and Xuejie Zhang. 2019. Investigating dynamic routing in tree-structured lstm for sentiment analysis. pages 3430–3435. Association for Computational Linguistics.

Jin Wang, Liang Chih Yu, K. Robert Lai, and Xuejie Zhang. 2020. Tree-structured regional cnn-lstm model for dimensional sentiment analysis. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 28.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Li Yuan, Jin Wang, and Xuejie Zhang. 2020. Ynu-hpcc at semeval-2020 task 8: Using a parallel-channel model for memotion analysis. pages 916–921. International Committee for Computational Linguistics.

Xingyu Zhu, Jin Wang, and Xuejie Zhang. 2021. YNU-HPCC at semeval-2021 task 6: Combining ALBERT and text-cnn for persuasion detection in texts and images. In *Proceedings of the 15th International Workshop on Semantic Evaluation, SemEval@ACL/IJCNLP 2021, Virtual Event / Bangkok, Thailand, August 5-6, 2021*, pages 1045–1050. Association for Computational Linguistics.

TIB-VA at SemEval-2022 Task 5: A Multimodal Architecture for the Detection and Classification of Misogynous Memes

Sherzod Hakimov[1,2], **Gullal S. Cheema**[1], and **Ralph Ewerth**[1,2]
[1]TIB – Leibniz Information Centre for Science and Technology
[2]Leibniz University Hannover, L3S Research Center
Hannover, Germany
{sherzod.hakimov, gullal.cheema, ralph.ewerth}@tib.eu

Abstract

The detection of offensive, hateful content on social media is a challenging problem that affects many online users on a daily basis. Hateful content is often used to target a group of people based on ethnicity, gender, religion and other factors. The hate or contempt toward women has been increasing on social platforms. Misogynous content detection is especially challenging when textual and visual modalities are combined to form a single context, e.g., an overlay text embedded on top of an image, also known as *meme*. In this paper, we present a multimodal architecture that combines textual and visual features to detect misogynous memes. The proposed architecture is evaluated in the *SemEval-2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification* challenge under the team name *TIB-VA*. We obtained the best result in the *Task-B* where the challenge is to classify whether a given document is misogynous and further identify the following sub-classes: *shaming*, *stereotype*, *objectification*, and *violence*.

1 Introduction

Detection of hate speech has become a fundamental problem for many social media platforms such as Twitter, Facebook, and Instagram. There have been many efforts by the research community and companies to identify the applicability of advanced solutions. In general, hate speech is defined as *a hateful language targeted at a group or individuals based on specific characteristics such as religion, ethnicity, origin, sexual orientation, gender, physical appearance, disability or disease*. The hatred or contempt expressed towards women has been drastically increasing, as reported by Plan International (2020) and Vogels (2021). Detection of such misogynous content requires large-scale automatic solutions (Gasparini et al., 2018; Suryawanshi et al., 2020; Menini et al., 2020) and comprehensive annotation processes (Zeinert et al., 2021).

The detection of hateful content has been mainly studied from the textual perspective based on the Computational Linguistics and Natural Language Processing (NLP) fields. However, hateful content on social media can be found in other forms, such as videos, a combination of text and images, or emoticons. Misogynous content detection is especially challenging when textual and visual modalities are combined in a single context, e.g., an overlay text embedded on top of an image, also known as *meme*. Recent efforts in multimodal representation learning (Lu et al., 2019; Radford et al., 2021) pushed the boundaries of solving such problems by combining visual and textual representations of the given content. Several datasets have been proposed using multimodal data (Gomez et al., 2020; Kiela et al., 2020b; Sharma et al., 2020; Pramanick et al., 2021; Suryawanshi et al., 2020; Menini et al., 2020) for various tasks related to hate speech. Each dataset includes an image and corresponding text, which is either an overlay text embedded on an image or a separate accompanying text such as tweet text. In contrast to existing datasets based on memes (Kiela et al., 2020b; Sharma et al., 2020; Pramanick et al., 2021; Suryawanshi et al., 2020), the addressed task in this paper aims to identify misogyny in memes specifically. Among the previously mentioned work, only the dataset from Menini et al. (2020) is intended for misogyny detection, in which the text is in the Italian language. In terms of dataset size, the dataset by Gomez et al. (2020) contains approximately 150,000 image-text pairs, while other datasets have moderate sizes that range between two and ten thousand image-text pairs. Moreover, existing model architectures that are evaluated on such benchmark datasets use a combination of various textual and visual features extracted from pre-trained visual and textual models (Kiela et al., 2020a).

The *SemEval-2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification* (Fersini et al.,

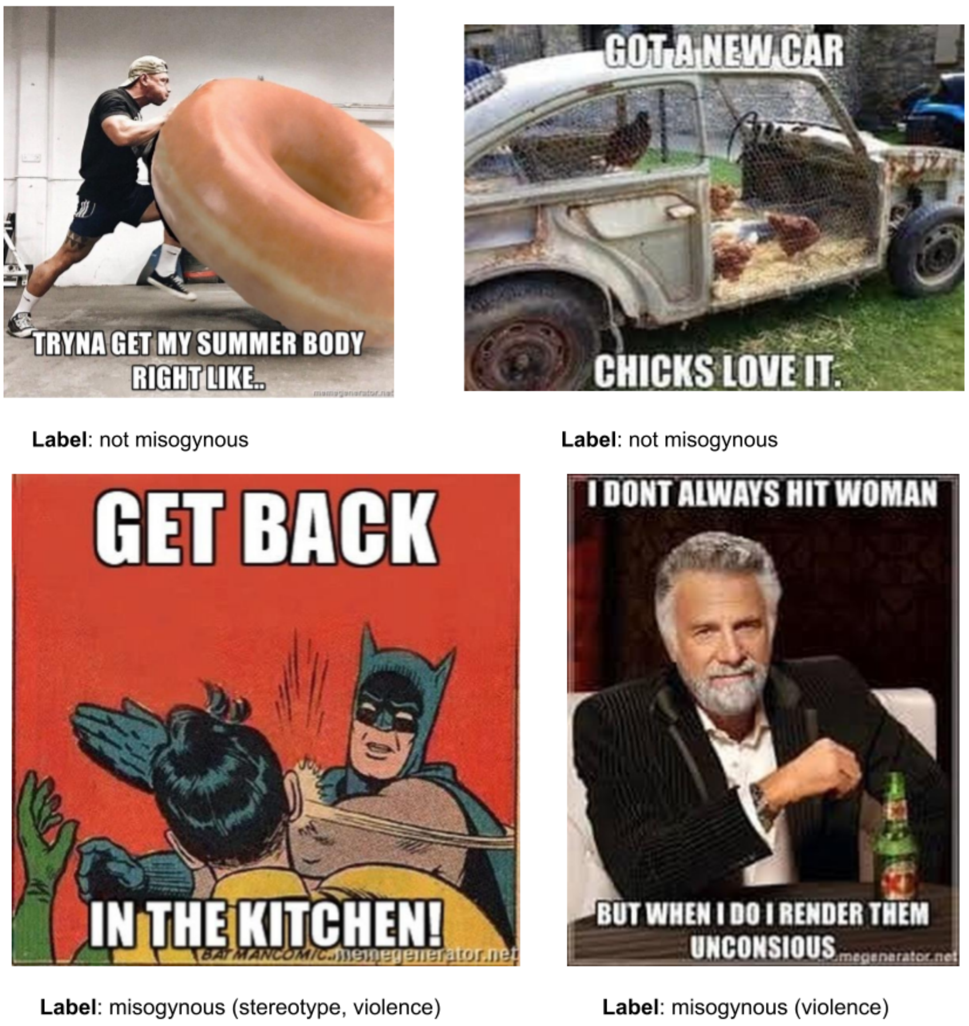

TRYNA GET MY SUMMER BODY RIGHT LIKE..
GOT A NEW CAR
CHICKS LOVE IT.
Label: not misogynous
Label: not misogynous
GET BACK
IN THE KITCHEN!
I DONT ALWAYS HIT WOMAN
BUT WHEN I DO I RENDER THEM UNCONSIOUS
Label: misogynous (stereotype, violence)
Label: misogynous (violence)

Figure 1: Four data samples from *MAMI - Multimedia Automatic Misogyny Identification* with their corresponding class labels. Misogynous samples have additional sub-classes from *stereotype*, *shaming*, *objectification*, and *violence*.

2022)[1] is a new challenge dataset that focuses on identifying misogynous memes. The memes in this dataset are composed of an image with an overlay text. Some samples with their corresponding class labels are shown in Figure 1. The dataset includes two sub-tasks as described below.

- Task-A: a basic task about misogynous meme identification, where a meme should be categorized either as misogynous or not misogynous
- Task-B: an advanced task, where the type of misogyny should be recognized among potential overlapping categories such as stereotype, shaming, objectification and violence.

[1]https://competitions.codalab.org/competitions/34175

In this paper, we present our model architecture for which we submitted results under the team name *TIB-VA*. The model architecture is based on a neural model that uses pre-trained multimodal features to encode visual and textual content and combines them with an LSTM (Long-short Term Memory) layer. Our proposed solution obtained the best result (together with two other teams) on the *Task-B*.

The remainder of the paper is structured as follows. In Section 2, we describe the proposed model architecture. In Section 3, the experimental setup, dataset details, as well as evaluations of the model architecture are described in detail. Finally, Section 4 concludes the paper and outlines areas for future work.

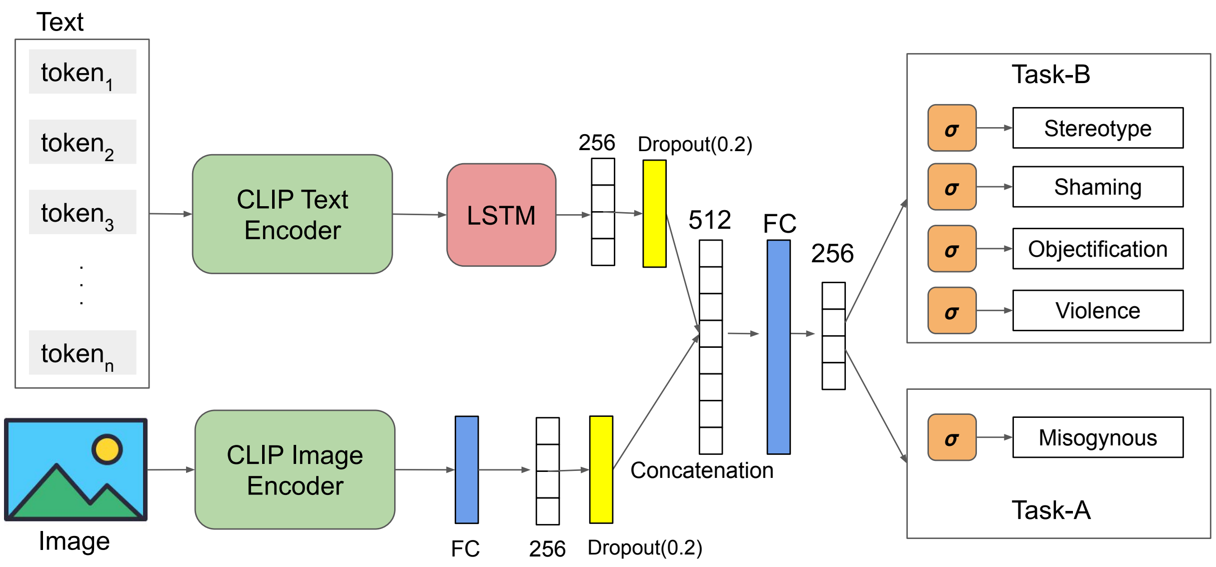


Figure 2: The model architecture that combines textual and visual features to output probabilities for Task-A (misogynous) and Task-B (stereotype, shaming, objectification, violence). FC: Fully connected layer, σ: sigmoid function.

2 Multimodal Architecture

Our model architecture is a neural model that uses a CLIP (Radford et al., 2021) model to extract textual and visual feature representations, which is pre-trained on over 400 million image-text pairs. The goal is to investigate whether this model is applicable to identify misogynous content in memes where both visual and textual content are considered. We used recently available *ViT-L/14* variant of CLIP. The tokens in the overlay text and the image are fed into *CLIP Text Encoder* and *CLIP Image Encoder* respectively. The text encoder outputs a sequence of 768-dimensional vectors for each input token. These token vectors are then fed into an LSTM layer with a size of 256. This layer is another essential part of the proposed architecture. It learns the contextual relatedness among tokens in the text by combining all token representations extracted from the CLIP text encoder branch. The output from the image encoder is fed into a fully-connected layer with a size of 256. The output from an LSTM layer for text and output from the fully-connected layer for the image are fed into separate dropout layers (dropout rate of 0.2), the outputs are concatenated, and then fed into another fully connected layer with a size of 256. The final vector representation is then fed into separate sigmoid functions for each task. For Task-A, the sigmoid outputs a single value that indicates the probability of misogyny. For Task-B, each sub-class of misogyny (stereotype, shaming, violence, objectification) has a separate sigmoid function that outputs a probability value for the corresponding class. The model architecture is shown in Figure 2. The source code of the described model is shared publicly with the community[2].

3 Experimental Setup and Results

3.1 Dataset

The *SemEval-2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification* (Fersini et al., 2022) aims at identifying misogynous memes by taking into account both textual and visual content. Samples from the dataset are given in Figure 1. The dataset includes the overlay text extracted from an image. The challenge is composed of two sub-tasks. Task-A is about predicting whether a given meme is misogynous or not. Task-B requires models to identify sub-classes of misogyny (stereotype, shaming, violence, objectification) in cases where a given meme is misogynous. The samples in Task-B can have multiple labels where a meme can have a single or all of the above sub-classes of misogyny. The train and test splits have 10 000 and 1000 samples, respectively. The distribution of samples for the corresponding two sub-tasks is given in Table 1.

[2]`https://github.com/TIBHannover/multimodal-misogyny-detection-mami-2022`

Splits	Task-A		Task-B				Total
	Misogynous	NOT	Shaming	Objectification	Violence	Stereotype	
Train	5000	5000	1274	2202	953	2810	10 000
Test	500	500	146	348	153	350	1000

Table 1: Distribution of samples in Task-A and Task-B for train and test splits in the *MAMI - Multimedia Automatic Misogyny Identification* dataset.

3.2 Experimental Setup

Training Process: The model architecture is trained using Adam optimizer (Kingma and Ba, 2015) with a learning rate of *1e-4*, a batch size of *64* for maximum of *20* epochs. We decrease the learning by half after every five epochs. We use 10% of the training split for validation to find the optimal hyper-parameters.

Implementation: The model architecture is implemented in Python using the PyTorch library.

Team	Task-A	Task-B
Ours (TIB-VA)	0.734	**0.731**
SRC-B	**0.834**	**0.731**
PAFC	0.755	**0.731**
DD-TIG	0.794	0.728
NLPros	0.771	0.720
R2D2	0.757	0.690

Table 2: Experimental results for the selected top-performing teams on the *MAMI* dataset. The results on Task-A and Task-B are Macro-F1 and Weighted F1 measures, respectively.

3.3 Results

The official evaluation results[3] for the top-performing teams are presented in Table 2. The results on Task-A and Task-B are macro-averaged F1 and weighted-average F1 measures, respectively. Our model architecture (team *TIB-VA*) achieves the best result (0.731) on Task-B along with other two teams: *SRC-B* and *PAFC*. The *SRC-B* team has the highest performance on Task-A. Our results on Task-A are ten points below the best result from the team *SRC-B*. Despite this gap in Task-A, our result is still among the top 20 percentile of all submitted results.

4 Conclusion

In this paper, we have presented a multimodal model architecture that uses image and text features to detect misogynous memes. The proposed solution is built on the pre-trained CLIP model to extract features for encoding textual and visual content. While the presented solution does not yield top results on Task-A, it achieves the best performance in Task-B for identifying sub-classes of misogyny such as stereotype, shaming, objectification, and violence. In future work, we will explore the combination of multiple multimodal features that measure different aspects of visual content such as violence, nudity or specific objects and scene-specific content.

[3]https://competitions.codalab.org/competitions/34175#results

Acknowledgements

This work has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 812997 (CLEOPATRA ITN).

References

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Francesca Gasparini, Ilaria Erba, Elisabetta Fersini, and Silvia Corchs. 2018. Multimodal classification of sexist advertisements. In *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018 - Volume 1: DCNET, ICE-B, OPTICS, SIGMAP and WINSYS, Porto, Portugal, July 26-28, 2018*, pages 565–572. SciTePress.

Raul Gomez, Jaume Gibert, Lluís Gómez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *IEEE Winter Conference on Applications of Computer Vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020*, pages 1459–1467. IEEE.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Casey A. Fitzpatrick, Peter Bull, Greg Lipstein, Tony Nelli, Ron Zhu, Niklas Muennighoff, Riza Velioglu, Jewgeni Rose, Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, Helen Yannakoudakis, Vlad Sandulescu, Umut Ozertem, Patrick Pantel, Lucia Specia, and Devi Parikh. 2020a. The hateful memes challenge: Competition report. In *NeurIPS 2020 Competition and Demonstration Track, 6-12 December 2020, Virtual Event / Vancouver, BC, Canada*, volume 133 of *Proceedings of Machine Learning Research*, pages 344–360. PMLR.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020b. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Stefano Menini, Alessio Palmero Aprosio, and Sara Tonelli. 2020. A multimodal dataset of images and text to study abusive language. In *Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020, Bologna, Italy, March 1-3, 2021*, volume 2769 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Plan International. 2020. Free to be online? Online; accessed February 2022.

Shraman Pramanick, Dimitar Dimitrov, Rituparna Mukherjee, Shivam Sharma, Md. Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021. Detecting harmful memes and their targets. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2783–2796. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Björn Gambäck. 2020. Semeval-2020 task 8: Memotion analysis- the visuo-lingual metaphor! In *Proceedings of the Fourteenth Workshop on Semantic Evaluation, SemEval@COLING 2020, Barcelona (online), December 12-13, 2020*, pages 759–773. International Committee for Computational Linguistics.

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, TRAC@LREC 2020, Marseille, France, May 2020*, pages 32–41. European Language Resources Association (ELRA).

Emily A. Vogels. 2021. The State of Online Harassment. Pew Research Center. Online; accessed February 2022.

Philine Zeinert, Nanna Inie, and Leon Derczynski. 2021. Annotating online misogyny. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3181–3197. Association for Computational Linguistics.

R2D2 at SemEval-2022 Task 5: Attention is only as good as its Values! A multimodal system for identifying misogynist memes

Mayukh Sharma, Ilanthenral Kandasamy and **W.B. Vasantha**
School of Computer Science and Engineering
Vellore Institute of Technology
Vellore - 632014, Tamil Nadu, India
04mayukh@gmail.com, ilanthenral.k@vit.ac.in, vasantha.wb@vit.ac.in

Abstract

This paper describes the multimodal deep learning system proposed for SemEval 2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification. We participated in both Subtasks, i.e. Subtask A: Misogynous meme identification, and Subtask B: Identifying type of misogyny among potential overlapping categories (stereotype, shaming, objectification, violence). The proposed architecture uses pre-trained models as feature extractors for text and images. We use these features to learn multimodal representation using methods like concatenation and scaled dot product attention. Classification layers are used on fused features as per the subtask definition. We also performed experiments using unimodal models for setting up comparative baselines. Our best performing system achieved an F1 score of 0.757 and was ranked 3rd in Subtask A. On Subtask B, our system performed well with an F1 score of 0.690 and was ranked 10th on the leaderboard. We further show extensive experiments using combinations of different pre-trained models which will be helpful as baselines for future work.

1 Introduction

Internet and social media sites have played an integral role in bringing people together by providing a simple yet effective way of communication. Over recent times, internet memes have become a popular choice for sharing sentiment on the internet. A meme is an approach, concept, idea, or style that spreads through social media within a society, often to express a trend, topic, or significance represented by it (Peirson and Tolunay, 2018). Memes shared on the internet are often harmless and used to express humour; however, recent trends have increased their usage to spread hate or cause social unrest (Lippe et al., 2020). Hate speech and, in particular, hate against women has seen an exponential rise in social media platforms (Pamungkas et al., 2020). Misogyny, a subset of hate-speech (Safi Samghabadi et al., 2020), is defined as hate or prejudice against women, which can be manifested in numerous ways, including social exclusion, sex discrimination, hostility, patriarchy, male privilege, belittling of women, disenfranchisement of women, violence against women, and sexual objectification(Anzovino et al., 2018). Women have a strong presence online, particularly on Instagram and Twitter. Women use social media multiple times a day compared to men(Fersini et al., 2020). This makes it extremely important to identify and remove such content to make the internet safer for women.

Efforts have been made to identify misogynous textual content on social media (Anzovino et al., 2018) (Fersini et al., 2018b) (Pamungkas et al., 2020) (Hewitt et al., 2016), however, no efforts have been made to identify the misogynous content spanning multiple modalities like memes. Memes are uniquely multimodal and convey information using images and text. The multimodal nature of memes allows them to combine harmless texts/images into misogynist memes when used together. This poses an exciting challenge as memes require joint language and visual understanding to infer their true meaning. SemEval 2022 Task 5: MAMI (Fersini et al., 2022) draws attention to the problem of identifying misogynous memes and further identifying the type of misogyny. The task provides a dataset of misogynist memes and its type. Both images and the corresponding text was available as a source of information; the text content of the provided dataset was in English.

Our proposed system for both subtasks uses the late fusion of visual and textual features obtained from pre-trained models. We use separate pre-trained models for each modality, i.e. text and image, and fuse the features to learn multimodal representation for memes. We experimented with simple concatenation of image and text features; and scaled dot product attention (Vaswani et al., 2017), followed by a convolution layer to learn multimodal features. Once multimodal features are learnt, we stack classification layers on them as per the subtask requirements. We also performed extensive experiments using a single modality (text/image) to set up comparative baselines. We used ViT (Vision Transformer) for image feature extraction. We experimented with various PLMs for textual feature extraction like Bidirectional Encoder Representations from Transformers(BERT), Robustly Optimized BERT Pretraining Approach(RoBERTa), MPNet, and Decoding-enhanced BERT with disentangled attention(DeBERTa).

The results of unimodal experiments showed that text-only models had a superior performance than image-only models. We experimented with feature concatenation and scaler dot product attention for multimodal models. Feature concatenation led to only minor performance gains. In case of scaler dot product attention, choice of query(Q) (Vaswani et al., 2017) turned out to be an essential factor. Our experiments showed that image features as query performed significantly well for all models and outperformed all unimodal and concatenation baselines. Our best performing model was a voting ensemble of attention based multimodal models and achieved an F1 score of 0.757 with a 3rd rank on the official leaderboard for Subtask A. For Subtask B we used BERT and ViT based attention model, which performed well, attaining an F1 score of 0.690 and 10th rank on the leaderboard. Our code available at GitHub[1] for method replicability.

2 Background

Identifying misogynous content is critical to making the internet accessible and safe for women. In recent times there has been an exponential rise in hateful content and, in particular, the phenomenon of hate against women on social media (Pamungkas et al., 2020) (Hewitt et al., 2016). There have been previous attempts to identify hate/toxic content on social media platforms ((Zampieri et al., 2020) (Zampieri et al., 2019) (Sharma et al., 2021a) (Pavlopoulos et al., 2021) but none deal specifically with identifying the hate against women. The first benchmark dataset to identify misogynous content was proposed in (Anzovino et al., 2018). The task and the papers of AMI@Evalita 2018 (Fersini et al., 2018a) and AMI@IberEval2018 (Fersini et al., 2018b) highlight the difficulties and barriers involved in automatically identifying misogynist content on social media. Workshop on Trolling, Aggression and Cyberbullying (TRAC) shared task (Kumar et al., 2020) contained a subtask to identify gender-based identification of hateful content.

Type	Misogynous	Not misogynous	Total
Train	4742	4758	9500
Validation	258	242	500
Test	500	500	1000

Table 1: Dataset Statistics for Subtask A

There has been a rise in multimodal content over the internet in the form of memes. Most efforts to identify toxic and misogynous content consider only the textual content. Due to the rapid increase of multimodal content, efforts have been made to analyse it. Memotion analysis (Sharma et al., 2020a) aimed to perform sentiment analysis on internet memes. It involved identifying offensive sentiment as part of its Subtasks. (Sharma et al., 2020b) used a feature fusion model using LSTMs, GRUs with attention and attained the best results in identifying offensive memes. Hateful memes challenge (Kiela et al., 2020) aimed to study and identify the hateful nature of internet memes. Analysing memes is an intrinsically difficult task as it requires multimodal reasoning capable of understanding textual and visual features. The common strategy used in analysing memes involved learning features for each modality and then fusing the features to represent joint features. Work done in (Pranesh and Shekhar, 2020) uses the simple concatenation of text and visual features for meme sentiment analysis. (Lippe et al., 2020) used early fusion models like LXMERT (Tan and Bansal, 2019), UNITER (Chen et al., 2020) uses Transformer (Vaswani et al., 2017) architecture and neural attention to learn joint representation.

Social media provides a platform for numerous people to express and share their thoughts. Misogyny which can be simplified as hate or prejudice against women, is on the rise in social media platforms. Hence, it is crucial to identify and remove such content from the internet. Attempts have been made to identify hateful/toxic and misogynist content on the internet, but none focuses on multimodal

[1]`https://github.com/04mayukh/R2D2-at-SemEval-2022-Task-5-MAMI`

Type	Shame	Stereotype	Objectification	Violence	Total
Train	1271	2810	2201	953	5000
Validation	60	141	105	47	250
Test	146	350	348	153	1000

Table 2: Dataset Statistics for Subtask B

content like memes. SemEval 2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification (Fersini et al., 2022) aims to study the misogynist nature of memes and is divided into two subtasks which we define as:

Subtask A: Given a labelled dataset D of internet memes and their text, the objective of the task is to learn a classification function that can predict if a meme is misogynous or not.

Subtask B: Given a labelled dataset D of internet memes and their text, the objective of the task is to learn a multilabel classification function that can predict the type of misogyny M for a given misogynous meme, where $M \in \{$stereotype, shaming, objectification and violence$\}$.

Dataset Statistics: The dataset for the task consisted of internet memes and their textual content. For Subtask A, the memes were labelled as misogynous/non-misogynous. Misogynous memes from Subtask A were further labelled into the type of misogyny. Subtask B involved recognising the type of misogyny from overlapping categories like stereotype, shaming, objectification, and violence. We also split the provided dataset into train and validation sets for training and evaluating the models before using them to make predictions on the test set. Table 1 and Table 2 show the dataset statistics for Subtask A and Subtask B.

3 System Overview

3.1 *Pre-trained Models:*

Finetuning (Qiu et al., 2020) pre-trained language models has become a popular approach in the deep learning community (Sharma et al., 2021b). It is a form of transfer learning that utilizes models trained on enormous amounts of unannotated text data to learn general-purpose representations. These models are then finetuned on downstream tasks. In recent times there has been a rapid rise in pre-trained models in Natural Language Processing (NLP). The knowledge from these models can be easily transferred to tasks where small amounts of data are present, making them extremely useful. The maximum of these PLMs like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), MPNet (Song et al., 2020), DeBERTa (He et al., 2021) are based on transformers. Pre-trained models used in computer vision mostly rely on convolutional networks. Architectures like classic ResNet (He et al., 2016) have attained state of the art performance in large scale image recognition tasks (Kolesnikov et al., 2020) (Xie et al., 2020). Recently the transformer architecture has also been used in computer vision and performs at par with convolutional networks (Touvron et al., 2021) (Dosovitskiy et al., 2021) (Bao et al., 2022). The use of transformers in computer vision has also led to multimodal pre-trained models (Kim et al., 2021) (Li et al., 2019) (Tan and Bansal, 2019). Pre-trained models provide an efficient and scalable way to use large-scale learning to simple downstream tasks efficiently. Next, we describe the pre-trained models used in our multimodal system.

3.2 *Brief overview of Pre-trained models:*

Vision Transformer (ViT): ViT (Dosovitskiy et al., 2021) was proposed by Google and aimed to use the transformer architecture with minimal changes to computer vision tasks. Transformers use sequences to process data and cannot process grid-structured data. The images in the transformer were converted into smaller image patches and used as sequences. Trainable positional encodings were used to retain positional information of smaller image patches. These positional encodings help to learn the relationship between smaller image patches. Finally, the whole model is pre-trained as a classification task on the ImageNet dataset (Russakovsky et al., 2015).

BERT: It stands for Bidirectional Encoder Representations from Transformers (Devlin et al., 2019) is a PLM based on the transformer architecture. It is pre-trained on text corpus using Masked Language Modelling (MLM) and Next Sentence Prediction (NSP) objective.

RoBERTa: A Robustly Optimized BERT Pretraining Approach (Liu et al., 2019) was developed by Facebook. They used the BERT architecture with few modifications and obtained better performance. They used dynamic masking in their pre-training and removed the NSP objective. They also trained the model using a larger batch size with more data for longer durations.

MPNet: MPNet (Song et al., 2020) was proposed by Microsoft. Most language models are pre-trained using either MLM or permuted language modelling objectives. MPNet makes the best of both permuted language modelling and MLM. It proposed a unified view of MLM and permuted language modelling by splitting and rearranging the tokens into predicted and non-predicted parts. It uses MLM to see the positional information of complete sentences and permuted language modelling to model dependency among predicted tokens.

DeBERTa: DeBERTa stands for decoding-enhanced BERT with disentangled attention (He et al., 2021), was proposed by Microsoft and outperformed human performance on the SuperGlue benchmark (Wang et al., 2019). It used disentangled attention where two vectors are used, one to represent the content and the other to store the positional information. Attention weights are calculated using disentangled matrices on their content and relative positions. It also uses an enhanced mask decoder during pre-training to incorporate absolute positions while predicting masked tokens.

3.3 *Unimodal models (baselines):*

Meme classification is an intrinsically complex problem due to textual and visual cues. Memes can convey a message using image, text, or both, thus requiring textual, visual, and multimodal understanding. We first modelled misogyny detection using purely unimodal approaches to form comparative baselines as part of our experiments. BERT, RoBERTa, MPNet, DeBERTa were used for meme text classification, and ViT was used for meme image classification. The unimodal baselines were also helpful in understanding the predictive power of text vs image features. We experimented with unimodal models only for Subtask A.

3.4 *Multimodal models:*

The multimodal nature of memes makes it extremely difficult to understand their true meaning. They may contain a combination of completely different visual and textual content, which, when joined, turn out to be misogynist in nature. To understand this multimodal nature of memes, we used two different techniques to join visual and textual representations to learn multimodal features, which we will discuss next.

Feature concatenation: To learn jointly from image and text features, we used late fusion to concatenate the features learnt by image and text pre-trained models. Concatenated features are then fed to the final classification layer to label the meme as misogynistic (Subtask A) or identify the type of misogyny (Subtask B). The image and text pre-trained models are jointly finetuned with the classification layer using the classification objective.

Attention-based feature fusion: Attention has been widely used in various NLP tasks. It forms the critical component of transformer architecture. The main idea behind attention is to learn representation for a given feature based on its relative relevance with respect to other features. We use the same idea to learn joint image-text features using attention mechanism. We use scaler dot product attention (Vaswani et al., 2017) which uses concept of queries(Q), keys(K), and values(V) and is defined as:

$$Attention(Q, K, V) = softmax(QK^T/\sqrt{d_k})V$$

where dimension of Q and K is (N, d_k), dimension of V is (N, d_v), and N is sequence length. The output of the attention step has the dimension of (N, d_v), which represent the context vectors. The language and vision models we used both utilised transformer architecture. As we know, transformers work with data sequences, which helped us get highly localised feature representations, allowing us to use attention to fuse visual and text features. The features from pre-trained text models represent the information corresponding to input tokens. The visual features from ViT represent information corresponding to $N \times N$ patches from the input image. We use the dot product attention defined above on these set of features to learn the multimodal representation.

Another essential aspect of scaler dot product attention is the choice of query and key during attention. In transformers, self-attention is calculated, which makes this choice trivial. However, while using attention on visual and text features, it is essential to note that attention is not commutative. Therefore, we performed experiments using both visual and textual features as the query parameter in attention. The context vectors obtained from the attention layer were then passed through a one-dimensional convolution layer to learn final

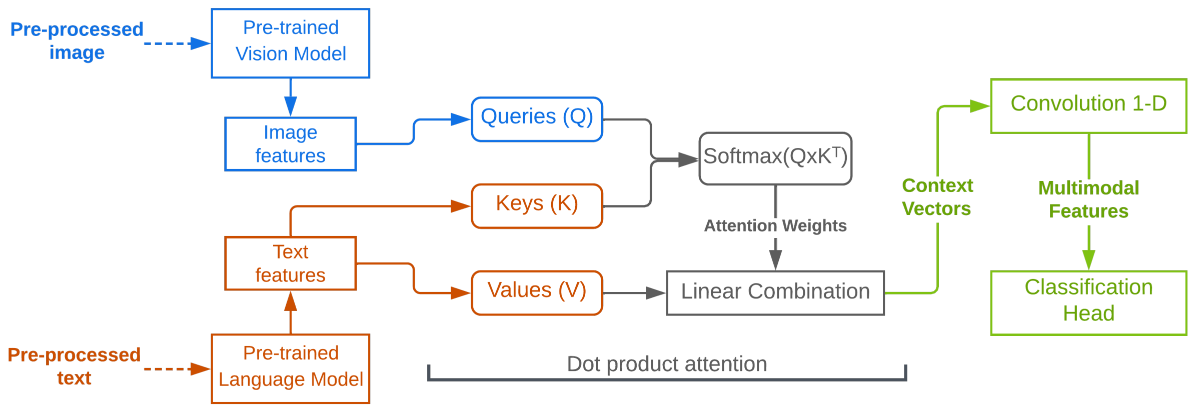


Figure 1: Architecture of our model using attention with image features as query.

multimodal features. These features were finally flattened and fed to the final classification layers. Figure 1 shows the architecture diagram of our attention based multimodal model using image features as query.

3.5 *Classification layers and finetuning:*

We finetuned the unimodal text models by stacking a simple dense and batch normalisation layer followed by a one-neuron classifier on top of features learnt by pre-trained models. We used the features from [CLS] token in the case of BERT and RoBERTa and start the token(<s>) in the case of MPNet and DeBERTa. For ViT we used the [class] token, which is taken as image representation. We pass the joint visual and text features through a batch normalisation and simple dense layer followed by a one-neuron classifier for multimodal models. Visual and text models along with concatenation/attention layer were wrapped into a single model for finetuning multimodal models.

4 Experimental setup

4.1 *Pre-processing images and texts*

Text: We used the ekphrasis (Baziotis et al., 2017) library for text pre-processing. It normalises time, date, numbers to a standard format and corrects misspelt words. Chatwords are commonly used in memes, so we converted them into their full forms. PLMs need text to be tokenised before it can be fed to them. We used Hugging Face's (Wolf et al., 2020) implementation of Fast tokenisers[2] for each pre-trained model.

Image: Images need to be pre-processed before being fed to ViT. We first resized the image to 224×224. We also divided the pixel values by 255 to bring them within a range of 0-1. Finally, the images were normalised using the mean and standard deviation of 0.5 across all channels.

4.2 *Task-wise model definition:*

Subtask A: We experimented with unimodal and multimodal techniques. Our unimodal baselines used BERT, RoBERTa, MPNet, DeBERTa for text and ViT for images. We extracted the features from these models and passed them through a dense layer with 32 neurons, followed by a batch normalisation layer. Finally, we used a classification layer with a single neuron and sigmoid activation to label the input as misogynous/not misogynous. For multimodal models, we experimented with concatenation as well as attention using BERT and ViT initially. We passed the fused features through a batch normalisation and dense layer consisting of 64 neurons, followed by a single neuron classification and sigmoid activation. Further, we experimented using a combination of RoBERTa, MPNet, and DeBERTa with ViT using attention mechanism (image features as query). The one-dimensional convolution layer in multimodal models used 32 filters with a kernel size of 30 and stride 15.

Subtask B: We used only the multimodal models using concatenation and attention to learn multimodal features. Subtask B was a multilabel task where a misogynous meme could belong to more than one category: stereotype, shaming, objectification, and violence. We trained a single model to classify the memes into the given categories. We created a multi-branch model where each branch tries to predict if meme belongs to one of the given categories. The branch takes the independent text

[2]Hugging Face's Fast Tokenizers

and visual features fused using simple concatenation or a combination of attention and 1-D Convolution. The fused multimodal features for each branch are then passed through a 32-neuron dense layer, batch normalisation layer and finally through a single neuron classification layer with sigmoid activation.

4.3 *Hyperparameters and training:*

We developed our models using TensorFlow[3], Keras [4] (Chollet et al., 2015) and Hugging Face's [5] implementation of transformer[6] (Wolf et al., 2020) models. The models were trained using GPU/TPU on Google Colab. We fixed the sequence length to 80 tokens across both subtasks for text modality. Sequences greater or shorter than 80 tokens were accordingly truncated or padded. We used their base versions for all PLMs; for ViT we used the base model with the patch size of 16×16 and input image size of $224\times$ 224. Finetuning was performed using Adam (Kingma and Ba, 2015) optimiser against a binary cross-entropy loss. We experimented with learning rates ranging from 2e-5 to 5e-5 with a batch size of 128 for TPU and 16 for GPU. Finetuning was done for ten epochs, and weights corresponding to the best results on the validation set were used to make predictions on the test set. For Subtask A, we finetuned the entire dataset containing misogynous as well as non-misogynous memes. For Subtask B, we trained the model only on misogynous memes to identify the type of misogyny. For evaluation on the test set, we used a hierarchical approach where we made predictions only on samples predicted as misogynous from the best performing model on Subtask A.

4.4 *Evaluation metric:*

Subtask A used the macro-averaged F1 score to evaluate the model's performance. F1 scores were calculated individually for each class and then averaged to give the Macro F1 score. For Subtask B, the weighted-average F1 measure is used as the evaluation metric. F1 scores are computed for each label, and then the weighted average is computed based on true instances belonging to each label category.

[3]https://www.tensorflow.org/
[4]https://keras.io
[5]https://huggingface.co
[6]https://huggingface.co/transformers

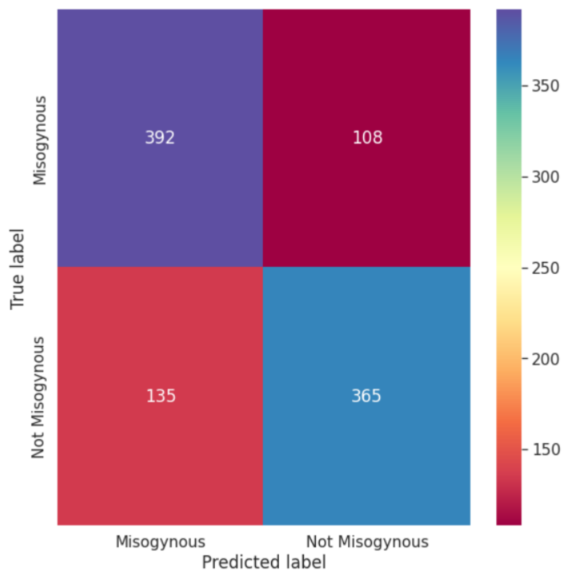


Figure 2: Confusion Matrix for Subtask A (ensemble model)

5 Results and Analysis

Table 3 and Table 4 contain the results of our models for Subtask A and Subtask B. In Subtask A our best performing model was a voting ensemble of attention (image as a query) models trained with different seed values attaining an F1 score of 0.757 and was ranked 3rd on the leaderboard. Figure 2 shows the confusion matrix for our ensemble model. In Subtask B we used a hierarchical model. We made predictions only for samples classified as misogynous by our best performing ensemble model on Subtask A. Our BERT + ViT model using attention (image as a query) performed best with an F1 score of 0.690 and was ranked 10th on the leaderboard.

Table 3 contains a comparative analysis of unimodal vs multimodal techniques we used as part of our experiments for Subtask A. If we compare unimodal models, we can see that text-based models have better performance than the image-based models on both development and test sets. It points to the possibility that text present in memes is a prominent factor in identifying misogynous content. DeBERTa outperforms other models by a considerable margin among text models, performing almost equivalent to the multimodal concatenation model. For multimodal models, we performed initial sets of experiments using BERT and ViT over different methods of fusing the modalities. The simple concatenation worked well and provided a slight improvement when compared to unimodal

Model	Type	Development					Test				
		Precision*	Recall*	Precision	Recall	F1	Precision*	Recall*	Precision	Recall	F1
BERT	TEXT ONLY	.813	.831	.829	.830	.830	.618	.786	.662	.650	.643
RoBERTa		.813	.792	.816	.815	.815	.657	.518	.630	.624	.620
MPNet		.821	.835	.835	.836	.836	.645	.712	.662	.660	.659
DeBERTa		.850	.792	.837	.834	.835	.665	.734	.684	.682	.6811
ViT	IMAGE ONLY	.792	.665	.744	.740	.737	.600	.820	.658	.637	.624
ViT + BERT	Concatenation	.860	.835	.858	.857	.857	.646	.852	.714	.692	.684
ViT + BERT	Attention(QUERY-Text)	.802	.659	.748	.743	.739	.618	.920	.731	.676	.655
ViT + BERT	Attention(QUERY-Image)	.921	.771	.857	.851	.848	.682	.836	.735	.723	.719
ViT + MPNet		.930	.725	.846	.833	.829	.700	.822	.742	.734	.732
ViT + RoBERTa		.928	.698	.836	.820	.814	.700	.822	.742	.735	.733
ViT + DeBERTa		.949	.725	.857	.842	.837	.683	.824	.731	.721	.718
ENSEMBLE		NA					.771	.730	.758	.757	.757

Table 3: Experimental results on development and test set for Subtask A. Metrics for misogynist class are represented using $*$ after the metric name.

Model	Type	F1 Stereotype		F1 Shaming		F1 Objectification		F1 Violence		F1 Weighted Avg.	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
BERT + ViT	Concatenation	.658	.652	.710	.636	.741	.695	.752	.722	.710	.672
BERT + ViT	Attention (QUERY – Image)	.648	.666	.663	.67	.745	.708	.747	.711	.696	.690

Table 4: Experimental results on development and test set for Subtask B.

techniques.

Our next set of experiments used scaler dot product attention using BERT and ViT. As we can see from the results, the choice of query(Q) played a crucial role in calculating the multimodal features using attention. BERT + ViT model using text features as query performed poorly when compared to concatenation and best performing textual models. However, when we use the same architecture and change the query(Q) term to image features, there is a significant gain that outperforms all other models. The attention mechanism uses queries(Q), keys(K) and values(V) to calculate the context vectors. We can observe from the formula defined in section 3 that while the query(Q) and key(K) is used to calculate attention weights, the final features are a linear combination of attention weights over the values(V). Our experiments using unimodal models showed that textual features perform better than visual features, almost as good as concatenation based multimodal models. Using image features as a query allows us to preserve the textual features which are used as values(V) while modelling the correlation between images and text using query-key matching. This might be one of the possible reasons for the better performance of attention-based models with image features as query. We performed further experiments using only attention-based fusion with image features as queries(Q). We used MPNet, RoBERTa, and DeBERTa with ViT, which outperformed all unimodal and multimodal baselines. For Subtask B, we experimented with only concatenation and attention methods and found that attention (image as a query) performed better than concatenation.

6 Conclusion

This paper describes our approach for SemEval 2022 Task 5: MAMI - Multimedia Automatic Misogyny Identification. We propose a dot product attention-based mechanism for learning multimodal representation from independent text/image features. Our work also describes a comprehensive set of experiments using unimodal/multimodal models using different pre-trained models. Our system performed well, attaining 3rd rank on Subtask A and 10th in Subtask B. Our experiments highlight the non-commutative nature of dot product attention, with the choice of the query being a critical design decision. Our results showed that textual features dominate over image features in multimodal understanding. In the future, we would like to explore more on how attentions learn multimodal features and further compare the role of individual modalities in multimodal tasks.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *Natural Language Processing and Information Systems*, pages 57–64, Cham. Springer International Publishing.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. 2022. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations*.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *Computer Vision – ECCV 2020*, pages 104–120, Cham. Springer International Publishing.

François Chollet et al. 2015. Keras. `https://keras.io`.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the evalita 2018 task on automatic misogyny identification (AMI). In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. Ami @ evalita2020: Automatic misogyny identification. In *EVALITA*.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. In *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) co-located with 34th Conference of the Spanish Society for Natural Language Processing (SEPLN 2018), Sevilla, Spain, September 18th, 2018*, volume 2150 of *CEUR Workshop Proceedings*, pages 214–228. CEUR-WS.org.

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

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. {DEBERTA}: {DECODING}-{enhanced} {bert} {with} {disentangled} {attention}. In *International Conference on Learning Representations*.

Sarah Hewitt, T. Tiropanis, and C. Bokhove. 2016. The problem of identifying misogynist language on twitter (and other online social spaces). In *Proceedings of the 8th ACM Conference on Web Science*, WebSci '16, page 333–335, New York, NY, USA. Association for Computing Machinery.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5583–5594. PMLR.

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

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. Big transfer (bit): General visual representation learning. In *Computer Vision – ECCV 2020*, pages 491–507, Cham. Springer International Publishing.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2020. Evaluating aggression identification in social media. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 1–5, Marseille, France. European Language Resources Association (ELRA).

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557.

Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, and Helen Yannakoudakis. 2020. A multimodal framework for the detection of hateful memes. *CoRR*, abs/2012.12871.

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

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

John Pavlopoulos, Jeffrey Sorensen, Léo Laugier, and Ion Androutsopoulos. 2021. SemEval-2021 task 5: Toxic spans detection. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 59–69, Online. Association for Computational Linguistics.

Abel L. V Peirson and E. Meltem Tolunay. 2018. Dank learning: Generating memes using deep neural networks. *CoRR*, abs/1806.04510.

Raj Ratn Pranesh and Ambesh Shekhar. 2020. Memesem:a multi-modal framework for sentimental analysis of meme via transfer learning.

XiPeng Qiu, TianXiang Sun, YiGe Xu, YunFan Shao, Ning Dai, and XuanJing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Niloofar Safi Samghabadi, Parth Patwa, Srinivas PYKL, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and misogyny detection using BERT: A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131, Marseille, France. European Language Resources Association (ELRA).

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Björn Gambäck. 2020a. SemEval-2020 task 8: Memotion analysis- the visuolingual metaphor! In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 759–773, Barcelona (online). International Committee for Computational Linguistics.

Mayukh Sharma, Ilanthenral Kandasamy, and Vasantha Kandasamy. 2021a. Deep learning for predicting neutralities in offensive language identification dataset. *Expert Systems with Applications*, 185:115458.

Mayukh Sharma, Ilanthenral Kandasamy, and W.b. Vasantha. 2020b. Memebusters at SemEval-2020 task 8: Feature fusion model for sentiment analysis on memes using transfer learning. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1163–1171, Barcelona (online). International Committee for Computational Linguistics.

Mayukh Sharma, Ilanthenral Kandasamy, and W.b. Vasantha. 2021b. YoungSheldon at SemEval-2021 task 7: Fine-tuning is all you need. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1146–1152, Online. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *NeurIPS 2020*. ACM.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. 2021. Training data-efficient image transformers and distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR.

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

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

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

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447, Barcelona (online). International Committee for Computational Linguistics.

AIDA-UPM at SemEval-2022 Task 5: Exploring Multimodal Late Information Fusion for Multimedia Automatic Misogyny Identification

Álvaro Huertas-García
Universidad Politecnica de Madrid
Madrid, Spain
Universidad Rey Juan Carlos de Madrid
Madrid, Spain
alvaro.huertas.garcia@alumnos.upm.es

Helena Liz López
Universidad Politecnica de Madrid
Madrid, Spain
Universidad Rey Juan Carlos de Madrid
Madrid, Spain
helena.liz@alumnos.upm.es

Guillermo Villar-Rodríguez
Universidad Politecnica de Madrid
Madrid, Spain
guillermo.villar@upm.es

Alejandro Martín
Universidad Politecnica de Madrid
Madrid, Spain
alejandro.martin@upm.es

Javier Huertas-Tato
Universidad Politecnica de Madrid
Madrid, Spain
javier.huertas.tato@upm.es

David Camacho
Universidad Politecnica de Madrid
Madrid, Spain
david.camacho@upm.es

Abstract

This paper describes the multimodal late fusion model proposed in the SemEval-2022 Multimedia Automatic Misogyny Identification (MAMI) task. The main contribution of this paper is the exploration of different late fusion methods to boost the performance of the combination based on the Transformer-based model and Convolutional Neural Networks (CNNs) for text and image, respectively. Additionally, our findings contribute to a better understanding of the effects of different image preprocessing methods for meme classification. We achieve 0.636 F1-macro average score for the binary sub-task A, and 0.632 F1-macro average score for the multi-label sub-task B. The present findings might help solve the inequality and discrimination women suffer on social media platforms.

1 Introduction

The proposed task, SemEval-2022 Task 5 Multimedia Automatic Misogyny Identification (MAMI) (Fersini et al., 2022) consists in the identification of misogynous memes in English language, taking advantage of both text and images available as a source of information.

Overall, our proposed method consists of a multimodal approach combining different features (e.g., logits, probabilities, embeddings) of a text Transformer-based model and an image CNN model in a late fusion approach. This late fusion step implies that both models are trained and fine-tuned separately to the task. Then, the features from each model are concatenated and jointly used as input for a final classifier that combines their knowledge to obtain a final prediction (see Figure 1). Different preprocessing steps, text and image models, concatenated feature combinations, and classifiers are explored to obtain the final multimodal architecture.

Our presented method has been developed for sub-task A and B from the MAMI competition independently. sub-task A consists of misogynous meme binary classification, where a meme should be categorized either as misogynous or not misogynous. On the other hand, sub-task B requires a more detailed multi-label classification where misogynous content should be recognized among potential overlapping categories such as stereotype, shaming, objectification, and violence.

It is noteworthy that our multimodal late fusion method outperforms single models in both sub-tasks, being more remarkable in complex sub-task B. Similarly, considering that both sub-tasks share the same data, the results of model evaluation on both tasks show how the model trained on complex sub-task B can achieve the same results as a model trained only on binary sub-task A. Therefore, future studies should investigate the complexity and pruning of the required models.

This paper provides new insights into information fusion for tackling multimodal tasks, presenting an in-depth exploration of different late fusion approaches and image processing steps. The presented work might help identify malicious be-

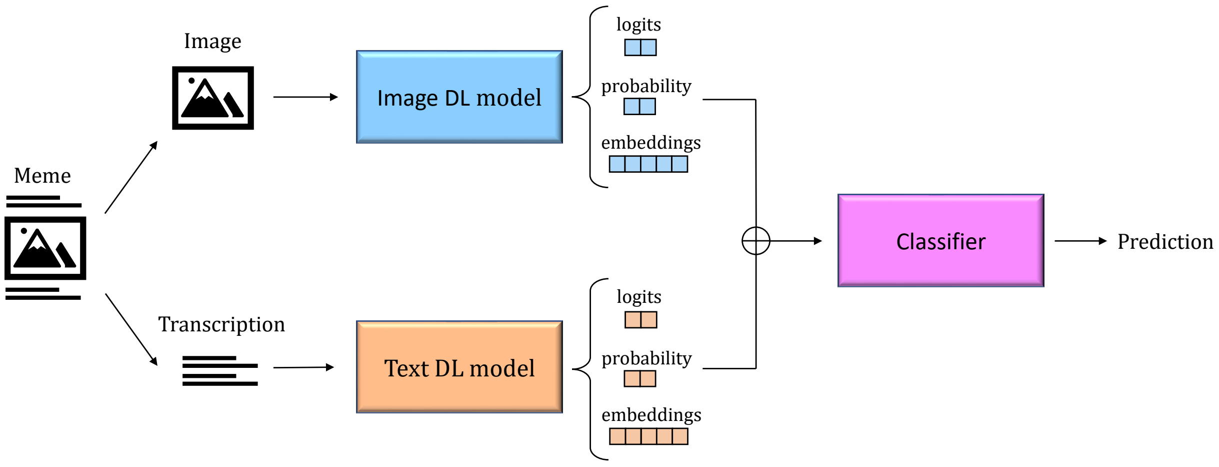


Figure 1: Summarized diagram of the late fusion multimodal system proposed for misogyny detection

haviours towards women on social media.

2 Background

Misogyny comprises every hateful and prejudicial action against women, ranging from discrimination, objectification, violence and disdain affecting women to all the types of manifested male superiority like patriarchy, androcentrism and privilege (Pamungkas et al., 2020). Misogynist posts represent one type of hate speech on Online Social Networks, but it is complex to distinguish them from other sorts of offensive discourses in an automated way (Shushkevich and Cardiff, 2019).

Whereas research shows a survey of methods for misogyny recognition in text including traditional methods and their ensembles and neural networks (Shushkevich and Cardiff, 2019), studies reviewing mysogyny in images are more scarce, and more work can may be found under the generalization of sexist content (Campisi et al., 2018; Fersini et al., 2019) than under the particular topic of misogyny.

These sexist images can be in the form of memes, multimedia content with a humorist goal composed of photos and/or illustrations with some text on it (Sabat et al., 2019). Furthermore, this initially shows that sexist and also misogynist content on social networks do not need just the isolated use of image and text processing but the combination of both. In fact, having images or texts on their own may not lead to hateful speech and only their mix is offensive (Sabat et al., 2019), underlining the major necessity of architectures that encode the global meaning of memes.

On the one hand, the latest advances in NLP for text classification include the use of Transformers, which has demonstrated to be successful in the detection of misogyny (Samghabadi et al., 2020; Aldana-Bobadilla et al., 2021). These distributional models encode the meaning of texts in vectors that also capture the context (Devlin et al., 2018). However, they may not be so optimal for the detection of subclasses, for instance between the absence and the implicit or sarcastic presence of aggressiveness (Samghabadi et al., 2020), or when the female subject that is attacked is not present in the text (Aldana-Bobadilla et al., 2021).

On the other hand, when images are also taken into account, the use of Convolutional Neural Networks (CNNs) for images is still present (Gomez et al., 2020) through image-based state-of-the-art models such as `VGG16`, `ResNet`, `DenseNet` and `Inception`. Specifically, `VGG16` works by itself for the prediction of offensive memes and succeeds when combined with different models for the text inside them (LSTMs, BiLSTMs and CNNs), which were later compared (Aman et al., 2021), but when the comparison is among vision models, depending on the dataset and the multimodal approach followed, `ResNet50` can surpass `VGG16` and `ResNet152` for VGCN-BERT combined pipelines for hateful images detection (Vlad et al., 2020), but `Inception` outperforms the F1-scores from `ResNet50` for multimodal approaches with BiLSTMs for 'troll' (sarcastic or offensive) meme detection (Hossain et al., 2021), and just using `DenseNet` alone for meme emotion recognition can be even better than `ResNet` alone and than either `DenseNet` or `ResNet` in image joining forces with BERT for the textual features (Guo et al., 2020). Alternative implementations for vision are Transformers such as VisualBERT, which represents an architecture with image and text models (Lippe et al., 2020) or CLIP, which predicts the text that better describes the images (Zia et al., 2021).

Recent research has shown that joining the probabilities from the CNN classification and the output statistical variables obtained after training a successive classifier (Huertas-Tato et al., 2022) improves the task. These results invite to use this advance in information fusion for practical applications such as hateful images and, specifically, misogynist memes. In line with this theoretical framework, our work will concatenate the different outputs (e.g, logits, probabilities, last hidden embeddings) from CNN-based image classification and from Transformers-based text classification towards a better performance in MAMI.

3 System overview

As previously mentioned, our proposed approach is depicted in Figure 1. This section details the preprocessing steps, the text and image models employed, and the methodology followed to train the final multimodal late fusion classifier that combines different features.

3.1 Image Preprocessing

To rule out the possibility of text misleading the image CNN model and to enhance its focus on the image, three different preprocessing steps are separately explored. Consequently, three different image models are trained: (1) no preprocessing, (2) blacking out, and (3) inpainting the text from the image. These preprocessing methods make use of EasyOCR[1] and OpenCV (Bradski, 2000). Figure 2 illustrate some examples of the results of these preprocessing steps. Additionally, an application for applying the inpainting preprocessing to images has been publicly published with the aim of contributing to the scientific community[2].

3.2 Text Preprocessing

Ftfy package (Speer, 2019) was used as a preprocessing step for fixing text and ensuring it is uniformly UTF-8 encoded. URLs, emojis, or other native features present in the text are not modified as we consider these characteristics crucial for this task.

[1]https://github.com/JaidedAI/EasyOCR

[2]https://huggingface.co/spaces/Huertas97/Inpaint_Me

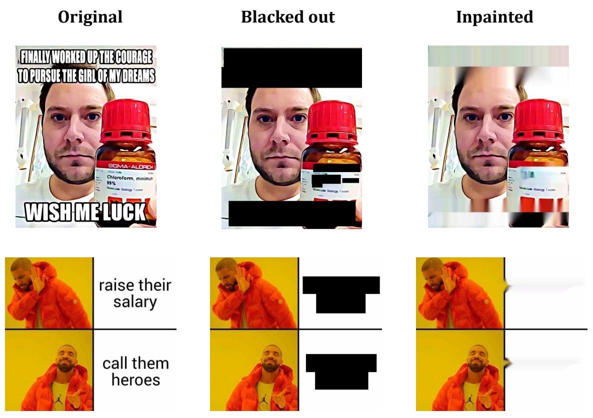


Figure 2: Example of different image preprocessing steps

3.3 Explored Models

According to the models employed, it is worth mentioning that different Transformer-based models publicly available at Hugging Face (Wolf et al., 2020) are evaluated as the textual model:

- `bertweet-base` (Nguyen et al., 2020): large-scale language model pre-trained for English Tweets based on RoBERTa (Liu et al., 2019) pre-training procedure and BERT architecture (Devlin et al., 2019).
- `all-distilroberta-v1`[3]: pre-trained `distilroberta-base` (Sanh et al., 2019) model fine-tuned on a 1B sentence pairs dataset using a contrastive learning objective.
- `all-miniLM-L6-v2`[4]: pre-trained Microsoft `MiniLM` (Wang et al., 2020) model fine-tuned on a 1B sentence pairs dataset using a contrastive learning objective.
- `twitter-roberta-base-offensive` (Barbieri et al., 2020): `roberta-base` model trained on 58M tweets and finetuned for offensive language identification with the TweetEval benchmark.
- `twitter-xlm-roberta-base` (Barbieri et al., 2021): `xlm-roberta-base` model trained on 198M multilingual tweets.

For image processing we have used CNNs, which extract features with convolutional layers and deduce knowledge with dense layer. We use

[3]https://huggingface.co/sentence-transformers/all-distilroberta-v1

[4]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Optimization	**Hyperparameters**	**Values**
Text Model	learning rate	min = 1e-6 , max = 1e-3
	epochs	min = 1, max = 10
	weight decay	min = 0 , max = 1
	gradient accumulation steps	min = 1 , max = 4
	scheduler	constant_schedule constant_schedule_with_warmup linear_schedule_with_warmup cosine_schedule_with_warmup cosine_with_hard_restarts_schedule_with_warmup polynomial_decay_schedule_with_warmup
	optimizer	AdamW
	sliding window	True, False
	pos_weigths*	[2, 1, 1, 2], [2, 0.5, 0.5, 2], [1, 1, 1, 1]
Image Data augmentation	shear range	0.1
	zoom range	0.1
	rotation range	45
	width shift range	0.1
	height shift range	0.1
	horizontal flip	True
	brightness range	0.7-1.1
	channel shift range	0.05
Image Model	optimizer	Adam
	learning rate	0,001
	preprocess_input	True
	pos_weigths*	[1,1,1,1], [1, 3.96, 1.78, 2.27, 5.25]
	percentage of frozen layers	0.1, 0.3*
Auto-sklearn	time_left_for_this_task	60, 120, 500, 3600, 7200
	memory_limit	6072
	exclude	None
	resampling_strategy	Cross Validation 5 folds
	ensemble_size	10

Table 1: Hyperparameters optimized during the development of the proposed approaches. *Only for multi-label subtask B.

different architectures pretrained from state of the art:

- `VGG16` (Simonyan and Zisserman, 2014): it is composed by a feed-forward set of units and is the most straightforward without additional forward connections or auxiliary outputs. However this architecture has to adjust lots of parameters.

- `ResNet50` (He et al., 2016): The main advantage of this architecture is the shortcut connections, these links skip one or more layers, aggregating their output to the outputs of the stacked layers.

- `DenseNet201` (Huang et al., 2017): Instead of adding more layers to the architecture, it increases the number of connections between units, connecting every units with later ones.

- `Inception_v3` (Szegedy et al., 2016): it factorises the convolution into smaller ones (that can be asymmetric) to reduce the cost. Moreover, this architecture has an auxiliary classifier between layers, that acts as regularizer.

- `EfficientNetB0` (Tan and Le, 2019): This architecture uses a compound coefficient to scale all dimensions of depth, width and resolution.

Finally, the Auto-sklearn package (Feurer et al., 2015, 2020) is used to automatically explore a wide range of models and preprocessing approaches available in scikit-learn and identify the best ensemble configuration for the multimodal late fusion step. We opted for this method because it implements Bayesian Optimization for searching the optimal pipeline configuration and Ensemble Selection to choose the suitable model.

3.4 Multimodal late fusion approaches

From the text and image models, three features are used for the late fusion step; the output of the activation function from the last classification layer (i.e., probabilities) and its input (i.e., logits), and the vectorize output representation of the last hidden layer (i.e., embeddings). In order to develop the final multimodal classification predictions, different late fusion approaches for combining these features are considered.

Naive baseline approaches consist of averaging or taking the maximum logit or probability values from both models for each class or label depending on the sub-task. An advanced baseline approach consists of finding the weights for each model that will give the lowest mean square error (MSE) score between multimodal predictions and real values using logits or probabilities. For this purpose, Sequential Least Squares Programming (SQLSP) from Scipy package (Virtanen et al., 2020) is the optimization method used. Finally, logits, probabilities or embeddings from both models are used as input in Bayesian Optimization for searching the optimal pipeline configuration and Ensemble Selection using Auto-sklearn.

4 Experimental Setup

As previously mentioned, both sub-tasks share the same data and the official metric for system evaluation, F1-macro averaged. The dataset consists of 10.000 memes and its corresponding text transcriptions. To develop the proposed approach, balanced data for sub-task A is split into 64% train, 16% validation and 20% test in a stratified way using scikit-learn package (Pedregosa et al., 2011) with 42 as random state. Regarding multi-label sub-task B where the data is unbalanced, the data is split into 64% train, 16% validation and 20% test using the "iterative_train_test_split" method from scikit-multilearn package (Szymański and Kajdanowicz, 2018) to equally represent the different combination of overlapping labels in the splits.

To obtain the best results and avoid overfitting, we optimized several hyperparameters. Table 1 summarizes the hyperparameters tuned for both sub-tasks using their respective development sets. The experiment tracking and the selected hyperparameter values are published in Weight and Biases[5][6]. The resulting model are openly available in HuggingFace[7].

5 Results

5.1 Sub-task A - Binary misogyny classification

5.1.1 Image

Firstly, we analysed which of the three preprocessing techniques performed best for this task, where we observed that the images without preprocessing showed the best results. Therefore, all models were trained on this dataset. Table 2 shows performance from the five models, where the best model is `EfficientNetB0` which achieves the highest F1 score.

Image Model	F1 macro Avg
VGG16	0.5224
ResNet	0.6143
DenseNet	0.6608
EfficientNet	**0.6825**
Inception	0.6792

Table 2: Evaluation results of image models in the validation split of subtask A.

5.1.2 Text

The Trasnforme-based models results for validation split are shown in Table 3. As can be derived from these results, `bertweet-base` model has the best score and it is the one selected for the next multimodal late fusion step.

5.1.3 Multimodal Late Fusion

As explained in 3.4 different late fusion methods are explored. The best score is obtained using Auto-sklearn and probabilities from both text and image models as input data (see Table 4). The best Auto-sklearn ensemble configuration is composed

[5]Tracking experiments W&B sub-task A
[6]Tracking experiments W&B sub-task B
[7]Multi-label sub-task B model in Hugging Face hub

Text Model	F1 macro Avg
bertweet-base	**0.8320**
all-miniLM-L6-v2	0.8254
all-distilroberta-v1	0.8239
twitter-roberta-base-offensive	0.8082
twitter-xlm-roberta-base	0.7950

Table 3: Evaluation results of Transformer-based models in the validation split of subtask A.

Late Fusion Method	F1 macro Avg
Avg Logit	0.7874
Avg Probs	0.8410
Max Logit	0.8425
Max Probs	0.8410
Weighted Avg Logit	0.8307
Weighted Avg Probs	0.8430
Auto-sklearn Logit	0.8400
Auto-sklearn Probs	**0.8430**
Auto-sklearn Embs	0.7890

Table 4: Evaluation results of multimodal late fusion methods in the validation split of subtask A.

of three SGD classifiers [8].

These scores presented are remarkable, as logits contain more information about the model's decisions, but the concatenation of probabilities as late fusion input proves to be more useful for sub-task A. This might be explained by the fact that logits from different models have different distributions, not being as useful as normalized inputs.

5.2 Sub-task B - Multi-label misogyny classification

5.2.1 Image

In multilabel task the preprocessing is interesting. The third technique, inpainting the text from the images has better performance than the no preprocessing, however, the difference between them is low (as f1 score is 0.02) so it was decided to continue with the non-preprocessed images in order to maintain the methodology of the sub-task A. In this sub-task the best model is `EfficientNetB0`, as in the first one, which reached the highest performance.

[8]https://github.com/AIDA-UPM/AIDA-UPM-SemEval-2022-Task-5-MAMI-

Image Model	F1 macro Avg
VGG16	0.2626
ResNet	0.27734
DenseNet	0,2857
EfficientNet	**0.3477**

Table 5: Evaluation results of image models in the validation split of subtask B.

5.2.2 Text

As in sub-task A, `bertweet-base` model has the best score and it is the one selected for the next multimodal late fusion step (see Table 6).

Text Model	F1 macro Avg
bertweet-base	**0.5785**
all-distilroberta-v1	0.5570
twitter-roberta-base-offensive	0.4666
twitter-xlm-roberta-base	0.4218
all-miniLM-L6-v2	0.2057

Table 6: Evaluation results of Transformer-based models in the validation split of subtask B.

5.2.3 Multimodal Late Fusion

As in sub-task A, Auto-sklearn and probabilities from both text and image models as input data (see Table 7) shows the best results. The Auto-sklearn ensemble configuration is composed of Random Forest MLP, and Naive Bayes classifiers.

It is interesting to note that in a more in-depth analysis of the classification results performed by the different fusion methods, the simplest (e.g., average, max) only learned to correctly separate the majority label (misogynous or non-misogynous). However, the models using Auto-sklearn did manage to also classify the less frequent labels.

Finally, we report our test competition results along with the baseline results from the organizers of the competition. For sub-task A, the baselines are grounded a fine-tuned sentence embedding using the USE pre-trained model; fine-tuned image classification model grounded on `VGG-16`; and a concatenation of deep image and text representations using a single layer neural network. For sub-task B, the baselines are grounded on a multi-label model, based on the concatenation of deep image and text representations; a hierarchical multi-label model, based on text representations, for predicting if a meme is misogynous or not and, if misogynous, the corresponding type.

Late Fusion Method	F1 macro Avg
Avg Logit	0.4475
Avg Probs	0.2014
Max Logit	0.4289
Max Probs	0.3308
Weighted Avg Logit	0.4977
Weighted Avg Probs	0.1627
Auto-sklearn Logit	0.5411
Auto-sklearn Probs	**0.5522**
Auto-sklearn Embs	0.3897

Table 7: Evaluation results of multimodal late fusion methods in the validation split of subtask B.

Subtask	Method	F1 macro Avg
A	Our Late Fusion method	0.636
	Baseline_Text	0.640
	Baseline_Image	0.639
	Baseline_Image_Text	0.543
B	Our Late Fusion method	0.632
	Baseline_Hierarchical_M.	0.621
	Baseline_Flat_Multilabel	0.421
	Baseline_Image_Text	0.000
	Baseline_Text	0.000
	Baseline_Image	0.000

Table 8: Competition results

6 Conclusion

As a conclusion of the results obtained in the exploration and evaluation of models for the development of the multimodal late information fusion architecture, it is evident that the contribution between image and text is different, being the text much more informative in both sub-tasks.

In the case of the image model, it is interesting to note that the different proposed pre-processing techniques do not seem to have a beneficial effect on model training. Although further future analysis is needed, one possible justification could be that the information supplied by the text present in the images provides valuable information rather than noise in the CNN models.

Finally, it is also important to point out that in sub-task A, the multimodal strategy is not as relevant as expected as the baseline strategy that combines image and text information and our approach has the lowest scores in Table 8. A possible explanation for the results obtained could be the difference distribution between train and test sets of the competition, which would have facilitated overfitting in the development of the models in sub-task A. On the contrary, our method proves to be beneficial in sub-task B. Therefore, this could reinforce the idea of overfitting in sub-task A since sub-task B is more complex and better results are obtained. Following the same line, obtaining text models in sub-task B that maintain sub-task A performance supports future research exploring pruning techniques to avoid this situation. Additionally, this study provides a springboard for exploring the late fusion methods applied in this work in different modal problems and other domain scenarios, and comparing them to an end-to-end deep classifier.

In general, our results from the presented multimodal late fusion approach are encouraging to counteract malicious misogynistic behavior against women on social media.

Acknowledgments

This work has been supported by the research project CIVIC: Intelligent characterisation of the veracity of the information related to COVID-19, granted by BBVA FOUNDATION GRANTS FOR SCIENTIFIC RESEARCH TEAMS SARS-CoV-2 and COVID-19, by the Spanish Ministry of Science and Innovation under FightDIS (PID2020-117263GB-100) and XAI-Disinfodemics (PLEC2021-007681) grants, by Comunidad Autónoma de Madrid under S2018/TCS-4566 grant, by European Comission under IBERIFIER - Iberian Digital Media Research and Fact-Checking Hub (2020-EU-IA-0252), by ”Convenio Plurianual with the Universidad Politécnica de Madrid in the actuation line of *Programa de Excelencia para el Profesorado Universitario*” and by the research project DisTrack: Tracking disinformation in Online Social Networks through Deep Natural Language Processing, granted by Barcelona Mobile World Capital Foundation.

References

Edwin Aldana-Bobadilla, Alejandro Molina-Villegas, Yuridia Montelongo-Padilla, Ivan Lopez-Arevalo, and Oscar S Sordia. 2021. A language model for misogyny detection in latin american spanish driven by multisource feature extraction and transformers. *Applied Sciences*, 11(21):10467.

Aayush Aman, Gopal Krishna, Tushar Anand, and Anubhaw Lal. 2021. Identification of offensive content in memes. In *Data Science and Security*, pages 438–445. Springer.

Francesco Barbieri, Luis Espinosa Anke, and Jose Camacho-Collados. 2021. Xlm-t: A multilingual language model toolkit for twitter.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Gaia Campisi, Silvia Corchs, Elisabetta Fersini, Francesca Gasparini, and Monica Mantovani. 2018. Automatic detection of sexist content in memes. *Image*, 46:53–9.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Elisabetta Fersini, Francesca Gasparini, and Silvia Corchs. 2019. Detecting sexist meme on the web: A study on textual and visual cues. In *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*, pages 226–231. IEEE.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. 2020. Auto-sklearn 2.0: Hands-free automl via meta-learning. *arXiv:2007.04074 [cs.LG]*.

Matthias Feurer, Aaron Klein, Jost Eggensperger, Katharina Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems 28 (2015)*, pages 2962–2970.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1470–1478.

Xiaoyu Guo, Jing Ma, and Arkaitz Zubiaga. 2020. Nuaa-qmul at semeval-2020 task 8: Utilizing bert and densenet for internet meme emotion analysis. *arXiv preprint arXiv:2011.02788*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Eftekhar Hossain, Omar Sharif, and Mohammed Moshiul Hoque. 2021. Nlp-cuet@ dravidianlangtech-eacl2021: Investigating visual and textual features to identify trolls from multimodal social media memes. *arXiv preprint arXiv:2103.00466*.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Javier Huertas-Tato, Alejandro Martín, Julian Fierrez, and David Camacho. 2022. Fusing cnns and statistical indicators to improve image classification. *Information Fusion*, 79:174–187.

Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, and Helen Yannakoudakis. 2020. A multimodal framework for the detection of hateful memes. *arXiv preprint arXiv:2012.12871*.

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

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

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. *arXiv preprint arXiv:1910.02334*.

Niloofar Safi Samghabadi, Parth Patwa, Srinivas Pykl, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and misogyny detection using bert: A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Elena Shushkevich and John Cardiff. 2019. Automatic misogyny detection in social media: A survey. *Computación y Sistemas*, 23(4):1159–1164.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Robyn Speer. 2019. ftfy. Zenodo. Version 5.5.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Piotr Szymański and Tomasz Kajdanowicz. 2018. A scikit-based python environment for performing multi-label classification.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

George-Alexandru Vlad, George-Eduard Zaharia, Dumitru-Clementin Cercel, and Mihai Dascalu. 2020. Upb@ dankmemes: Italian memes analysis-employing visual models and graph convolutional networks for meme identification and hate speech detection. *EVALITA Evaluation of NLP and Speech Tools for Italian-December 17th, 2020*, page 288.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

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

Haris Bin Zia, Ignacio Castro, and Gareth Tyson. 2021. Racist or sexist meme? classifying memes beyond hateful. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 215–219.

YMAI at SemEval-2022 Task 5: Detecting Misogyny in Memes using VisualBERT and MMBT MultiModal Pre-trained Models

Mohammad Habash **Yahya Daqour** **Malak Abdullah** **Mahmoud Al-Ayyoub**
Computer Science Department
Jordan University of Science and Technology
Irbid, Jordan
{mshabash187,yidaqour18}@cit.just.edu.jo, {mabdullah,maalshbool}@just.edu.jo

Abstract

This paper presents a deep learning system that contends at SemEval-2022 Task 5. The goal is to detect the existence of misogynous memes in sub-task A. At the same time, the advanced multi-label sub-task B categorizes the misogyny of misogynous memes into one of four types: stereotype, shaming, objectification, and violence. The Ensemble technique has been used for three multi-modal deep learning models: two MMBT models and VisualBERT. Our proposed system ranked 17th place out of 83 participant teams with an F1-score of 0.722 in sub-task A, which shows a significant performance improvement over the baseline model's F1-score of 0.65.

1 Introduction

Participating in online social networks (OSN) has become more common nowadays, especially for women, as 78 percent of them use social media several times per day compared to 65 percent of the men. Anyhow, hateful speech against women did not stop until offline methods but also found its way on the web using popular communication tools such as memes. A meme is an online spread of captioned pictures or GIFs meant to be funny or critical to people or society. It has the most humor characteristics on social network platforms. However, in a short period, some minorities started to develop memes aiming at gender discrimination, inequality, and spreading hate against women.

Due to the growth of Artificial Intelligence (AI), Natural Language Processing (NLP), and Computer Vision (CV), the machine started to recognize and understand languages and images more than ever. Therefore, preventing hate speech and misogyny became more achievable by applying these methodologies. This paperwork presents our participation in SemEval-2022 Task 5 (Elisabetta Fersini, 2022) which aims to detect misogyny in memes. In this study, MultiModal BiTransformers (MMBT) (Kiela et al., 2019) and VisualBERT (Li et al., 2019) have been used to build the proposed model by employing the ensemble technique. The proposed model reaches an F1-score of 0.722 in sub-task A depending on the provided 10K memes dataset and without external datasets.

This paperwork is constructed as follows: Section 2 presents the related work, followed by Section 3, which clarifies the task description. Section 4 shows dataset details and preprocessing procedures. Section 5 shows off the architecture of the solution system. Finally, sections 6, 7, and 8 provide the experiments, results, and conclusion.

2 Related Work

Several researchers are attracted to detecting hate, harm, sarcasm, and satire on Social media (Faraj and Abdullah, 2021; Isaksen and Gambäck, 2020; Watanabe et al., 2018). Internet Memes are considered one of the most popular ways to communicate on all topics on social media. Since it is a global issue and a world trend, many researchers have started to compete in preventing harmful memes from pervasion. This paperwork (Shang et al., 2021) aimed to detect offensive analogy memes to prevent this from spreading out. The focus was here on images as it attracts more people and has richer information than the text alone.

One competition from Facebook (Kiela et al., 2021) aimed to detect hate speech in memes as it is hard to be tackled by having humans checking out every meme. From their vision, hate speech was understood as "any communication that disparages a target group of people based on some characteristic such as race, color, ethnicity, gender, sexual orientation, nationality, religion, or other characteristics" (Levy et al., 2000). The dataset consisted of

12540 memes, split between 7834 for non-hate and 4706 for harmful.

On the other hand, deep learning models have been successfully applied to learning features for single modalities: text (Abdullah and Shaikh, 2018; Abedalla et al., 2019) and images (Abedalla et al., 2021, 2020). Moreover, learning-based models that combine elements from multiple modalities are more robust in visual-linguistic tasks such as detecting hateful memes (Sandulescu, 2020).

3 Task Description

The main objective of this task (Elisabetta Fersini, 2022) is the identification of misogynous memes. Memes can be identified using text, images, or both. Using both images and text surely produces better results. The task is divided into two sub-tasks:

- Sub-task: This task aims to classify the meme as either misogynous or not misogynous.
- Sub-task B: this is a multi-label task, and the goal is to identify the type(s) of misogyny in misogynous memes. Types of misogyny are stereotypes, shaming, objectification, and violence.

Although we did not use any external datasets, it is allowed for participants to do so.

4 Dataset

4.1 Dataset Description

The training dataset provided by (Elisabetta Fersini, 2022) consists of 10,000 memes along with a file containing the extracted text, image file names, and five columns representing labels. The evaluation dataset consists of 1,000 memes, provides the extracted text and does not contain the labels. Competitors are supposed to use the evaluation dataset to produce and submit evaluation results. Memes were gathered by searching on social media platforms, such as Twitter and Reddit, and other meme creation and sharing websites, such as 9GAG, Knowyourmeme, and Imgur.

4.2 Data Preprocessing

Regarding the extracted text of memes, it is considered clean, and there is not much pre-processing required, but a lot of memes contain a web address in the bottom-right corner, which might better be removed. Also, Twitter hashtags and usernames are removed. Furthermore, any unwanted non-English or non-understandable characters are removed. The previously mentioned pre-processing steps provide a slight performance improvement. Regarding the images, no pre-processing is applied.

Statistical Property	Value
Maximum sentence length	325
Minimum. sentence length	2
Average sentence length	21.46
Standard deviation of sentence length	16.97

Table 1: Statistical properties of memes text

4.3 Data Analysis

The dataset contains 5,000 misogynous memes and 5,000 non-misogynous memes. Thus, the dataset is perfectly balanced for sub-task (A). As shown in Table 1, the longest sequence of memes text is 325 words long, and the shortest sequence is only two words long. The average sentence length is 21.46, and the standard deviation of sentence lengths is 16.97. The previous statistics indicate that most sequences are nearly 2-38 words long, and fewer sentences contain more than 38 words. Therefore, the chosen threshold for sequence length (maximum sequence length) is 64.

5 Systems Description

5.1 Text-based Model

The first approach uses pre-trained language models to identify misogynous memes based on text. We ended up choosing RoBERTa (Liu et al., 2019) as it is one of the state-of-the-art language models. Unfortunately, relying on text-only is not good enough to classify memes robustly.

5.2 Image-based Model

The second approach uses pre-trained computer vision models to identify misogynous memes based on images. The chosen model is VGG (Simonyan and Zisserman, 2014), it is one of the most popular Convolutional Neural Network (CNN) models, and the chosen variant is VGG-16 which consists of 16 layers. Unfortunately, similar to text, using images only does not classify memes robustly.

5.3 Multimodal Models

As memes contain both text and images, it is reasonable to use deep learning models that use both text and images to produce more accurate and robust predictions.

The first model used in our solution is Multimodal Bi-Transformers (MMBT) (Kiela et al., 2019). MMBT was developed based on the transformer architecture. It uses the attention modules of the transformer to combine embeddings of different modalities (text and image in our case). The key approach used in MMBT is to take the image as an input, extract its features using a CNN model, concatenate the generated features with the text input tokens used in BERT (Devlin et al., 2018), and feed the model with the text and image tokens. We used ResNet-152 (He et al., 2016) as the CNN model for feature extraction (image encoding) in MMBT.

The second model in our solution is VisualBERT (Li et al., 2019) which is built to fulfill a wide range of vision and language tasks. It consists of a stack of transformer layers similar to BERT architecture to prepare embeddings for image-text pairs. BERT tokenizer is used as a text encoder. For images, a custom pre-trained object detector must be used to extract regions and bounding boxes fed to the model as visual embeddings. We used Detectron2 (Wu et al., 2019) to generate the visual embeddings using MaskRCNN+ResNet-101+FPN model checkpoint.

As shown in Figure 1, The final model is constructed using the voting technique between two MMBT models and the VisualBERT model. Both MMBT models use ResNet-152 for image encoding, but they differ slightly as one uses BERT-base-uncased for the text while the other uses BERT-large-uncased. It is worth mentioning that the two MMBT models were trained using different random seeds.

6 Experiments

Regarding sub-task A, we have experimented with RoBERTa as it is one of the state-of-the-art language models. We have used Huggingface (Wolf et al., 2019) and Pytorch (Paszke et al., 2019) to implement roberta-base. We chose a maximum sequence length of 64. Following this, we have experimented with VGG-16, which is one of the most popular pre-trained CNN models, and implemented it using Keras (Chollet, 2015). We set the model to expect images of size 300x300 and added a dropout layer with a drop rate of 0.3, followed by a dense layer. For RoBERTa and VGG-16, we split the provided dataset into training and validation datasets with a validation split value of 0.1. Both RoBERTa and VGG-16 performed poorly, as memes classification tasks require models that understand images and text.

Model	RoBERTa	VGG-16	MMBT	VisualBERT
Epochs	2 5	50	5 10	5 10
Batch Size	32 64	32 64	16 32 64	32 64
Learning Rate	2e-5 5e-5	0.001	1e-5 2e-5 5e-5	3e-5 5e-5
Weight Decay	0	0	0.0001 0.001	0.001
Gradient Acc. Steps	1	1	1 2	1
Random Seed	17	17	1337 17	42

Table 2: Hyper-parameters of all models for sub-task A.

Subsequently, we have experimented with two vision+language models: MMBT and VisualBERT. Both of them were implemented using Huggingface and Pytorch. MMBT was experimented with using ResNet-152 as image encoder and BERT as text encoder. Two BERT variants were used: BERT-base-uncased and BERT-base-large. Visual embeddings for visualBERT were generated using MaskRCNN+ResNet-101+FPN model checkpoint from Detectron2. For MMBT and VisualBERT, we used k-fold cross-validation with a k value of 5. In other words, the training and validation datasets consist of 8000 and 2000 memes, respectively, and after each epoch, the validation dataset switches to a completely different 2000 memes. This allows the models to train on the entire dataset eventually. In our experiments, MMBT outperformed VisualBERT. AdamW optimizer was used for all models. Table 2 shows different sets of hyper-parameters used for all previously mentioned models.

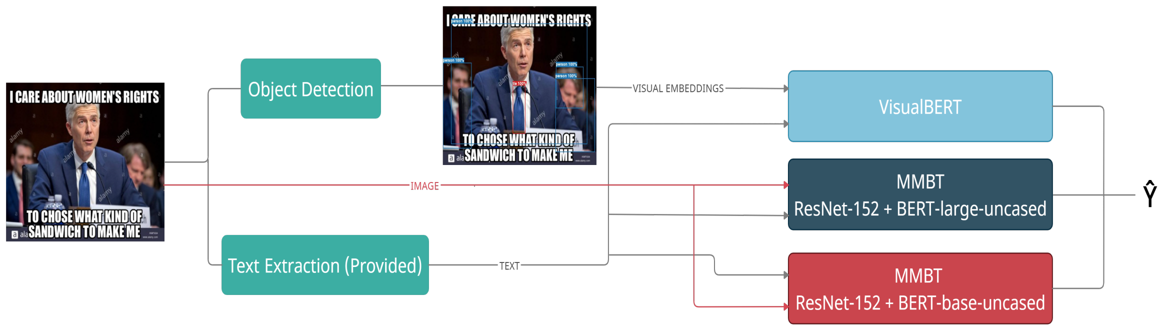

Object Detection
VISUAL EMBEDDINGS
VisualBERT
IMAGE
MMBT
ResNet-152 + BERT-large-uncased
Ŷ
Text Extraction (Provided)
TEXT
MMBT
ResNet-152 + BERT-base-uncased

Figure 1: The architecture of the final system for sub-task A. ŷ represents the final prediction after applying voting technique between models. The pipeline also illustrates some pre-processing steps before modelling.

7 Results

The final results for sub-task A are produced using the ensemble technique between two MMBT models trained using different random seeds and VisualBERT. RoBERTa and VGG-16 were not included in the final system as they performed poorly compared to MMBT and VisualBERT. As shown in Table 3, both MMBT models slightly outperformed VisualBERT, but the latter still has a positive effect on the final result after the ensemble.

Looking at Figure 2, which shows the confusion matrix for the final results in Sub-task A, We can see that the final system succeeded in predicting 359 non-misogynous memes and 363 misogynous memes. The system also failed to identify 137 misogynous memes and mistakenly identified 141 non-misogynous memes as misogynous.

Model	MMBT (ResNet + Bert-base)	MMBT (ResNet + Bert-large)	VisualBERT	Ensemble
Epochs	5	5	5	*
Batch Size	32	32	32	*
Learning Rate	1e-5	1e-5	5e-5	*
Weight Decay	0	0	0.001	*
Gradient Acc. Steps	1	1	1	*
Random Seed	1337	17	42	*
F1-Score	0.695	0.697	0.679	0.722

Table 3: Final experiments used in sub-task A. The final F1-score of 0.722 is obtained by using ensemble technique between the three models.

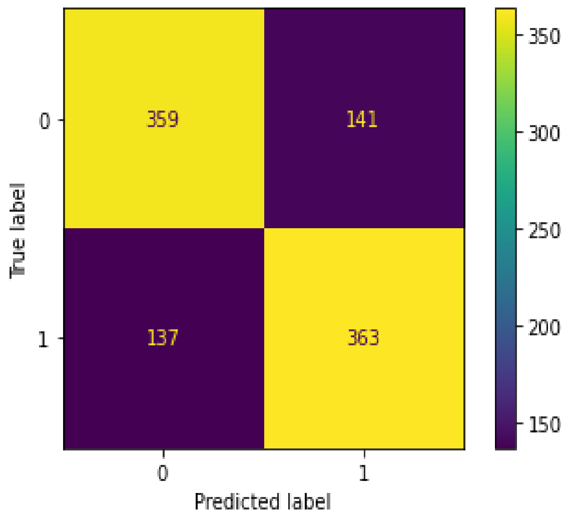


Figure 2: Confusion matrix for the final results in sub-task A

8 Conclusion

This paper presented our deep learning system that contended at SemEval-2022 Task 5. We experimented with different deep learning models, starting with the RoBERTa language model and the VGG-16 CNN model. Subsequently, we experimented with vision and language models, such as MMBT and VisualBERT, which significantly outperformed VGG-16 and RoBERTa. Using ensemble technique between two MMBT models and VisualBERT produced an F1-score of 0.7222 which led to ranking 17th place in sub-task A.

References

Malak Abdullah and Samira Shaikh. 2018. Teamuncc at semeval-2018 task 1: Emotion detection in english and arabic tweets using deep learning. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 350–357.

Ayat Abedalla, Malak Abdullah, Mahmoud Al-Ayyoub, and Elhadj Benkhelifa. 2020. 2st-unet: 2-stage training model using u-net for pneumothorax segmentation in chest x-rays. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE.

Ayat Abedalla, Malak Abdullah, Mahmoud Al-Ayyoub, and Elhadj Benkhelifa. 2021. Chest x-ray pneumothorax segmentation using u-net with efficientnet and resnet architectures. *PeerJ Computer Science*, 7:e607.

Ayat Abedalla, Aisha Al-Sadi, and Malak Abdullah. 2019. A closer look at fake news detection: A deep learning perspective. In *Proceedings of the 2019 3rd International Conference on Advances in Artificial Intelligence*, pages 24–28.

François Chollet. 2015. keras. https://github.com/fchollet/keras.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Giulia Rizzi Aurora Saibene Berta Chulvi Paolo Rosso Alyssa Lees Jeffrey Sorensen Elisabetta Fersini, Francesca Gasparini. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Dalya Faraj and Malak Abdullah. 2021. Sarcasmdet at semeval-2021 task 7: Detect humor and offensive based on demographic factors using roberta pretrained model. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 527–533.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Vebjørn Isaksen and Björn Gambäck. 2020. Using transfer-based language models to detect hateful and offensive language online. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 16–27.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, Ethan Perez, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *arXiv preprint arXiv:1909.02950*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Casey A Fitzpatrick, Peter Bull, Greg Lipstein, Tony Nelli, Ron Zhu, et al. 2021. The hateful memes challenge: competition report. In *NeurIPS 2020 Competition and Demonstration Track*, pages 344–360. PMLR.

Leonard Williams Levy, Leonard W Levy, Kenneth L Karst, and Adam Winkler. 2000. Encyclopedia of the american constitution.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Vlad Sandulescu. 2020. Detecting hateful memes using a multimodal deep ensemble. *arXiv preprint arXiv:2012.13235*.

Lanyu Shang, Yang Zhang, Yuheng Zha, Yingxi Chen, Christina Youn, and Dong Wang. 2021. Aomd: An analogy-aware approach to offensive meme detection on social media. *Information Processing & Management*, 58(5):102664.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Hajime Watanabe, Mondher Bouazizi, and Tomoaki Ohtsuki. 2018. Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. *IEEE access*, 6:13825–13835.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. https://github.com/facebookresearch/detectron2.

UIC-NLP at SemEval-2022 Task 5: Exploring Contrastive Learning for Multimodal Detection of Misogynistic Memes

Charic Farinango Cuervo and **Natalie Parde**
Natural Language Processing Laboratory
Department of Computer Science
University of Illinois at Chicago
{cfarin2, parde}@uic.edu

Abstract

Misogynistic memes are rampant on social media, and often convey their messages using multimodal signals (e.g., images paired with derogatory text or captions). However, to date very few multimodal systems have been leveraged for the detection of misogynistic memes. Recently, researchers have turned to contrastive learning solutions for a variety of problems. Most notably, OpenAI's CLIP model has served as an innovative solution for a variety of multimodal tasks. In this work, we experiment with contrastive learning to address the detection of misogynistic memes within the context of SemEval-2022 Task 5. Although our model does not achieve top results, these experiments provide important exploratory findings for this task. We conduct a detailed error analysis, revealing promising clues and offering a foundation for follow-up work.

1 Introduction

Hateful expressions on the Internet are widespread and mostly based on religion, gender, race, or physical attributes (Lippe et al., 2020). Such language exacerbates damaging societal problems such as racism, sexism, and other types of discrimination. In particular, misogynistic abuse has become very prevalent and poses a serious problem in cyberspace (Citron, 2014). Although extensive research on hate speech and misogyny has been conducted (Kumar et al., 2020), it has thus far centered on the analysis of text or images alone.

Memes, or social media images that communicate messages through the creative use of imagery understood to carry specific rhetorical value among members of a community, are a common platform for misogyny and hateful expressions. Approximately 78% of women use image-based social media multiple times per day (compared to 65% of men) (Fersini et al., 2022), making exposure to this harmful content alarmingly common. Classifying memes is challenging because of their multimodal interplay between image and text, as well as their region-specific interpretation, and existing multimodal approaches do not perform very well in the classification of hateful memes (Kiela et al., 2020). The underlying goal of SemEval-2022 Task 5 was to address this research gap, tackling the challenge of identifying misogynistic memes using multimodal data by inviting researchers to experiment with a variety of approaches.

Our team, *UIC-NLP*, investigated an adapted version of the Contrastive Language-Image Pre-training (CLIP) technique recently established and applied with success to numerous other tasks (Radford et al., 2021; Conde and Turgutlu, 2021; Galatolo et al., 2021). Our model, recorded on the leaderboard in the 6th leaderboard cluster under the OpenReview username "Charicfc," ranked 71st out of 83 participants and obtained an average F_1 score of 0.62. We analyze our model's predictions and offer insights and recommendations for improving upon it in the future.

2 Background

2.1 SemEval-2022 Task 5 (MAMI)

SemEval-2022 Task 5 was created to address the rise in the use of memes as a form of hate against women, which contributes to sexual stereotyping and gender inequality. The data used for this challenge is comprised of English-language memes collected from the web and manually annotated via crowdsourcing platforms. Each data sample has an image, its raw text in English (*transcript*), a binary annotation indicating the presence of misogyny, and (if applicable) the type of misogyny (shaming, objectification, stereotype, or violence). Our model addresses *Subtask 1*, which focuses on binary classification of memes as misogynist or not misogynist. The output for a given sample is a confidence score for the predicted class. Although no approaches have sought to perform multimodal detection of misogynistic memes to date, we review work on classifying misogynistic expressions in text and multimodal classification of hateful memes in the following sections.

2.2 Identifying Misogyny in Text

Previous approaches in the related IberEval 2018 Automatic Misogyny Identification (AMI) task for misogyny detection in tweets (Fersini et al., 2018) leveraged statistical classification models, including variations of Support Vector Machines (SVM) (Nina-Alcocer, 2018; Pamungkas et al., 2018). Other recent work towards identifying misogyny in text has leveraged CNNs, LSTMs, and combined representations from models like BERT (Basile et al., 2019; Parikh et al., 2021). Recently, the Evalita 2020 AMI challenge best results were obtained by ensembles of fine-tuned BERT models (Lees et al., 2020).

2.3 Multimodal Classification of Hateful Memes

Multimodal learning has recently gained attention due to the poor performance of existing (unimodal) models on multimodal tasks (Lippe et al., 2020), with most recent solutions (context aware) employing neural architectures such as CNNs, RNNs, and Transformer-based attention models like BERT (Afridi et al., 2020; Modi and Parde, 2019; Parde, 2020). Although existing work on hate speech detection has largely relied on text-based features, this has gradually started to shift with the introduction of multimodal datasets (Lippe et al., 2020). In general, the focus has been shifted to BERT-based models (Afridi et al., 2020). In Facebook AI's Hateful Memes Challenge (Kiela et al., 2021) the top two models involved an ensemble of four Transformer-based models (Zhu, 2020; Muennighoff. 2020). The third and fourth place used fine-tuned VisualBERT (Velioglu and Rose, 2020) and UNITERT (Hossain et al., 2021) models. Aggarwal et al. (2021) extracted image features with a pretrained ResNet-152 model while passing the text data through Facebook's FastText encoder (Bojanowski et al., 2017), concatenating both feature vectors and passing them to a fully connected layer for classification. Due to its novelty, few solutions to date have explored the use of CLIP in challenging scenarios such as misogynistic memes classification. For this work, we trained a version of CLIP resembling Shariatnia's (2021).

3 System Overview

Our system architecture is a variation of OpenAI's CLIP model (Radford et al., 2021). We refer readers to the original paper for a detailed overview and illustrations of the architecture, and summarize it here. Inspired by the idea of usability and generality, CLIP uses a contrastive learning objective to build a joint visual-linguistic space for learning visual concepts from natural language supervision. We describe the various components of our architecture below.

3.1 Encoders

ResNet-50: OpenAI's best CLIP performance was achieved using a pretrained encoder which the authors called ViT-L/14@336px. In our case, we experimented with several pretrained image encoders. Although Shariatnia (2021) used ResNet-50 by default, we also considered ViT-B/16@224px, ViT-L/16@224px, and ViT-L/16@384px (Dosovitskiy et al, 2020). Nonetheless, we empirically determined that ResNet-50 (He, 2016), a deep CNN trained on more than a million images from the ImageNet database with an objective of classifying images into 1000 categories, yielded the best performance.

DistilBERT: OpenAI's CLIP used a modified Transformer to encode text. We instead used a lighter version of BERT (Devlin et al., 2018) called DistilBERT (Sanh et al, 2019) which Shariatnia (2021) also uses. BERT is a large Transformer-based language model that has achieved strong performance in many NLP tasks, and DistilBERT uses a process known as knowledge distillation to reproduce its behavior by training a smaller model to replicate its probability distributions across class predictions.

3.2 Learning Objective

Radford et al. (2021) introduced contrastive objectives as a mechanism for learning multimodal representations from raw images and paired descriptions. In essence, the contrastive objective seeks to learn a multimodal embedding space where image embeddings and text embeddings are

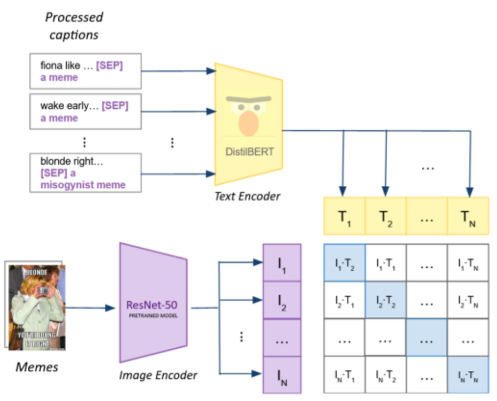

Processed captions
fiona like ... [SEP] a meme
wake early... [SEP] a meme
blonde right... [SEP] a misogynist meme
DistilBERT
Text Encoder
T_1 T_2 ... T_N
Memes
ResNet-50
Image Encoder
I_1 I_2 ... I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ ... $I_2 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ ... $I_N \cdot T_N$

Figure 1. Learning stage. Adapted from Radford et al., 2021.

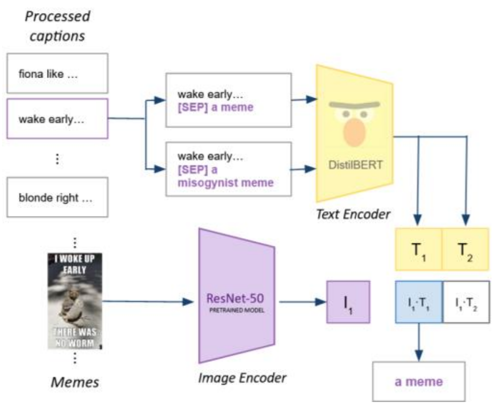


Figure 2. Evaluation stage. Adapted from Radford et al., 2021.

mapped to the same point if they describe the same thing, and different points otherwise. Cosine similarity is used to measure the distance between embeddings. Figure 1 shows the "logits" matrix obtained after applying the dot product between images and text embeddings. Each cell in the matrix (*logits*) is a measure of similarity between an image and a text caption in the dataset (N^2 pairs). It is expected that cells along the diagonal (N pairs), which contain the similarity between an image and its actual text, are maximized. Simultaneously, the $N^2 - N$ incorrect pairs should be minimized.

The *targets* for the images and texts where the similarities are maximum are obtained by computing dot products between embedding matrices. These are averaged and passed through a SoftMax, with the result being a *target* matrix where the diagonal is close to 1.0 and the other pairs are close to 0.0. The loss for images and texts is obtained by calculating the cross-entropy between the target and the logits matrix.

3.3 Training Procedures

The target output for the original CLIP model was the correct caption for an image. For example, one could input an image of a cat along with the captions: {"Image depicting a cat," "Image depicting a dog"} with the expectation that it would return the correct caption. We refer readers to the original paper for further implementation details.

We slightly modified this approach in our own model, such that the training text instead was the language content from the meme followed by the correct label. We separated the text and label using the [SEP] token as defined for BERT's next-sentence prediction objective. Therefore, we concatenated each instance with the following sentences depending on the example's class:

1. For class MISOGYNY: <text_1> + "[SEP] a misogynist meme"
2. For class NOT MISOGYNY: <text_1> + "[SEP] a meme"

When evaluating new samples using this architecture, each instance must be a paired image and text caption. Thus, we created two versions of each text caption: one for class MISOGYNY, and one for class NOT MISOGYNY. Figure 2 shows the procedure. The predicted label for the test instance was the one for which the model made its prediction with higher confidence.

4 Experimental Setup

4.1 Exploratory Analysis

Prior to conducting our experiments, we performed preliminary descriptive analyses on the training data, comprising 10,000 memes (image + paired text content) with a total vocabulary of 12,611 tokens. From these, 6,649 tokens only appear once. Figure 3 shows a word cloud for the 100 most frequent words in the corpus. Interestingly, we determined that the vocabulary for non-misogynist memes is much richer, including 9,285 compared to 7,348 unique words. We also observed that while the word "woman" is repeated 1,416 times in the misogyny class, it is repeated only 528 times in the not misogyny class.

Aiming to find greater insights we computed the Pointwise Mutual Information (PMI) score for all words given each class. PMI is a feature scoring metric that can be used to estimate the association between a feature and a class (it has also traditionally been used to identify collocations in text). A close association indicates which features

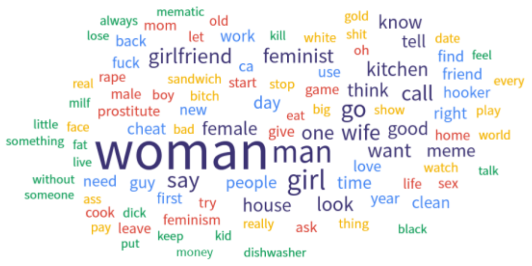


Figure 3: Word cloud of the 100 most frequent words in the corpus. *Warning: This image includes language that may be offensive or upsetting.*

(words) are more important for a class. PMI is computed using the following formula:

$$PMI(w,c) = \log\frac{p(w,c)}{p(w).p(c)} = \log\frac{p(w|c)}{p(w)} = \log\frac{p(c|w)}{p(c)} \qquad (1)$$

Table 1 shows the 15 words with the highest PMI for each class. As shown, despite its frequency, the word "woman" carries less value when measured via PMI. Some swear words are classified as important for the MISOGYNY class, but not the NOT MISOGYNY class.

4.2 Preprocessing

A large challenge in this task was that since the text content from memes was extracted with an OCR tool, it contained substantial noise. Therefore, we created the following preprocessing pipeline that we applied to all texts before training and evaluating:

1. **Case Normalization:** All text was normalized by converting it to lowercase.
2. **Part of Speech (POS) Tagging:** POS tags were assigned to each word.
3. **Proper Name Removal:** Regular expressions were applied to convert some wordforms (e.g., urls) to generic tokens.
4. **Special Token Categorization:** Words belonging to several word sets of interest, including *celebrity names* and *profanity terms*, were kept.
5. **Lemmatization:** Words were converted to their base forms.
6. **Stopword Removal:** Highly frequent words (e.g., "the") were removed.
7. **Special Character Removal:** Non-alphabetical characters (e.g., digits or punctuation) were removed.

Aspects of this preprocessing pipeline were similar to those reported by Cardoza (2022) and

Misogynist			Non-Misogynist	
Token	**PMI**		**Token**	**PMI**
dishwasher	0.660357		gold	0.591098
sandwich	0.599215		house	0.587787
rape	0.567609		cheat	0.470893
feminist	0.560566		clean	0.457903
fat	0.479573		call	0.456758
feminism	0.478848		people	0.430466
girl	0.453501		cook	0.407265
bitch	0.425832		game	0.40027
woman	0.403349		kid	0.394994
always	0.291434		girlfriend	0.391106

Table 1: Top 15 PMI scores. *Warning: This table includes language that may be offensive or upsetting.*

Kovács et al. (2020). POS tagging allowed us to target the "proper noun" and "other" tags. By excluding these tags, we resolved many lingering issues following application of regular expressions, such as remaining usernames and gibberish words. Removing specific instances of these terms aided the model in avoiding overfitting to superfluous names or unknown tokens that were irrelevant to the overarching task of recognizing misogyny. Since certain terms removed by our POS filtering may carry importance to the task (e.g., certain celebrity names or swear terms), we also searched for these terms in several predetermined word sets and kept them if and when they were found.

4.3 Experimental Settings

The training set provided by the task organizers was divided into training (80%) and validation (20%) sets of 8000 and 2000 examples respectively. We primarily adapted our hyperparameter settings from Shariatnia (2021), using AdamW (Loshchilov and Hutter, 2017) as our optimizer with an initial learning rate (LR) of 1e-3 and a scheduler to reduce the LR on plateau. The batch size is left at 32 and the maximum number of epochs was set to 4. The model converges quickly, so this early stop helps to control overfitting. The learning rates for the image and text encoders were left at 1e-4 and 1e-5 respectively. All of the texts were tokenized using the DistilBERT base model with a max number of tokens set to 200. We experimented with CLIP's temperature hyperparameter, finding that the best results were achieved with a value of 1e-0.2.

We selected the best epoch and the best hyper-parameters as measured by F_1 score and accuracy. In the test set evaluation, Subtask 1 systems were evaluated using macro-averaged F_1; thus, the final score is the mean of the F_1 for the two classes.

Development			
	Precision	Recall	F1
Non-Misogynist	0.66	0.71	0.68
Misogynist	0.69	0.64	0.66
Accuracy			0.67
Evaluation			
	Precision	Recall	F1
Non-Misogynist	0.73	0.43	0.54
Misogynist	0.60	0.84	0.70
Accuracy			0.64

Table 2: Results on the development (from training data) and evaluation (from test data).

5 Results

During development we used the training data provided by the MAMI task. We divided it into training (90%) and test (10%) sets, and measured performance using F1, accuracy, precision, and recall on the 10% test set. We provide our results on the development data in Table 2. Once the official test set was released (*Evaluation* in Table 2), we computed the same metrics on that set. The macro-averaged F1 as returned by the task organizers was 0.62, with the system ranking 71st in performance.

5.1 Quantitative Analysis

We briefly analyze our best models' results on the test set for Subtask 1. In particular, we observe a fairly low recall for the NOT MISOGYNY class (0.43), indicating that the system may be struggling to capture all members of this likely more diverse class. As further highlighted in the confusion matrix in Figure 4, the model classifies 28.6% of the NOT MISOGYNIST memes as MISOGYNIST. Thus, most errors were false positives.

5.2 Error Analysis

When manually analyzing the misclassified instances, we observed that they were diverse, containing cartoons, animals, people, drawings, and more. Some images contained text that was unrelated to the caption, creating additional noise. Examples of these images are shown in Figure 5. Several recurring themes occurred among false positives and false negatives, which we summarize below:

False Positives: Most images that were incorrectly classified as misogynistic were primarily dominated by one or more people. For images where humans were not present, the text

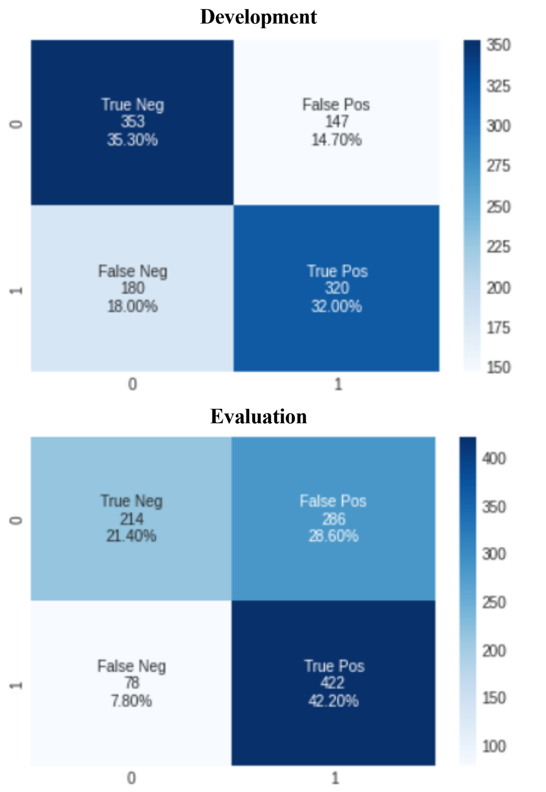


Figure 4: Confusion matrices for the development and evaluation data.

often contained swear words or synonyms for "woman," some of which were offensive although not employed with purposeful misogyny.

False Negatives: Most images that were incorrectly classified as not misogynistic contained cartoons, animals, storyboards or non-explicitly sexist images, although women were occasionally present.

To address these errors, we recommend actions leveraged in prior work for other tasks to improve the performance of future systems. For instance, Lippe et al. (2020) upsampled their dataset as a solution for poor performance on text confounders. Although the dataset they used (Facebook's Hateful Memes) was specially designed to introduce benign confounders, it might also work in this problem. Nozza et al., (2019) discuss biases introduced in the model by a set of identity terms that are frequently associated with the misogynistic class (e.g., "woman"). The authors propose to upsample the dataset with examples that have the identity terms for the alternate class.

6 Conclusion and Future Work

In this paper, we describe our system implementation for SemEval-2022 Task 5. Our

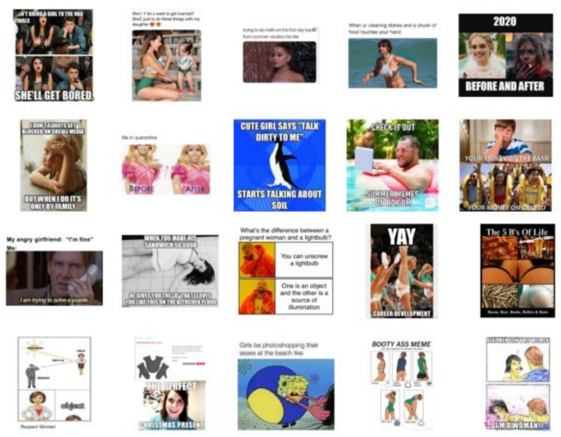


Figure 5: False positives (top 10) and false negatives (bottom 10). *Warning: This image includes content that may be offensive or upsetting.*

model ranked 71st out of 83 participants' teams on Subtask 1. We comprehensively investigate the use of a state-of-the-art multimodal contrastive learning approach for the classification of misogynistic memes. More experiments and tests should be done to improve the model's performance on this task. In particular, upsampling the dataset and addressing the possible biases caused by identity terms should be investigated. Finally, at an architectural level, our current system encodes images as one form of input and encodes paired text content and labels as another form of input, similarly to text encoding strategies used for unimodal sequence prediction tasks. Exploring joint encodings of image and paired text content as a single form of input, with only labels as the other form of input, may be an additional design avenue worth pursuing. Overall, it is our hope that this work motivates additional interest in contrastive learning solutions for multimodal misogynistic meme detection. We make our source code[1] available to other researchers to facilitate follow-up work by others.

7 Acknowledgements

We thank the anonymous reviewers for their comments and suggestions.

References

Tariq Habib Afridi, Aftab Alam, Muhammad Numan Khan, Jawad Khan and Young-Koo Lee. (2020, October). A multimodal memes classification: A survey and open research issues. In *The Proceedings of the Third International Conference on Smart City Applications* (pp. 1451-1466). Springer, Cham. https://doi.org/10.1007/978-3-030-66840-2_109

Apeksha Aggarwal, Vibhav Sharma, Anshul Trivedi, Mayank Yadav, Chirag Agrawal, Dilbag Singh, Vipul Mishra and Hassène Gritli. (2021). Two-way feature extraction using sequential and multimodal approach for hateful meme classification. *Complexity*, *2021*. https://doi.org/10.1155/2021/5510253

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana, Patti, Francisco Manuel Rangel Pardo, Paolo Ross and Manuela Sanguinetti. (2019). Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation* (pp. 54-63). Association for Computational Linguistics. http://dx.doi.org/10.18653/v1/S19-2007

Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov. (2017). Enriching word vectors with subword information. Transactions of the association for computational linguistics, 5, 135-146.

Maria Alejandra Cardoza Ceron. (2022). Using Word Embeddings to Analyze Protests News. arXiv preprint arXiv:2203.05875.

Danielle Keats Citron. (2014). Hate crimes in cyberspace. Harvard University Press.

Marcos V. Conde and Kerem Turgutlu. (2021). CLIP-Art: contrastive pre-training for fine-grained art classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3956-3960).

Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit and Neil Houlsby. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, Jeffrey Sorensen. (2022). SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification. In *Proceedings of the 16th International Workshop*

[1] `https://github.com/charicf/MAMI-CLIP`

on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. (2018). Overview of the Task on Automatic Misogyny Identification at IberEval 2018. *Ibereval@ sepln*, *2150*, 214-228.

Federico A. Galatolo, Mario G.C.A. Cimino and Gigliola Vaglini. (2021). Generating images from caption and vice versa via CLIP-Guided Generative Latent Space Search. arXiv preprint arXiv:2102.01645.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Eftekhar Hossain, Omar Sharif and Mohammed Moshiul Hoque. (2021). NLP-CUET@DravidianLangTech-EACL2021: Investigating Visual and Textual Features to Identify Trolls from Multimodal Social Media Memes. arXiv preprint arXiv:2103.00466.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Casey A. Fitzpatrick, Peter Bull, Greg Lipstein, Tony Nelli, Ron Zhu, Niklas Muennighoff, Riza Velioglu, Jewgeni Rose, Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, Helen Yannakoudakis, Vlad Sandulescu, Umut Ozertem, Patrick Pantel, Lucia Specia, Devi Parikh. (2021, August). The hateful memes challenge: competition report. In *NeurIPS 2020 Competition and Demonstration Track* (pp. 344-360). PMLR.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia and Davide Testuggine. (2020). The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, *33*, 2611-2624.

Ádám Kovács, Judit Ács, Andras Kornai, and Gábor Recski. 2020. Better Together: Modern Methods Plus Traditional Thinking in NP Alignment. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3635–3639, Marseille, France. European Language Resources Association.

Ritesh Kumar, Atul Kr. Ojha, Bornini Lahiri, Marcos Zampieri, Shervin Malmasi, Vanessa Murdock and Daniel Kadar. (2020, May). Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*.

Alyssa Lees, Jeffrey Sorensen and Ian Kivlichan. (2020). Jigsaw@ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model. In *EVALITA*.

Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova and Helen Yannakoudakis. (2020). A multimodal framework for the detection of hateful memes. *arXiv preprint arXiv:2012.12871*.

Ilya Loshchilov and Frank Hutter. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yatri Modi and Natalie Parde (2019). The Steep Road to Happily Ever after: an Analysis of Current Visual Storytelling Models. In *Proceedings of the Second Workshop on Shortcomings in Vision and Language*, pages 47–57, Minneapolis, Minnesota. Association for Computational Linguistics.

Niklas Muennighoff. (2020). Vilio: state-of-the-art Visio-Linguistic models applied to hateful memes. *arXiv preprint arXiv:2012.07788*.

Victor Nina-Alcocer. (2018, September). AMI at IberEval2018 Automatic Misogyny Identification in Spanish and English Tweets. In *Ibereval@ sepln* (pp. 274-279).

Debora Nozza, Claudia Volpetti, Elisabetta Fersini. (2019, October). Unintended bias in misogyny detection. In *Ieee/wic/acm international conference on web intelligence* (pp. 149-155). https://doi.org/10.1145/3350546.3352512

Endang Wahyu Pamungkas, Alessandra Teresa Cignarella, Valerio Basile, and Viviana Patti. (2018). 14-ExLab@ UniTo for AMI at IberEval2018: Exploiting lexical knowledge for detecting misogyny in English and Spanish tweets. In *3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018* (Vol. 2150, pp. 234-241). CEUR-WS.

Natalie Parde (2020). And, Action! Towards Leveraging Multimodal Patterns for Storytelling and Content Analysis. In *Proceedings of the 2nd International Workshop on AI for Smart TV Content Production, Access and Delivery (AI4TV '20)*. Association for Computing Machinery, New York, NY, USA, 3. DOI:https://doi.org/10.1145/3422839.3423060

Pulkit Parikh, Harika Abburi, Niyati Chhaya, Manish Gupta and Vasudeva Varma. (2021). Categorizing Sexism and Misogyny through Neural Approaches. *ACM Transactions on the Web (TWEB)*, *15*(4), 1-31. https://doi.org/10.1145/3457189

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger and Ilya Sutskever. (2021, July). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (pp. 8748-8763). PMLR.

Victor Sanh, Lysandre Debut, Julien Chaumond and Thomas Wolf. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Moein Shariatnia. (2021). Moein-Shariatnia/OpenAI-clip: Simple implementation of openai clip model in PYTORCH. GitHub. Retrieved November 29, 2021, from https://github.com/moein-shariatnia/OpenAI-CLIP.

Riza Velioglu and Jewgeni Rose. (2020). Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge. *arXiv preprint arXiv:2012.12975*.

Ron Zhu. (2020). Enhance multimodal transformer with external label and in-domain pretrain: Hateful meme challenge winning solution. *arXiv preprint arXiv:2012.08290*.

Poirot at SemEval-2022 Task 5: Leveraging Graph Network for Misogynistic Meme Detection

Harshvardhan Srivastava
Oracle India
Indian Institute of Technology, Kharagpur
harshvardhan.srivastava@oracle.com

Abstract

In recent years, there has been an upsurge in a new form of entertainment medium called memes. These memes although seemingly innocuous have transcended the boundary of online harassment against women and created an unwanted bias against them. To help alleviate this problem, we propose an early fusion model for the prediction and identification of misogynistic memes and their type in this paper for which we participated in SemEval-2022 Task 5. The model receives as input meme image with its text transcription with a target vector. Given that a key challenge with this task is the combination of different modalities to predict misogyny, our model relies on pre-trained contextual representations from different state-of-the-art transformer-based language models and pre-trained image pre-trained models to get an effective image representation. Our model achieved competitive results on both SubTask-A and SubTask-B with the other competing teams and significantly outperforms the baselines.

1 Introduction

Meme culture in today's virtual climate gives us a variety of insight into the pop culture, general ideology and linguistic conversational manner of the generation. To understand the internet culture, it becomes essential to study memes (Shifman, 2013) and the impact it has on people on the internet. Some of the most popular communication tools on social media platforms are memes. Memes are essentially images characterized by the content of a picture overlaid with text that was introduced by people with the main purpose of being interesting and ironic. Women have a strong presence online, especially on image-based social media like Twitter, Snapchat, and Instagram. 78% of women use social media several times a day, compared to 65% of men (Fersini et al., 2022). While new opportunities are being opened up for women online, systematic inequality and discrimination are being replicated offline from these online spaces in the form of offensive content for women. Most of them were created to make funny jokes, but soon people began to use them as a form of hatred for women, leading to sexist and offensive messages in the online environment, and as a consequence, the sexual stereotyping and gender inequality of the offline world where sexuality stereotypes and gender inequality have been strengthened. This insensitive and obscene type of meme has a profound effect on a person's mental health and can exhibit harmful effects on cognitive and emotional processes leading to mental illnesses as shown in Paciello et al. (2021).

In this work, we present team Poirot's solution to SemEval - 2022 Task 5 competition as described in detail in Fersini et al. (2022). We focused our efforts on our primary approach of building a Multi-Modal module that uses features from both images and text. Furthermore, in this paper, we provide ablation studies on different modalities, relative importance of the different modalities and some training parameters, and show how by changing the module parameters, the predictions on the misogynistic identification of memes aggravates or allays.

2 Background

2.1 Task Description

The organisers have provided us with data tasked with the identification of misogynous memes, taking advantage of both text and images available as source of information. The task is comprised around two main sub-tasks:

- Sub-Task A: first task about misogynous meme identification, where a meme is categorized in a binary format; either as misogynous or not misogynous;
- Sub-Task B: second task, where the type of misogyny is recognized among potential over-

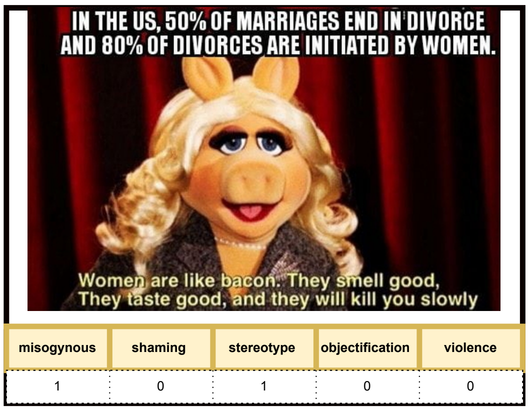

IN THE US, 50% OF MARRIAGES END IN DIVORCE AND 80% OF DIVORCES ARE INITIATED BY WOMEN.
Women are like bacon. They smell good, They taste good, and they will kill you slowly
misogynous
shaming
stereotype
objectification
violence
1
0
1
0
0

Figure 1: Misogyny in meme

lapping categories such as stereotype, shaming, objectification and violence as described in (Fersini et al., 2022).

The sub-tasks are arranged in an increasing range of difficulty. The competition is challenging, as identifying the misogynous nature of a meme is more complex in a multi-modal setting than performing the same task only on textual data. For memes, comprised of image and text information, a multi-modal approach for understanding both visual and textual cues is needed. Also, in sub-task B, the nature of problem difficulty is increased as the type of misogyny has to be identified, which can belong to multiple categories due to the nature of the dataset.

2.2 Dataset

The datasets for the competition provided by the task organisers are memes collected from the web and manually annotated via crowdsourcing platforms (Fersini et al., 2022). Each sample is supported by an image and the corresponding text transcription (if it exists) on the image. An example of the sample is given in Figure 1. The statistical information about the datasets can be found in Table 1.

Additionally, we provide a quick look into the training dataset which has a significant data imbalance for *4* of labels in a number of samples belonging to each of the *4* given labels except the label "misogynous". This imbalance affects the performance of the models on the test set specially in the case of multilabel prediction as not equal training instances are available for each of the classes. The information about the number of samples belonging to each of the 5 classes for the training set is given in Table 1. This dataset imbalance is dealt with in section 3.3.

2.3 Evaluation Criteria

The teams' performance is evaluated by the macro F1 score for task A. For tasks B, the weighted F1 score is computed for each subtask (misogynous, shaming, stereotype, objectification, violence)(Fersini et al., 2022), and the average F1 score of these subtasks is used to rank the systems.

2.4 Related Work

Earlier, some other meme datasets have been created like the dataset created in Oriol et al. (2019) with the intention of automatically detecting hate speech, and the hateful memes dataset by Facebook (Kiela et al., 2020), which created a challenge set for multimodal classification of hatred in memes. Previous work encompassing categories like hate speech, sexism, and toxicity detection in memes has primarily been explored from a textual perspective using Natural Language Processing(NLP). However, recent methods are aiming to use multimodal approaches to solve the issue at hand. VL-BERT(Su et al., 2020) used the single-stream architecture, where a single Transformer is applied to both images and text. ViLBERT(Lu et al., 2019) and LXMERT(Tan and Bansal, 2019) introduced a two-stream architecture where two transformers are applied to images and text independently and later merged by a third transformer. ERNIE-ViL(Yu et al., 2021) incorporates structured knowledge obtained from scene graphs to learn joint representations of vision-language. Zia et al. (2021) presents the multimodal pipeline based on pre-trained visual and textual representations for the shared task involving the detection of hateful memes.

Set	Number of Samples
`Trial`	100
`Train`	10000
`Test`	1000

Label	Positive Samples
misogynous	5000
shaming	1274
stereotype	2810
objectification	2202
violence	953

Table 1: Dataset and Labels Information

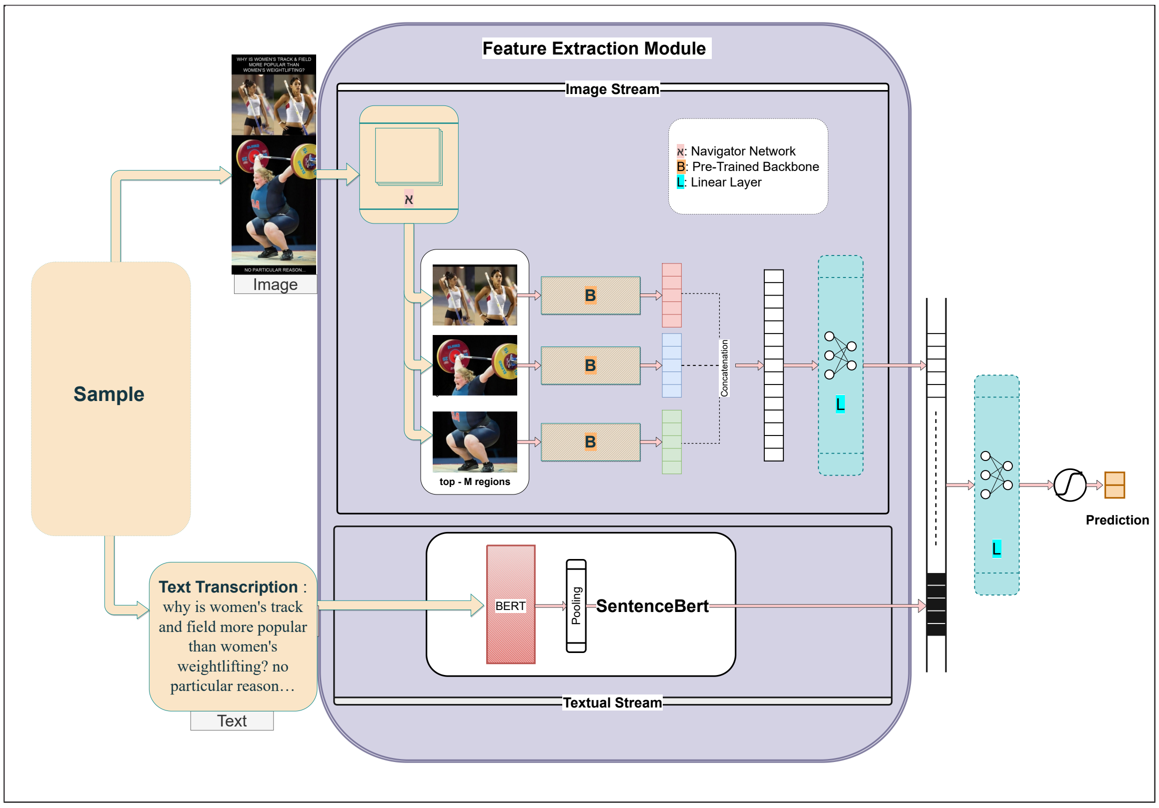


Figure 2: Overview of the binary model used for misogyny detection. The two modalities are passed simultaneously through the Feature Extraction Module in two separate paths and trained together and finally fused together and passed through a linear layer to get binary logits.

3 System Overview

The solution system comprises of 2 separate systems for both the sub-tasks. The approach can be broadly divided into binary approach and multi-label approach.

3.1 Binary Model

Broadly, the model consists of two modal information streams, text and image. The proposed approach leverages on multi-modal information to provide the classification of a sample. We exploit text transcriptions written in natural language jointly with visual information coming from the meme image. In the initial stage the pipeline is divided into two streams running in parallel which on later stage is joint together. The outline of the proposed architecture is shown in Figure 2. The sub-modules are described below:

1. **Image Features Extraction** : This stream is also separated into two sub-modules :
   - *pre-trained Representation Module* : We can use any backbone CNN base models to learn the features of an image. For our experimentation, we use ResNet-101 and ResNet-50 (He et al., 2016) as the backbone *B* model. The rationale behind choosing two separate backbone models is to compare generalised image representation as compared to domain-specific image representation in this case where outright misogyny in images can be detected quickly by looking at nsfw content in the image part of the meme. Thus if the input image $I \in \mathbb{R}^{d \times d}$, where d = 224, the output of the feature extractor would give us intermediate level features $\kappa \in \mathbb{R}^{D}$, where D = 2048,

$$\kappa = B(I)$$

   - *Navigator Module* : We use Navigator Module from the NTS-net as described in (Yang et al., 2018) model to decouple the image into several parts. For an input image, the image is fed into the Navigator network to compute the informativeness of all regions. It is fed to the navigator network, which extracts meaningful parts to separate *top-M* regions. The feature extractor extracts its deep feature map for each of those parts. These features are then concatenated C together:

$$\kappa_i = \aleph(I), i \in [0, M-1]$$

$$f_v = C(\kappa_0, \kappa_1, \kappa_2, ..., \kappa_{M-1}) \in \mathbb{R}^{M \times D}$$

2. **Text Features Extraction:** : The second modality being the textual Stream, uses the SentenceBERT model (Reimers and Gurevych, 2019). SBert modifies the BERT network using a combination of siamese and triplet networks to derive semantically meaningful embedding of sentences. As a state of the art language model, BERT has greatly influenced results in the text classification task as shown in Minaee et al. (2021), we use SentenceBert S model trained on Siamese BERT networks. Thus we convert the given text T transcription into features vector f_t. Formally :

$$f_t = S(T) \in \mathbb{R}^E$$

where E = 768.

The image extraction part and text extraction part is clubbed together to form the **Feature Extraction Module**. If C_{mc} is multimodal feature concatenation logic, then,

$$F = C_{mc}(f_v, f_t)$$

This module outputs a feature vector of size $N = E + D$ features. These features are then passed through a f(linear layer) to output logits which are then passed on to σ function to generate predictions.

$$y_{pred} = \sigma(f(F))$$

3.2 Multi-Label Model

The multi-label model, keeps the feature extracting pipeline of the network in the binary model intact while changing the final output method by using Graph Neural Networks. The model consists of two essential parts : (i) Feature Extraction Module and (ii) Graph classification module. The overall architecture of our model is explained in the Fig. 3.

1. **Feature Extraction Module** : Same as in binary model 3.1

2. **Graph Classification Module** : A graph has an effective message passing system, which can be modelled to find the inter-dependency of the labels amongst each other, and hence, efficiently capture the semantic importance of a label u_1 depending on co-occurring label u_2. We represent each node of the graph input to be a label, having the node features as GloVe embedding having *e* features. Formally, we use Graph Network to learn the multi-label classification model to learn label representation:

$$L_{n+1} = \phi(L_n, A)$$

where $L_n \in \mathbb{R}^{u \times e}$ represents class label representation at *nth* graph layer, ϕ represents the message passing network and $A \in \mathbb{R}^{u \times u}$ represents the adjacency matrix. Through stacking multiple Graph Network Layers, we model the complex inter-relationships among classes.

Creation of Adjacency Matrix : We calculate the label adjacency matrix $\mathbb{A}$ by mining label co-occurrence patterns in the training and trial dataset. Let the label matrix $L_m \in \mathbb{R}^{n_s \times u}$, where n_s are the number of training and trial samples. Then the co-occurrence matrix

$$A_{coo} = L_m^T \times L_m \in \mathbb{R}^{u \times u}$$

To create the adjacency matrix from the co-occurrence matrix and to remove the self node loop from the graph, we create a vector $N_u \in \mathbb{R}^u$, having

$$N_u[i] = A_{coo}[i][i]$$

Finally, the adjacency matrix $\mathbb{A}$ can be constructed as:

$$A_{ij} = \begin{cases} 0 & \text{if } i = j, \\ \frac{A_{coo}[i][j]}{N_u[i]} & \text{otherwise} \end{cases}$$

3.3 Multi-Label Classification Loss

We notice the imbalance present in data for different classes which we can see in Table 1, but the extent of imbalance is different for different labels.

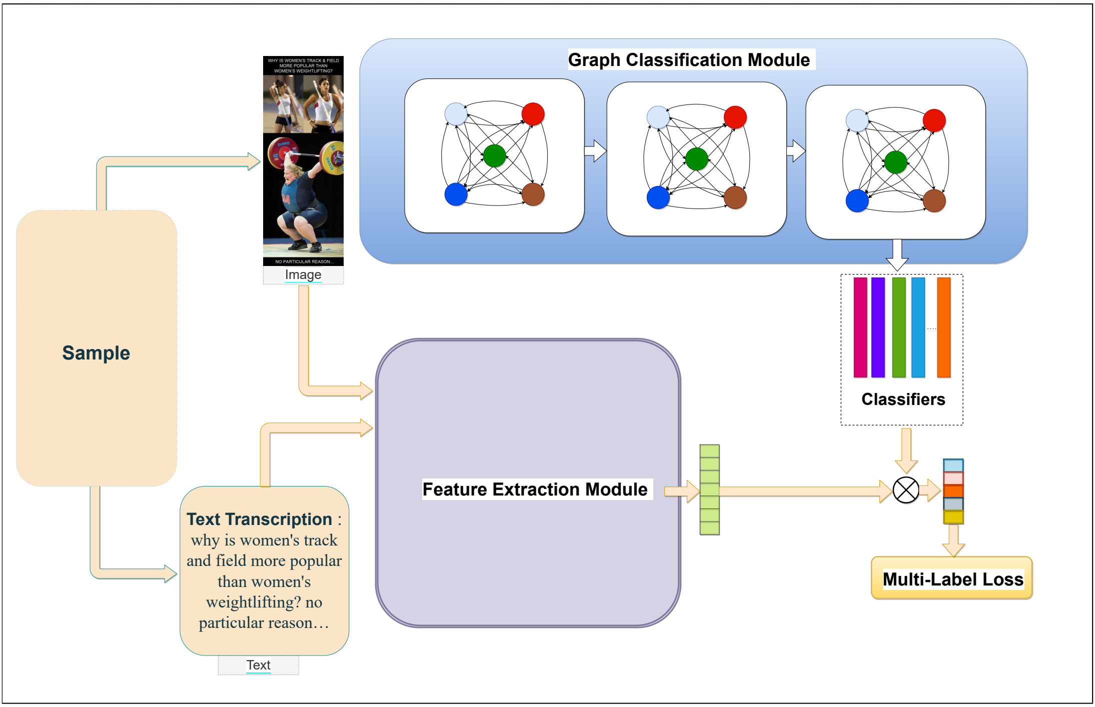

Graph Classification Module
Sample
Image
Classifiers
Feature Extraction Module
Text Transcription : why is women's track and field more popular than women's weightlifting? no particular reason…
Text
Multi-Label Loss

Figure 3: : The general architecture of our Multi-Modal-Multi-Label model. We use image features and text features from the *Feature Extraction* (FE) which has a pre-trained ResNet, pre-trained sentence transformer SBert. The features are passed through a 5-layer classifier stack generated from the Graph Classification Module which takes input the label's semantic information to generate the output multi-label prediction.

This knowledge can be passed to the neural network in terms of class weights in order to penalize adequately. Let n_s be the number of samples in the dataset. We calculate the weighted importance of a class using the below equations:

$$W_p[i] = \frac{N_p[i]}{n_s}, W_n[i] = \frac{N_n[i]}{n_s}$$

where $N_p[i]$, is the number of positive samples for class *i* and $N_n[i]$, is the number of negative samples for class *i*.

$$-dl = W_p * y * \log(p) + W_n * (1 - y) * \log(1 - p)$$

where y is the ground truth and p is the predicted output. The calculated weights are shown in table 2.

	Weights	
Label	**Positive** (W_p)	**Negative** (W_n)
misogynous	1.000	1.000
shaming	3.924	0.573
stereotype	1.779	0.695
objectification	2.270	0.641
violence	5.246	0.552

Table 2: Calculated weights for regularizing cross-entropy loss in the custom loss function

4 Experimental Setup

4.1 Baselines

We use the baseline provided by the task organisers [1] which depend on the Sub-Task and use a different set of features for different tasks:

1. Sub-Task A: Misogynous Meme Identification
   - ***Baseline-Text***: deep representation of text, i.e. a fine-tuned sentence embedding using the USE(Cer et al., 2018) pre-trained model
   - ***Baseline-Image***: deep representation of image content, i.e. based on a fine-tuned image classification model grounded on VGG-16(Liu and Deng, 2015).
   - ***Baseline-IT***: concatenation of deep image and text representations, i.e. based a single layer neural network.
2. Sub-Task B : Type of Misogynous Meme Identification

[1] https://github.com/MIND-Lab/MAMI

- ***Baseline-Flat***: a multi-label model, based on the concatenation of deep image and text representations, for predicting simultaneously if a meme is misogynous and the corresponding type
- ***Baseline-Hierarchical***: a hierarchical multi-label model, based on text representations, for predicting if a meme is misogynous or not and, if misogynous, the corresponding type.

4.2 Hyperparameters and Implementation Details

Before passing the text transcription to the text stream, we apply some basic text preprocessing to all our sentences. First, we normalize all the sentences by converting all white-space characters to spaces. Also, in the image stream, before passing the image to the navigator network, we resize the image to size [224,224] for uniformity and perform random crops and flips before feeding it to the network. When concatenating the image features with textual features, we use a parameter λ to combine the two together. The final feature vector is formulated as following,

$$F = [(1 - \lambda) \times f_v; \lambda \times f_t]$$

We adopt a 2-layer graph network for our best performing system. For node features, we use the 300-Dimensional GloVe embeddings (Pennington et al., 2014) trained on the Wikipedia Dataset. Table 3 contains the list of general hyperparameters we used. We implement the network based on PyTorch.

Parameter Name	Value
Optimizer	AdamW
Pre-Trained BERT LR	2e-4
Navigator Network LR	1e-3
Graph Learning Rate	1e-2
Graph Layer 1 Dim	512
Graph Layer 2 Dim	2048
λ (Concatenation Parameter)	0.7

Table 3: Major hyperparameters used

5 Results

Table 5 and 6 compares the macro and weighted f_1 scores of our best performing models on the binary classification task and the multi-label task respectively alongside the score achieved by the baseline models. We also present several ablations for the best performing models on λ parameter and its effect on the final score achieved by the model. The λ ablations can be found in Table 4.

		Binary	**Multi-Label**
Backbone Model	λ	f_1^{macro}	$f_1^{weighted}$
ResNet-101$_{imagenet}$	0.1	0.601	0.590
	0.2	0.619	0.597
	0.3	0.645	0.622
	0.4	0.689	0.628
	0.5	0.702	0.641
	0.6	0.736	**0.645**
	0.7	**0.741**	0.643
	0.8	0.728	0.631
	0.9	0.702	0.612
ResNet-50$_{nsfw}$	0.1	0.611	0.591
	0.2	0.620	0.595
	0.3	0.691	0.612
	0.4	0.703	0.632
	0.5	0.736	0.634
	0.6	0.749	**0.635**
	0.7	**0.759**	0.632
	0.8	0.734	0.623
	0.9	0.698	0.601

Table 4: λ effect on model performance

ResNet-101$_{imagenet}$ uses the backbone *B* pre-trained on the ImageNet dataset, while ResNet-50$_{nsfw}$[2] model uses the backbone fine-tuned on around 40GB of *nsfw* data. We divide our model for multi-label classification according to different types of loss used during the training stage.

SubTask-A	
Model	f_1^{macro}
Baseline-Text	0.640
Baseline-Image	0.639
Baseline-IT	0.543
Ours(ResNet-101$_{imagenet}$)	0.751
Ours(ResNet-50$_{nsfw}$)	**0.759**

Table 5: Comparing the f_1^{macro} of our methods and the baselines for binary classification task.

5.1 Task Results

Subtask-A: The results of the experiments for the binary classification task can be seen in Table 5.

[2]https://github.com/emiliantolo/pytorch_nsfw_model

SubTask-B	
Model	$f_1^{weighted}$
Baseline-Flat	0.421
Baseline-Heirarchical	0.621
Ours(ResNet-101$_{imagenet}$)	
+ `SM Loss`	0.641
+ `Custom Loss`	**0.645**
Ours(ResNet-50$_{nsfw}$)	
+ `SM Loss`	0.632
+ `Custom Loss`	0.638

Table 6: Comparing the $f_1^{weighted}$ of our methods and the baselines for multi-label classification task.`SM Loss` refers to multi-label SoftMargin loss

		Multi-Label
Backbone Model	**Graph Depth**	$f_1^{weighted}$
ResNet-101$_{imagenet}$	*2-layer*	**0.644**
	3-layer	0.644
	4-layer	0.632
	5-layer	0.628
ResNet-50$_{nsfw}$	*2-layer*	0.641
	3-layer	**0.643**
	4-layer	0.629
	5-layer	0.621

Table 7: Graph Network Depth effect on model performance

For Subtask-A, the models that used multi-model training and an additional navigator network on the image end outperformed the single modality models and the simple multimodal concatenation model. One of the major reasons for our model outperforming the image-only baseline model could be that the navigator network learns to recognise relevant parts of the image as compared to passing the complete image as one. Amongst the model using *ResNet* backbone, the model fine-tuned on *nsfw* had an edge over the model which had been pre-trained on *imagenet* dataset. This can indicate that there is an indicator of women's image representation with the meme being a misogynistic one.

Subtask-B: The results of the experiments for the multi-label classification task can be seen in Table 6. For Subtask-B, the models that used graph network to create independent classifiers and an additional navigator network on the image end outperformed the models using the simple multi-modal concatenation model with classification head and the hierarchical multi-label model using text representations. Amongst the model using *ResNet* backbone, the model fine-tuned on *nsfw* dataset performed poorer to the model which had been pre-trained on *imagenet* dataset. This can be an indicator that general feature representations are perhaps more important for the identification of the specificity of misogyny as compared to that of the fine-tuned feature representations.

5.2 Ablation Studies

In this section, we perform ablation studies from two different aspects, particularly including the sensitivity of the classification models to effects of λ when concatenating the two different types of modalities, visual and textual together, to determine the relative importance of the two with respect to each other, and the other being the depths of Graph Classification Module which we use for the multi-label classification model.

Effects of different threshold values λ : We vary the values of the threshold concatenation parameter λ from 0.1 to 0.9 in steps of 0.1. $\lambda = 0$ corresponds to building the entire feature vector from the visual stream while $\lambda = 1$ corresponds to the entire information coming from the textual stream. The results are shown in table 4, where the performance of the two models based on ResNet-101 backbone are compared pre-trained on two different datasets. It can be observed that the textual stream information is of higher importance in both the classification problem as the performance boost is skewed for roughly $\lambda = [0.6, 0.7]$. It may be due to the fact that in the images as well, a good amount of information that is used to recognise the misogyny of the meme is cognitively of the textual nature, while the image content of the meme is lesser in comparison to its textual counterpart. It may also be that the image content is not of high quality.

Effects of different depth of Graph Classification Network: We vary the values of the number of layers of the graph network from 2 to 5 and observe its effect on the model performance. For the two-layer model, the output dimensions of the layers are 512, 2048, for the three-layer model, the output dimensionalities are 512, 1024 and 2048 for the sequential layers, for the four-layer model, the dimensionalities are 512, 1024, 1024 and 2048, and for the five-layer model, the output dimensions are 512, 1024, 1024, 1024, 2048. As shown in table 7, when the number of graph convolution layers increases, multi-label recognition performance drops on both datasets. The possible reason for the performance drop may be that when using more GCN layers, the propagation between nodes will be accumulated, which can result in over-smoothing.

Effects of using Custom Loss Function: We compare the results for multi-label classification with two types of losses : (i) MultiLabel Soft Margin Loss (`SM Loss`); (ii) Custom Loss as described in 3.3. From table 6, we can see that the `Custom Loss` out performs the `SM Loss` in the experimental runs.

The result can be explained by the fact that weighted classes affect the loss value for positive as well as negative labels.

(i) If the model predicts a positivity for the label which has a higher positive weightage the loss value would increase, thereby forcing the model to not favour one particular label. Similarly, when the model predicts a negative value for a particular label that has a higher negative weightage, the loss value would increase, forcing the model to not favour negativity of a particular label.

(ii) If the model predicts a positivity for the label which has a lower positive weightage, the loss value would decrease, thereby forcing the model to predict favourably for that particular label. Similarly, when the model predicts a negative value for a particular label that has a lower negative weightage, the loss value would decrease, forcing the model to favour the negativity of that particular label.

6 Conclusion

We have described the systems developed by as to solve the Multimedia Automatic Misogyny Identification challenge at Semeval 2022. In our best performing submission for SubTask-A, we framed the problem as a binary classification task and used two separate streams of information simultaneously to identify misogyny, while for our model for SubTask-B, we tried to find the semantic relation between the type of misogyny and their relative importance to solve the problem for Multi-Label classification. By making use of powerful, state-of-the-art, pre-trained models for text and images, our models were able to achieve a high F1 score for both the tasks. Our best performing model ranked 3rd out of the 10 teams submissions on SubTask-A and 22nd out of 30 team submissions on SubTask-B.

As part of future work, we aim to explore alternate approaches to model the multi-label dependencies using Knowledge-Graph and GAT Networks. Also, there seems to be a problem of oversmoothing when increasing the depth of the Graph Classification Module, which we aim to resolve using effective Normalization layers between the graph layers.

References

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624.

Shuying Liu and Weihong Deng. 2015. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv.*, 54(3).

Benet Oriol, Cristian Canton-Ferrer, and Xavier Giró i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. In

NeurIPS 2019 Workshop on AI for Social Good, Vancouver, Canada.

Marinella Paciello, Francesca D'Errico, Giorgia Saleri, and Ernestina Lamponi. 2021. Online sexist meme and its effects on moral and emotional processes in social media. *Comput. Hum. Behav.*, 116:106655.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Limor Shifman. 2013. Memes in a Digital World: Reconciling with a Conceptual Troublemaker. *Journal of Computer-Mediated Communication*, 18(3):362–377.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Ze Yang, Tiange Luo, Dong Wang, Zhiqiang Hu, Jun Gao, and Liwei Wang. 2018. Learning to navigate for fine-grained classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 420–435.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. In *AAAI*.

Haris Bin Zia, Ignacio Castro, and Gareth Tyson. 2021. Racist or sexist meme? classifying memes beyond hateful. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 215–219.

SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic

Ibrahim Abu Farha[1], Silviu Vlad Oprea[1], Steven R. Wilson[1,3], Walid Magdy[1,2]
[1] School of Informatics, The University of Edinburgh, Edinburgh, UK
[2] The Alan Turing Institute, London, UK
[3] Oakland University, Rochester, MI, USA
{i.abufarha,silviu.oprea,steven.wilson,wmagdy}@ed.ac.uk

Abstract

iSarcasmEval is the first shared task to target intended sarcasm detection: the data for this task was provided and labelled by the authors of the texts themselves. Such an approach minimises the downfalls of other methods to collect sarcasm data, which rely on distant supervision or third-party annotations. The shared task contains two languages, English and Arabic, and three subtasks: sarcasm detection, sarcasm category classification, and pairwise sarcasm identification given a sarcastic sentence and its non-sarcastic rephrase. The task received submissions from 60 different teams, with the sarcasm detection task being the most popular. Most of the participating teams utilised pre-trained language models. In this paper, we provide an overview of the task, data, and participating teams.

1 Introduction

Sarcasm is a form of verbal irony that occurs when there is a discrepancy between the literal and intended meanings of an utterance. Through this discrepancy, the speaker expresses their position towards a prior proposition, often in the form of surface contempt or derogation (Wilson, 2006).

Sarcasm is present on the social web and, due to its nature, it can be disruptive of computational systems that harness this data to perform tasks such as sentiment analysis, opinion mining, author profiling, and harassment detection (Liu, 2012; Rosenthal et al., 2014; Maynard and Greenwood, 2014; Van Hee et al., 2018). In the context of SemEval, in particular, Rosenthal et al. (2014) show a significant drop in sentiment polarity classification performance when processing sarcastic tweets, compared to non-sarcastic ones. Such computational systems are widely deployed in industry, driving marketing, administration, and investment decisions (Medhat et al., 2014). In the context of Arabic, Abu Farha and Magdy (2021) show the effect of sarcasm on Arabic sentiment analysis systems, where the performance dropped significantly for the sarcastic tweets. As such, it is imperative to devise models for sarcasm detection.

Such models are usually built in a supervised learning paradigm, relying on a dataset of texts labelled as either sarcastic or non-sarcastic. Two methods have typically been used to label texts for sarcasm: distant supervision (Ptáček et al., 2014; Khodak et al., 2018; Barbieri et al., 2014), where texts are considered sarcastic if they meet predefined criteria, such as including the tag #sarcasm; or manual labelling (Filatova, 2012; Riloff et al., 2013a; Abercrombie and Hovy, 2016), where texts are presented to human annotators. However, as argued by Oprea and Magdy (2020a), both methods could produce noisy labels, in terms of both false positives, and false negatives. For instance, when human annotators label texts, they are limited by their subjective perception of sarcasm, which might differ from the intention of the authors of those texts.

In response, we suggest the current shared task, iSarcasmEval[1]. We rely on a novel method of labelling texts for sarcasm, where the sarcastic nature of texts is self-reported by the authors of those texts. Our shared task is also novel in that it includes two languages, English and Arabic, and includes three subtasks. The first subtask, covering both languages, is sarcasm detection as commonly understood: given a text, determine whether or not it is sarcastic. Next, as the sarcastic texts in our English dataset are also further labelled for the ironic speech category that they represent out of the categories specified by Leggitt and Gibbs (2000), the second subtask is: given an English text, determine which ironic speech category it represents, or whether it is non-sarcastic. Finally, for both languages, we also ask authors to provide

[1]iSarcasmEval datasets are available at: https://github.com/iabufarha/iSarcasmEval

non-sarcastic rephrases of their sarcastic texts. As such, the third subtask, covering both languages, is: given a sarcastic text and its non-sarcastic rephrase, identify the sarcastic text.

We discuss related work in dataset creation, and related SemEval tasks, in Section 2. We introduce the data labelling method, and present statistics on the resulted datasets, in Section 3. We provide details on the shared task in Section 4, and on the submissions in Section 5.

2 Related Work

Most previous textual sarcasm detection datasets have been annotated using a **distant supervision** method. In this approach, texts are considered sarcastic if they meet predefined criteria, such as including specific tags (e.g. #sarcasm, #irony) (Ptáček et al., 2014; Khodak et al., 2018), or being generated by specific accounts (Barbieri et al., 2014). However, this can lead to noisy labels for several reasons. First, the tags may not mark sarcasm, but may constitute the subject or object of conversation, e.g. *"there is so much #sarcasm around!"*. Second, the assumption that certain tags always appear in conjunction with sarcasm, or that certain accounts always generate sarcasm (Barbieri et al., 2014), could lead to further false positives. Third, considering those texts that do not meet the criteria as non-sarcastic is a strong assumption that can lead to false negatives.

Due to the issues outlined above, other work has relied on **manual labelling**, where sarcasm labels are provided by human annotators (Filatova, 2012; Riloff et al., 2013a; Abercrombie and Hovy, 2016). As such, the labels represent *annotator perception*, which may actually differ from *author intention*. Annotators might lack awareness of the contextual devices that, as linguistic studies suggest (Grice, 1975; Sperber and Wilson, 1981; Utsumi, 2000), could be essential for clarifying the sarcastic intention of the authors.

Previous shared tasks in sarcasm detection (Van Hee et al., 2018; Ghanem et al., 2019; Ghosh and Muresan, 2020; Abu Farha et al., 2021) present datasets annotated via the two methods discussed above. The potential noisy labels that these methods can produce gives us reason to be concerned about the effectiveness of models that were trained on such datasets. Recently, (Shmueli et al., 2020) proposed a third method, **reactive supervision**, which aims to collect sarcastic examples based on the conversation dynamics, addressing some of these issues by using statements such as "I was being sarcastic" to automatically label texts. However, this method relies on specific cues of sarcasm which may lead to a sample that is biased toward more confusing examples that required clarification.

Further, the vast majority of sarcasm detection work (Campbell and Katz, 2012; Riloff et al., 2013b; Joshi et al., 2016; Wallace et al., 2015; Rajadesingan et al., 2015; Bamman and Smith, 2015; Amir et al., 2016; Hazarika et al., 2018; Oprea and Magdy, 2019) has focused exclusively on the English language and, due to the sociocultural aspects of sarcastic communication (Oprea and Magdy, 2020b), it is unclear if models trained on English could generalise to other languages. To our knowledge, the small amount of work on other languages such as Arabic (Karoui et al., 2017; Ghanem et al., 2019; Abbes et al., 2020; Abu-Farha and Magdy, 2020) relies on either manual labelling or distant supervision. Representative of distant supervision is the work of Karoui et al. (2017), who consider Arabic equivalents of #sarcasm, such as #سخرية, #مسخرة, and #استهزاء, to collect sarcastic tweets. Other work, (Abbes et al., 2020; Ghanem et al., 2019; Abu-Farha and Magdy, 2020; Abu Farha et al., 2021), used either manual labelling, or a mix between manual labelling and distant supervision. When working with Arabic data, these two labelling methods are even more problematic considering the large number of dialects of the language that vary both across and within countries. Relying on predefined tags in modern standard Arabic (MSA), such as those specified above, can thus lead to a plethora of false negatives. Similarly, the third-party annotators might be unfamiliar with the dialect of the texts they are annotating, resulting in erroneous manual labels.

3 Dataset

3.1 Overview

In light of the issues raised in Section 2, we propose the current shared task for sarcasm detection. We introduce a new data collection method where the sarcasm labels for texts are *provided by the authors themselves*, thus eliminating labelling proxies (in the form of predefined tags, or third-party annotators). We use this method to collect two datasets, one in English and one in Arabic.

Within each dataset, for each sarcastic text, we also ask its author to rephrase the text to convey the same intended message without using sarcasm. Finally, for the English texts, we ask trained annotators to further label each text into one of the categories of ironic speech defined by Leggitt and Gibbs (2000): sarcasm, irony, satire, understatement, overstatement, and rhetorical question. For the Arabic dataset, we also include the dialect label of the text. As such, in both datasets, each text has at least the following information attached to it: (a) a label specifying its sarcastic nature (sarcastic or non-sarcastic), provided by its author; and (b) a rephrase provided by its author that conveys the same message non-sarcastically.

3.2 Data Collection

For both English and Arabic, the sarcasm labels of texts, as well as their non-sarcastic rephrases, are provided by the authors those texts. However, the methods in which we reach authors, and how their texts are sourced, differ slightly across the two languages.

For **English texts**, we used the Prolific Academic platform[2] to recruit native English speakers who were Twitter users. We asked these participants to provide links to one sarcastic and three non-sarcastic tweets that they had posted in the past. The tweet labels were, thus, implicitly specified by the authors themselves in the process.

To collect **Arabic texts**, we were unable to find a suitable number of native Arabic speakers through Prolific Academic. Further, through our pilot study, we found that asking for tweets directly resulted in low quality data. Therefore, we used the Appen crowdsourcing platform[3] to recruit native Arabic speakers, and instead of asking for previous tweets, we asked the participants to write a short sarcastic text on the spot. Through our pilot study, we found this on-the-spot generation approach to result in high quality data. However, this methodology only provided us with sarcastic examples. As non-sarcastic examples, we used a subset of the ArSarcasm-v2 dataset (Abu Farha et al., 2021), mainly those tweets that were annotated as non-sarcastic with 100% confidence, i.e. labelled non-sarcastic by all annotators.

For each sarcastic text in **both** the English and the Arabic datasets, we also asked participants to provide an *explanation* of why the text was sarcastic, and a *rephrase* that would convey the same message non-sarcastically. For Arabic, we also collected the dialect label. We included five main dialects: Modern Standard Arabic (MSA), Gulf, Nile Basin, Levant, and North Africa.

While we asked participants to provide examples of *sarcastic* texts, we found that the provided English texts that reflected a range of different ironic speech categories, not just sarcasm. Therefore, in a second annotation stage, we paid a trained annotator to further label each English-language text with the ironic speech categories that it reflected. We adopted the categorisation presented by Leggitt and Gibbs (2000): (1) *sarcasm*: contradicts the state of affairs, is directed towards an addressee, and expresses a critical attitude; (2) *irony*: contradicts the state of affairs, may or may not be directed towards an addressee, but if it is, is not obviously critical towards that addressee; (3) *satire*: is directed towards and addressee whom it appears to support, but underneath it express disagreement, mocking, contempt, or derogation; (4) *understatement*: does not contradict the state of affairs, but undermines its weight; (5) *overstatement*: does not contradict the state of affairs, but assigns unrealistically high weight to it; (6) *rhetorical question*: a question with an implicated answer that contradicts the state of affairs. Note that these categories are not mutually exclusive. A text could belong to more than one category, e.g. it could be both sarcastic, and an understatement. Regarding Arabic, we did not go the next step to include the sarcasm categories. This is because Arabic linguists had similar disagreements regarding the differences between sarcasm and irony (Andalib and Far Shirazi, 2019). Also, it would have been a challenging task to recruit linguists who are familiar with available dialects.

3.3 Test Sets

To construct our test sets, we employed, for both languages, an approach similar to that used to collect training data in Arabic. We chose this method for collecting English test data due to restrictions that were imposed on us by the Prolific Academic crowdsourcing platform on the collection of tweets that belonged directly to survey participants.

3.4 Quality Control

For English, we made sure all tweets were posted at least 48 hours before the survey submission, and came from the same account. Further, a trained an-

[2]https://prolific.co
[3]https://appen.com

Language	**Sarcastic text**	**Unsarcastic rephrased**
EN	Gotta love people who follow you and unfollow because you don't follow them within in an hour or 2. Sorry I don't stay on Twitter 24/7.	I dislike people who follow me, only to unfollow me when I don't follow back right away. I'm not on Twitter that much to follow right away.
AR	محمد صلاح ينقذ سمكة من الغرق، الله على اخلاقك يا فخر العرب (Mo Salah saves a fish from drowning. Amazing manners, you Arabs' pride)	محمد صلاح يحمل سمكة (Mohammad Salah holds a fish)

Table 1: Examples of sarcastic tweets (tweet text) from our English and Arabic dataset along with the rephrase that authors gave that convey the same meaning non-sarcastically (rephrased).

notator consulted all survey responses provided and filtered out spurious sarcastic examples that were either unlikely to reflect sarcasm, or had uninformative explanations as to why they were sarcastic.

For Arabic, the data collection was run multiple times during a period of 8 months. In this stage, we managed to collect around 2,000 sarcastic sentences. After manual inspection, we noticed that a large portion of the texts were not truly sarcastic, or that the non-sarcastic phrasing was not informative. Thus, we hired native speakers for each of the dialects to check texts for sarcasm, and to provide or improve the non-sarcastic phrasing, if needed.

4 iSarcasmEval Details

4.1 Task Description

We formulate three subtasks:

- **Subtask A - Sarcasm Detection**: Given a text, determine whether it is sarcastic or non-sarcastic;
- **Subtask B - Sarcasm Category Classification**: Given a text, determine which ironic speech categories it belongs to, if any;
- **Subtask C - Pairwise Sarcasm Identification**: Given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which is the sarcastic one.

Subtasks A and C are suggested for both languages. Subtask B is only suggested for English, as we only have ironic speech category labels for English texts.

4.2 iSarcasmEval dataset

The datasets for both languages are provided as a list of texts. Each text is accompanied by a sarcasm label, indicating whether or not it is sarcastic. For sarcastic texts, there is a rephrase that conveys the same message non-sarcastically. For English sarcastic texts, there is a label specifying the category of ironic speech that it reflects. For Arabic texts, there is a label specifying the dialect. Table 1 shows a sample from our datasets, one in English, and one in Arabic. For English, the training set

split	**total**	**sarcastic**	**non-sarcastic**
train	3,103	745	2,358
test (subtask A)	1,400	200	1,200
test (subtask C)	400	200	200

Table 2: Statistics for the Arabic training set, and test sets for subtasks A and C, as discussed in Section 4.2.

split	**total**	**sarcastic**	**non-sarcastic**
train	4,335	867	3,468
test (subtask A)	1,400	200	1,200
test (subtask C)	400	200	200

Table 3: Statistics for the English training set, and test sets for subtasks A and C, as discussed in Section 4.2.

contains 867 and 2,601 sarcastic and non-sarcastic texts, respectively. Recall that each sarcastic text has an associated non-sarcastic rephrase. These 867 rephrases can be used as additional non-sarcastic examples. The division of sarcastic texts into ironic speech categories in the training set is shown in Table 4. There is a separate test set for each subtask. As such, the test set for subtask A contains 200 sarcastic texts, and a total of 1,200 non-sarcastic texts. The same texts, in the same arrangement, constitute the test set for subtask B. The test set for subtask C contains 200 sarcastic texts, along with their 200 non-sarcastic rephrases. These are presented as pairs, the task being to distinguish the sarcastic text from its rephrase. This information is summarised in Table 3 (training set, and test sets for subtasks A and C); and in Table 4 (training set, and test set for subtask B).

For Arabic, the training set contains 3,103 texts, 745 of which are sarcastic. Similar to English, the sarcastic text have their non-sarcastic phrasing too. The test sets are the same size as the English test sets for both subtasks A and C. Table 2 provides a summary of the Arabic dataset splits. Table 5 provides the distribution of the whole dataset over the available dialects. It is noticeable that the majority of the sarcastic examples are in the Egyptian dialect (Nile Basin). In the future, we hope to have a higher coverage of the other dialects.

split	sarcasm	irony	satire	underst.	overst.	rhet. quest.
train	713	155	25	10	40	101
test (subtask B)	180	20	49	1	10	11

Table 4: Statistics for the English training set, and test set for subtask B, as discussed in Section 4.2.

dialect	total	sarcastic	non-sarcastic
MSA	2,035	82	1,953
Nile Basin	2,072	827	1,245
Levant	322	76	246
Gulf	278	36	242
North Africa	195	124	71

Table 5: Distribution of the Arabic dataset over the dialects.

4.3 Evaluation Metrics

The main evaluation metric for subtask A is the F1-score of the sarcastic class, referred to as $F_1^{\text{sarcastic}}$. It is computed as follows:

$$F_1^{sarcastic} = 2 \cdot \frac{P^{sarcastic} \cdot R^{sarcastic}}{P^{sarcastic} + R^{sarcastic}}, \qquad (1)$$

Where $P^{sarcastic}$, $R^{sarcastic}$ are the precision and recall with respect to the sarcastic class, respectively.

For subtask B, the main evaluation metric is the macro-F1 score over all the categories of ironic speech:

$$F_1 = \frac{1}{n}\sum_{c=1}^{n}(F_1^c) \qquad (2)$$

Where F_1^c represents the F_1 score for the cth category and n is the number of categories.

For subtask C, the main evaluation metric is accuracy. This is appropriate since we have an equal number of sarcastic and non-sarcastic examples.

$$Accuracy = \frac{C}{N} \qquad (3)$$

Where C is the total number of correct predictions and N is the total number of pairs of text.

5 Participating Teams

5.1 Overview

The shared task saw the participation of 60 unique teams. The most popular task was subtask A (sarcasm detection) with 43 participants for English and 32 for Arabic. Subtask B received 22 submissions and subtask C received 16 submissions for English and 13 for Arabic. The following sections provide an overview of the top teams' approaches.

5.2 Subtask A (Sarcasm Detection) - English

Table 6 shows the results for English. We created two baseline models for subtask A. The first one uses the BERT language model (Devlin et al., 2019) to produce contextual representations of the input text, and considers the embedding corresponding to the [CLS] token as the aggregated representation of the input. Finally, this is provided to a classification head whose output we interpret as the probability that the input is sarcastic. We use the implementation provided as part of the transformers library Wolf et al. (2020), and initialise the encoder with the `bert-base-uncased` checkpoint published on the Huggingface model hub [4]. We fine-tune it for a maximum of 100 epochs, but use early stopping regularisation with a patience of 3. We use a learning rate of $5e-5$, and clip the norm of the gradients to 1. This results in a baseline $F_1^{\text{sarcastic}}$ of 0.348, listed as baseline-bert in Table 6. The second baseline uses a support vector machine (SVM), with a polynomial kernel of degree 3, to classify tf-idf representations of input texts. This results in a baseline $F_1^{\text{sarcastic}}$ of 0.275, listed as baseline-svm in Table 6. For both baselines, we consider the rephrases as additional non-sarcastic examples. In a preprocessing step, we remove all hashtags and urls, and replace user handles with the token `@user`.

As shown in Table 6, the team ranking first, `stce` (Yuan et al., 2022), achieved an $F_1^{\text{sarcastic}}$ of 0.605. They use an ensemble learning approach with a combination of hard and soft voting between three models, all based on the transformer architecture: RoBERTa (Liu et al., 2019), initialised with the `roberta-large` checkpoint; DeBERTa (He et al., 2021), initialised with the `deberta-v3-large` checkpoint; and XLM-RoBERTa (Conneau et al., 2020), initialised with the `xlm-roberta-large` checkpoint. XLM-RoBERTa is employed to make use of the Arabic training data for informing the classification of English texts. They experiment with several strategies to achieve their results. First, in addition to the task dataset, they also consider public datasets, including iSarcasm (Oprea and Magdy, 2020a), the dataset published by Van Hee et al. (2018), and a sample of texts from the multimodal sarcasm dataset [5]. Second, they extract statistical

[4] https://huggingface.co

[5] https://github.com/headacheboy/data-of-multimodal-sarcasm-detection

and text features that they concatenate to the text itself before providing it to the models above, such as emoji and part-of-speech information. They also use multi-sample dropout, contrastive loss functions, and adversarial training.

The team ranking second, `X-PuDu` (Han et al., 2022), achieved an $F_1^{\text{sarcastic}}$ of 0.569. They ensemble two transformer-based models: ERNIE-M (Ouyang et al., 2021), and DeBERTa, mentioned above. After providing the input text to the models, they consider the embedding corresponding to the [CLS] token as the representation of the input, which they provide to a classification head. The final ensemble considers not just the individual architectures above, but also the same architecture under different hyperparameter configurations. Using ERNIE-M, they train on both English and Arabic at the same time.

The team ranking third, `TUG-CIC` (Aroyehun et al., 2022), achieved an $F_1^{\text{sarcastic}}$ of 0.530. They use the BERT model mentioned above, but initialised with different `BERTweet` checkpoints, which they fine-tune on the SPIRS sarcasm dataset (Shmueli et al., 2020), before fine-tuning it on the data provided here. They also apply label smoothing.

5.3 Subtask A (Sarcasm Detection) - Arabic

As mentioned previously, the main metric for subtask A is the F-score of the sarcastic class. Table 7 shows the results for Arabic. The participating teams made extensive use of Arabic pre-trained language models such as MARBERT (Abdul-Mageed et al., 2021). We created two baselines for this task, the first is a Bert-based model and the other is an SVM model. We fine-tuned MARBERT for 6 epochs with a learning rate of 5e-5. For the SVM model, we used uni-gram features. Both models were trained without the non-sarcastic phrasing.

As shown in the Table 7, the top team `CS-UM6P` (El Mahdaouy et al., 2022a) achieved an $F_1^{\text{sarcastic}}$ of 0.563. This team utilised a transformer encoder (MARBERT), attention layer, and a classifier. They applied the attention to the contextualised embeddings. The classifier, which is composed of one hidden layer, is fed the concatenation of the pooled output of the encoder and the attention's output. The official submission was an ensemble of two variants of this model that are trained with and without the non-sarcastic rephrasing. `AlexU-AL` (Lotfy et al., 2022) achieved the second place with an $F_1^{\text{sarcastic}}$ of 0.508. Their model is similar to our baseline where the fine-tuned MARBERT for text classification. The results are quite close to our baseline with a small difference that can be attributed to the choice of hyperparameters. The third team, `remarchka` (Abdel-Salam, 2022), also used MARBERT in a similar way to the baseline and `AlexU-AL` team. Their results are quite close to the other two models with $F_1^{\text{sarcastic}}$ of 0.477. The other teams followed a similar approach where they utilise one of the many flavours of Arabic-specific models or the multilingual ones. A few of the participants relied on hand-engineered features along with conventional classifiers such as SVM and Decision Trees.

5.4 Subtask B (Sarcasm Category Classification)

Table 8 shows the results. We created two baseline models for subtask B. The first baseline, listed as baseline-majority in the table, always predicts that the input reflects the ironic speech category of sarcasm, and no other category. This was chosen as it is dominant in the training set, as seen in Table 4. As a second baseline, we use the BERT language model to produce contextual representa-

r	Team Name	Affiliation	$F_1^{\text{sarcastic}}$
1	stce	PALI Inc., China	0.605
2	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.569
3	TUG-CIC	TU Graz, Austria	0.530
4	Plumeria	Indian Institute of Technology Kanpur, India	0.477
5	John Thomson	University of Alberta, Canada	0.456
6	Naive	Dalian University of Technology, China	0.452
7	MarSan_AI	Part AI Research Center, Iran	0.434
8	LISACTeam	Sidi Mohamed Ben Abdellah University, Morocco	0.429
9	LT3	Ghent University, Belgium	0.424
10	niksss	-	0.402
11	Amobee	-	0.401
12	YNU-HPCC	Yunnan University, China	0.392
13	Dartmouth	Dartmouth College, USA	0.386
14	underfined	Ping An Life Insurance Company of China, China	0.383
15	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.371
16	UTNLP	University of Tehran, Iran	0.369
17	Jumana-Safa	-	0.356
18	cnxup	University of Chinese Academy of Sciences, China	0.351
-	baseline-bert	-	0.348
19	IISERB Brains	Indian Institute of Science Education and Research, Bhopal, India	0.345
20	rematchka	Cairo University, Egypt	0.341
21	R2D2	Vellore Institute of Technology, India	0.328
22	AMI_UofA	University of Alberta, Canada	0.312
23	Amrita-CEN	Amrita Vishwa Vidyapeetham, India	0.308
24	DUCS	University of Delhi, India	0.307
25	Happy New Year	-	0.276
-	baseline-svm	-	0.275
26	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.270
27	TechSSN	Sri Sivasubramaniya Nadar College of Engineering, India	0.264
28	NULL	Auburn University, USA	0.260
29	Cyborgs	-	0.248
30	I2C	Universidad de Huelva, Spain	0.245
31	MaChAmp	IT University of Copenhagen, Denmark	0.241
32	ISD	Stanford University, USA	0.240
33	SPDB	-	0.215
34	xuyt3	-	0.215
35	MACHON	Jerusalem College of Technology, Israel	0.215
36	FII_UAIC	University of Iasi, Romania	0.207
37	connotation_clashers	University of Tübingen, Germany	0.202
38	GetSmartMSEC	Meenakshi Sundararajan Engineering College, Chennai, India	0.201
39	UoR-NCL	University of Reading, UK	0.195
40	JCT	Jerusalem College of Technology, Israel	0.184
41	UMUTeam	Universidad de Murcia, Spain	0.180
42	MACHON	Jerusalem College of Technology, Israel	0.168
43	NARD@KGP	IIT Kharagpur, India	0.155

Table 6: Subtask A (English) results in descending order according to the main metric ($F_1^{\text{sarcastic}}$). The table shows the teams' names, rank, affiliation, and score.

r	Team Name	Affiliation	$F_1^{\text{sarcastic}}$
1	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.563
2	AlexU-AL	Alexandria University, Alexandria, Egypt	0.508
-	baseline-bert	-	0.480
3	rematchka	Cairo University, Egypt	0.477
4	HIGH-TECH Team	High Technology School, Morcco	0.468
5	Naive	Dalian University of Technology, China	0.461
6	akaBERT	Helwan University, Egypt	0.444
7	SarcasmDet	Jordan University of Science and Technology	0.431
8	Alexa	Open-Insights, Tarjamah	0.420
9	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.419
10	Plumeria	Indian Institute of Technology Kanpur, India	0.407
11	niksss	-	0.400
12	MaChAmp	IT University of Copenhagen, Denmark	0.396
13	underfined	Ping An Life Insurance Company of China, China	0.378
14	BFCAI	Benha University	0.375
15	AM	Alexandria University,Egypt	0.369
16	cnxup	University of Chinese Academy of Sciences, China	0.367
17	stce	PALI Inc., China	0.367
18	NULL	Auburn University, USA	0.358
19	Dartmouth	Dartmouth College, USA	0.350
20	Amrita-CEN	Amrita Vishwa Vidyapeetham, India	0.349
21	YNU-HPCC	Yunnan University, China	0.323
22	UMUTeam	Universidad de Murcia, Spain	0.318
23	connotation_clashers	University of Tübingen, Germany	0.301
24	LEV	Jerusalem College of Technology, Israel	0.295
25	NARD@KGP	IIT Kharagpur, India	0.281
26	JCT	Jerusalem College of Technology, Israel	0.257
27	MACHON	Jerusalem College of Technology, Israel	0.256
28	iaf7	-	0.229
29	TechSSN	Sri Sivasubramaniya Nadar College of Engineering, India	0.229
30	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.192
31	MarSan_AI	Part AI Research Center, Iran	0.188
-	baseline-svm	-	0.139
32	UoR-NCL	University of Reading, UK	0.115

Table 7: Subtask A (Arabic) results in descending order according to the main metric ($F_1^{\text{sarcastic}}$). The table shows the teams' names, rank, affiliation, and score.

r	Team Name	Affiliation	macro F-score
1	PALI-NLP	Ping An, China	0.1630
2	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.0875
3	MaChAmp	IT University of Copenhagen, Denmark	0.0851
4	Naive	Dalian University of Technology, China	0.0809
5	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.0799
6	Plumeria	Indian Institute of Technology Kanpur, India	0.0778
7	R2D2	Vellore Institute of Technology, India	0.0760
8	IISERB Brains	Indian Institute of Science Education and Research, India	0.0751
9	MarSan_AI	Part AI Research Center, Iran	0.0743
10	I2C	Universidad de Huelva, Spain	0.0699
11	YNU-HPCC	Yunnan University, China	0.0646
12	John Thomson	University of Alberta, Canada	0.0601
13	AMI_UofA	University of Alberta, Canada	0.0601
14	Dartmouth	Dartmouth College, USA	0.0590
15	Amrita-CEN	Amrita Vishwa Vidyapeetham, India	0.0567
16	rematchka	Cairo University, Egypt	0.0560
17	TechSSN	Sri Sivasubramaniya Nadar College of Engineering, India	0.0465
18	NARD@KGP	IIT Kharagpur, India	0.0446
-	baseline-bert	-	0.0431
19	GetSmartMSEC	Meenakshi Sundararajan Engineering College, Chennai, India	0.0387
20	niksss	-	0.0380
-	baseline-majority	-	0.0380
21	Suhaib-Aburaidah	-	0.0346
22	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.0313

Table 8: Subtask B results in descending order according to the main metric (macro F-score). The table shows the teams' names, rank, affiliation, and score.

tions of the input text, and consider the [CLS] embedding. We provide this to a classification head with a 6-dimensional output, one corresponding to each category of ironic speech. We apply the sigmoid function to each unit in the classification head, interpreting the output as the probability that the input text reflects the ironic speech category corresponding to that unit. We fine-tune the model in a similar setting as we did for subtask A. This results in a baseline macro F-score of 0.0431, listed as baseline-bert in Table 8[6].

As shown in Table 8, the team ranking first, `PALI-NLP` (Du et al., 2022), achieved a macro F-score of 0.1630. They use an ensemble learning approach, where the weight assigned to a model corresponds to its performance on a validation set. The models they consider are BERT, initialised with the `BERT-base` checkpoint; RoBERTa, initialised with the `RoBERTa-base` checkpoint; and BERTweet, initialised with the `BERTweet-base` checkpoint. Models have a classification head attached that inputs the embedding corresponding to the [CLS] token. They also use adversarial training and multi-sample dropout to improve generalisation.

The team ranking second, `CS-UM6P` (El Mahdaouy et al., 2022b), achieved a macro F-score of 0.0875. They use a model similar to GAN-BERT (Croce et al., 2020). It uses a generator that, conditioned on an ironic speech category, produces fake embeddings from a random noise that would resemble representations of examples from that ironic speech category. A discriminator is trained to recognise real examples from fake ones, while the generator is trained to cause the discriminator to classify fake examples as real. The discriminator is also trained to classify the real examples as either sarcastic, or non-sarcastic.

The team ranking third, `MaChAmp`, achieved a macro F-score of 0.0851. They first we pretrain a RemBERT (Chung et al., 2020) multi-task model across all the tasks. Then, they re-train a model for each task individually. They use the hyperparameters of `MaChAmp v0.3`(van der Goot et al., 2021), which were finetuned on the xTREME benchmark (Hu et al., 2020).

5.5 Subtask C (Pairwise Sarcasm Identification) - English

Table 9 shows the results for English. We used baselines similar to those from subtask A, but modified the input. Specifically, given a sarcastic text and its rephrase, we produced two training examples. The first was the concatenation of the sarcastic text and the rephrase, in this order, separated by a [SEP] token. This example had label 0, indicating the position of the sarcastic text. The second example was the concatenation of the rephrase and the sarcastic text, in this order, and had label 1. The first baseline, shown as baseline-bert in Table 9, achieves an accuracy of 0.765, while the second

[6]The complete results are available in Table 11 in Appendix A. Those include the scores over each sarcasm category.

r	Team name	Affiliation	Accuracy
1	X-PuDu	Baidu, China	0.870
2	Naive	Dalian University of Technology, China	0.855
3	YNU-HPCC	Yunnan University, China	0.805
4	Plumeria	Indian Institute of Technology Kanpur, India	0.790
5	LISACTeam	Sidi Mohamed Ben Abdellah University, Morocco	0.775
6	UTNLP	University of Tehran, Iran	0.770
7	MarSan_AI	Part AI Research Center, Iran	0.765
-	baseline-bert	-	0.765
8	R2D2	Vellore Institute of Technology, India	0.750
9	NARD@KGP	IIT Kharagpur, India	0.735
10	rematchka	Cairo University, Egypt	0.720
11	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.695
12	Dartmouth	Dartmouth College, USA	0.660
13	IISERB Brains	Indian Institute of Science Education and Research, Bhopal, India	0.625
14	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.495
-	baseline-svm	-	0.495
15	GetSmartMSEC	Meenakshi Sundararajan Engineering College, Chennai, India	0.340
16	MaChAmp	IT University of Copenhagen, Denmark	0.250

Table 9: Subtask C (English) results in descending order according to the main metric (accuracy). The table shows the teams' names, rank, affiliation, and score.

r	Team Name	Affiliation	Accuracy
1	Naive	Dalian University of Technology, China	0.930
2	HIGH-TECH Team	High Technology School, Morocco	0.885
3	MarSan_AI	Part AI Research Center, Iran	0.875
4	Plumeria	Indian Institute of Technology Kanpur, India	0.870
5	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.840
6	rematchka	Cairo University, Egypt	0.800
-	baseline-bert	-	0.800
7	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.780
8	YNU-HPCC	Yunnan University, China	0.755
9	AlexU-AL	Alexandria University, Alexandria, Egypt	0.745
10	Dartmouth	Dartmouth College, USA	0.680
11	NARD@KGP	IIT Kharagpur, India	0.665
-	baseline-svm	-	0.585
12	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.465
13	MaChAmp	IT University of Copenhagen, Denmark	0.200

Table 10: Subtask C (Arabic) results in descending order according to the main metric (accuracy). The table shows the teams' names, rank, affiliation, and score.

one, baseline-svm, achieves 0.495.

As shown in Table 9, the team ranking first, X-PuDu (Han et al., 2022), achieved an accuracy of 0.870. The same team ranked second for task A, and the approach here is rather similar, except for representing the input as we do above. The team ranking second, Naive, achieved an accuracy of 0.855. They used a RoBERTa model, initialised with the RoBERTa-large checkpoint, with a classification head appended. The team ranking third, YNU-HPCC (Zheng et al., 2022), achieved an accuracy of 0.805. They also used a RoBERTa model. They did not use any external datasets during training. We suspect the difference in performance between the second and third teams to be, at least in part, the result of data preprocessing and hyperparameter optimisation.

5.6 Subtask C (Pairwise Sarcasm Identification) - Arabic

Table 10 shows the results of this task. To prepare the baselines, we utilised the models from subtask A. Since the task is to decide which text is sarcastic out of the given pair, we ran the models from subtask A on each sentence and chose the one that had a higher probability of being sarcastic. The top team, Naive (Zefeng et al., 2022), achieved an accuracy of 0.930. They utilised the model created for subtask A, where they would compare the probabilities for each sentence and choose the one with a higher probability. Their model in subtask A relied on the voting of a 5 folds cross-validation of a Bert model. High-Tech team (Hamza et al., 2022) achieved the second place with an accuracy of 0.885. They fine-tuned AraBERT (Antoun et al., 2020) on the concatenation of the sarcastic sentence and its non-sarcastic phrasing. The third team, MarSan_AI (Najafi and Tavan, 2022), achieved an accuracy of 0.875. Their model consisted of a T5 encoder (Raffel et al., 2020) followed by a transformer and Bi-LSTM, the output of the Bi-LSTM is fed to an attention layer followed by a fully connected layer. The final prediction is the softmax of the output from the fully connected layer. The other teams followed the same trend where they utilised the models from subtask A for this task. Most of these models are transformer-based models such as MARBERT and AraBERT.

In general, it is noticeable that the results on Arabic are slightly higher than the ones on English. This can be due to the slight difference in the nature of the data. As mentioned in Section 3.2, the English data are original tweets that the authors wrote before our data collection process. The Arabic data was collected on the fly, and therefore more likely to contain clear signs of sarcasm as the authors were specifically asked to provide new sarcastic and non-sarcastic phrasings.

6 Conclusion

This paper provides an overview of SemEval-2022 task 6, iSarcasmEval, which targets intended sarcasm detection. We provide an overview of the current state of research on sarcasm detection focusing on data collection methods. We introduce two new datasets for sarcasm detection in English and Arabic. The data was collected by asking people to provide and label their own words as sarcastic or not, hence intended sarcasm. iSarcasmEval contains three subtasks: sarcasm detection, sarcasm category classification, and sarcasm identification given a pair of sentences. The task was quite popular with the participation of around 62 teams. In this paper, we provide a high-level overview of the approaches of top teams in each of the subtasks. Transformer models were dominant in all subtasks.

Detecting sarcasm in texts remains challenging; detecting the ironic speech category even more so. We hope our shared task will draw the attention of the community towards these important tasks. We suggest two main directions that future work could consider.

First, in this shared task, sarcasm detection was performed by solely mining lexical and pragmatic cues from the texts being classified. However, the sarcastic intention of the authors might be unclear without reference to their previous utterances, and their sociocultural background (Oprea and Magdy, 2020b). We suggest future datasets are needed to provide access to such information, and future models that account for it effectively.

Second, the low performance achieved by the models on subtask B requires further investigation. First, alternative categorisations could be considered. Second, the ironic speech category labels should either be provided by the authors themselves, to avoid any bias introduced by trained annotators, or more emphasis should be placed on annotator training and annotation guideline clarity, to mitigate labelling noise that might indeed account, at least in part, for the low performance presented here. Finally, more effort is needed to develop more effective models, likely making use of information outside of the texts being classified, including prior assumptions about the nature of ironic speech, sociocultural information about the authors, if available, as well as commonsense facts.

7 Acknowledgements

This work was partially supported by the Defence and Security Programme at the Alan Turing Institute, funded by the UK Government; the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1); the University of Edinburgh; and The Financial Times.

References

Ines Abbes, Wajdi Zaghouani, Omaima El-Hardlo, and Faten Ashour. 2020. DAICT: A dialectal Arabic irony corpus extracted from Twitter. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6265–6271, Marseille, France. European Language Resources Association.

Reem Abdel-Salam. 2022. reamtchka at semeval-2022 task 6: Investigating the effect of different loss functions for sarcasm detection for unbalanced datasets. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Gavin Abercrombie and Dirk Hovy. 2016. Putting Sarcasm Detection into Context: The Effects of Class Imbalance and Manual Labelling on Supervised Machine Classification of Twitter Conversations. In *Proceedings of the ACL 2016 Student Research Workshop*, pages 107–113. ACL.

Ibrahim Abu-Farha and Walid Magdy. 2020. From arabic sentiment analysis to sarcasm detection: The arsarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT)*. European Language Resources Association (ELRA).

Ibrahim Abu Farha and Walid Magdy. 2021. A comparative study of effective approaches for arabic sentiment analysis. *Information Processing & Management*, 58(2):102438.

Ibrahim Abu Farha, Wajdi Zaghouani, and Walid Magdy. 2021. Overview of the wanlp 2021 shared task on sarcasm and sentiment detection in arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 296–305, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mario J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. In *CoNLL*, pages 167–177. ACL.

Ali Andalib and Seyyed Heydar Far Shirazi. 2019. Controversy over the concept of irony (=al-mophareqeh) from sarcasm to contradiction; a linguistic and semantic approach. *Researches in Arabic language*, 11(20):121–134.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Segun Aroyehun, Jason Angel, and Alexander Gelbukh. 2022. Tug-cic at semeval-2021 task 6: Two-stage fine-tuning for intended sarcasm detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

David Bamman and Noah A. Smith. 2015. Contextualized sarcasm detection on twitter. In *ICWSM*, pages 574–577. AAAI Press.

Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. 2014. Italian irony detection in twitter: a first approach. In *CLiC-it*, page 28. AILC.

John D Campbell and Albert N Katz. 2012. Are there necessary conditions for inducing a sense of sarcastic irony? *Discourse Processes*, 49(6):459–480.

Hyung Won Chung, Thibault Févry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2020. Rethinking embedding coupling in pre-trained language models. *CoRR*, abs/2010.12821.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

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

Xiyang Du, Dou Hu, JIN MEI ZHI, Lianxin Jiang, and Xiaofeng Shi. 2022. Pali-nlp at semeval-2022 task 6: isarcasmeval- fine-tuning the pre-trained model for detecting intended sarcasm. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Abdelkader El Mahdaouy, Abdellah El Mekki, Kabil Essefar, Abderrahman Skiredj, and Ismail Berrada. 2022a. Cs-um6p at semeval-2022 task 6: Transformer-based models for intended sarcasm detection in english and arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Abdelkader El Mahdaouy, Abdellah EL MEKKI, Kabil Essefar, Abderrahman Skiredj, and Ismail Berrada. 2022b. Cs-um6p at semeval-2022 task 6: Transformer-based models for intended sarcasm detection in english and arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *LREC*. ELRA.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Véronique Moriceau, and Paolo Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in arabic tweets. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 10–13.

Debanjan Ghosh and Smaranda Muresan. 2020. Figlang2020 - sarcasm detection shared task.

H. P. Grice. 1975. Logic and conversation. In Peter Cole and Jerry L. Morgan, editors, *Syntax and Semantics: Vol. 3: Speech Acts*, pages 41–58. Academic Press.

Alami Hamza, Abdessamad Benlahbib, and Alami Ahmed. 2022. High tech team at semeval-2022 task 6: Intended sarcasm detection for arabic texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Yaqian Han, Yekun Chai, Shuohuan Wang, Yu Sun, Hongyi Huang, Guanghao Chen, Yitong Xu, and Yang Yang. 2022. X-pudu at semeval-2022 task 6: Multilingual learning for english and arabic sarcasm detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. In *COLING*, pages 1837–1848. ACL.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080.

Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection? In *EMNLP*, pages 1006–1011. ACL.

Jihen Karoui, Farah Banamara Zitoune, and Veronique Moriceau. 2017. Soukhria: Towards an irony detection system for arabic in social media. *Procedia Computer Science*, 117:161–168.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018. A large self-annotated corpus for sarcasm. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

John S Leggitt and Raymond W Gibbs. 2000. Emotional reactions to verbal irony. *Discourse processes*, 29(1):1–24.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

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

Aya Lotfy, Marwan Torki, and Nagwa El-Makky. 2022. Alexu-al at semeval-2022 task 6: Detecting sarcasm in arabic text using deep learning techniques. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Diana Maynard and Mark Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

Walaa Medhat, Ahmed Hassan, and Hoda Korashy. 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4):1093–1113.

Maryam Najafi and Ehsan Tavan. 2022. Marsan at semeval-2022 task 6: isarcasm detection via t5 and sequence learners. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Silviu Oprea and Walid Magdy. 2019. Exploring author context for detecting intended vs perceived sarcasm. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2859, Florence, Italy. Association for Computational Linguistics.

Silviu Oprea and Walid Magdy. 2020a. isarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Silviu Vlad Oprea and Walid Magdy. 2020b. The effect of sociocultural variables on sarcasm communication online. *Proc. ACM Hum.-Comput. Interact.*, 4(CSCW1).

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE-M: Enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 27–38, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *COLING*, pages 213–223. ACL.

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

Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *WSDM*, pages 97–106. ACM.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013a. Sarcasm as contrast between a positive sentiment and negative situation. In *EMNLP*, pages 704–714. ACL.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013b. Sarcasm as contrast between a positive sentiment and negative situation. In *EMNLP*, pages 704–714. ACL.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. SemEval-2014 task 9: Sentiment analysis in twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland. Association for Computational Linguistics.

Boaz Shmueli, Lun-Wei Ku, and Soumya Ray. 2020. Reactive Supervision: A New Method for Collecting Sarcasm Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2553–2559, Online. Association for Computational Linguistics.

Dan Sperber and Deirdre Wilson. 1981. Irony and the use-mention distinction. *Philosophy*, 3:143–184.

Akira Utsumi. 2000. Verbal irony as implicit display of ironic environment: Distinguishing ironic utterances from nonirony. *Journal of Pragmatics*, 32(12):1777–1806.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Byron C. Wallace, Do Kook Choe, and Eugene Charniak. 2015. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1035–1044, Beijing, China. Association for Computational Linguistics.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743.

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

Mengfei Yuan, Zhou Mengyuan, Lianxin Jiang, Yang Mo, and Xiaofeng Shi. 2022. stce at semeval-2022 task 6: Sarcasm detection in english tweets. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Li Zefeng, Yu Bingjie, Tuerxun Tunike, Li Zhaoqing, and Wang Yuhan. 2022. Naive at semeval-2022 task 6. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Guangmin Zheng, Jin Wang, and Xuejie Zhang. 2022. Ynu-hpcc at semeval-2022 task 6: Transformer-based model for intended sarcasm detection in english and arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

A Appendix A

Table 11 shows the complete results for subtask B.

r	Team Name	Affiliation(s)	macro F-score	F1-Sarcasm	F1-irony	F1-satire	F1-understatement	F1-overstatement	F1-rhetorical question
1	PALI-NLP	Ping An, China	0.1630	0.4828	0.1863	0.0667	0.0000	0.0870	0.1556
2	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.0875	0.2314	0.1622	0.0392	0.0000	0.0000	0.0923
3	MaChAmp	IT University of Copenhagen, Denmark	0.0851	0.2404	0.0567	0.1379	0.0000	0.0000	0.0755
4	Naive	Dalian University of Technology, China	0.0809	0.2370	0.1489	0.0000	0.0000	0.0000	0.0992
5	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.0799	0.2271	0.1685	0.0000	0.0000	0.0000	0.0840
6	Plumeria	Indian Institute of Technology Kanpur, India	0.0778	0.2251	0.1266	0.0263	0.0000	0.0000	0.0889
7	R2D2	Vellore Institute of Technology, India	0.0760	0.2480	0.0323	0.1387	0.0034	0.0000	0.0339
8	IISERB Brains	Indian Institute of Science Education and Research, India	0.0751	0.2294	0.0963	0.0833	0.0000	0.0000	0.0414
9	MarSan_AI	Part AI Research Center, Iran	0.0743	0.1981	0.0653	0.0733	0.0000	0.0000	0.1091
10	I2C	Universidad de Huelva, Spain	0.0699	0.2430	0.0485	0.0000	0.0000	0.0000	0.1280
11	YNU-HPCC	Yunnan University, China	0.0646	0.2382	0.0577	0.0000	0.0000	0.0000	0.0920
12	John Thomson	University of Alberta, Canada	0.0601	0.2039	0.1569	0.0000	0.0000	0.0000	0.0000
13	AMI_UofA	University of Alberta, Canada	0.0601	0.2039	0.1569	0.0000	0.0000	0.0000	0.0000
14	Dartmouth	Dartmouth College, USA	0.0590	0.2293	0.0202	0.0824	0.0000	0.0077	0.0143
15	Amrita-CEN	Amrita Vishwa Vidyapeetham, India	0.0567	0.2180	0.0293	0.0461	0.0074	0.0245	0.0150
16	rematchka	Cairo University, Egypt	0.0560	0.2251	0.0285	0.0664	0.0000	0.0161	0.0000
17	TechSSN	Sri Sivasubramaniya Nadar College of Engineering, India	0.0465	0.2278	0.0282	0.0000	0.0000	0.0095	0.0137
18	NARD@KGP	IIT Kharagpur, India	0.0446	0.2281	0.0282	0.0000	0.0000	0.0000	0.0112
-	baseline-bert	-	0.0431	0.3130	0.1667	0.0000	0.0000	0.0000	0.0597
19	GetSmartMSEC	Meenakshi Sundararajan Engineering College, Chennai, India	0.0387	0.2321	0.0000	0.0000	0.0000	0.0000	0.0000
20	niksss	-	0.0380	0.2278	0.0000	0.0000	0.0000	0.0000	0.0000
-	baseline-majority	-	0.0380	0.2279	0.0000	0.0000	0.0000	0.0000	0.0000
21	Suhaib-Aburaidah	-	0.0346	0.2075	0.0000	0.0000	0.0000	0.0000	0.0000
22	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.0313	0.1538	0.0337	0.0000	0.0000	0.0000	0.0000

Table 11: Subtask B results in descending order according to the main metric (macro F-score). The table shows the teams' names, rank, affiliation, and score for each class.

PALI-NLP at SemEval-2022 Task 6: iSarcasmEval- Fine-tuning the Pre-trained Model for Detecting Intended Sarcasm

Xiyang Du and **Dou Hu** and **Meizhi Jin** and **Lianxin Jiang** and **Yang Mo** and **Xiaofeng Shi**
Ping An Life Insurance Company of China, Ltd.
{DUXIYANG037, HUDOU470, JINMEIZHI005
JIANGLIANXIN769, MOYANG853, SHIXIAOFENG309}
@pingan.com.cn

Abstract

This paper describes the method we utilized in the SemEval-2022 Task 6 iSarcasmEval: Intended Sarcasm Detection In English and Arabic. Our system has achieved 1st in SubtaskB, which is to identify the categories of intended sarcasm. The proposed system integrates multiple BERT-based, RoBERTa-based and BERTweet-based models with finetuning. In this task, our contribution is listed as follow: 1) we reveal several large pre-trained models' performance on tasks coping with the tweet-like text. 2) Our methods prove that we can still achieve excellent results in this particular task without a complex classifier adopting some proper training method. 3) we found there is a hierarchical relationship of sarcasm types in this task.

1 Introduction

Generally speaking, when we communicate through natural language, the literal meaning of the words is consistent with the meaning we want to express. Sarcasm is a form of linguistic expression when this "congruence" is broken (Wilson, 2006).

Due to the inherent metaphorical nature and subtle sentimental expression of this particular form of language expression. The detection task related to this kind of text, which is a negative expression of a positive emotion or the positive expression of negative emotion, is extremely difficult for machines (Yaghoobian et al., 2021). This sarcasm data also weakens the detection modules that are widespread in our society (Maynard and Greenwood, 2014).

Previous work shows that sarcasm often comes with incongruity between expectation and reality (Gibbs Jr et al., 1994). Many works attempt to model this incongruity within the text (Tay et al., 2018; Xiong et al., 2019). For multi-model data, some works use the features from different modalities (Schifanella et al., 2016; Cai et al., 2019), and some works shows that inter-modality incongruity is also an important feature for multi-modal sarcasm detection (Pan et al., 2020).

The SemEval-2022 Task6 (Abu Farha et al., 2022) is designed to detect sarcasm and sarcasm types in twitter texts. In Subtask B, if a tweet is not sarcastic, it should not be annotated with any sarcasm label; if it is sarcastic, we need to detect which sarcasm it is, and it could have different sarcasm type at the same time. The main metric of this task is the Macro-F1 score of all sarcasm types.

We design a simple and effective system for this task. The system is based on a large-scale pre-trained model based on bi-directional transformers (Vaswani et al., 2017) and fine-tuned to obtain the final output. First, we augmented the iSarcasm dataset with additional datasets, and then we set an appropriate learning rate for each layer and set the model with an appropriate initialization state. Next, we strengthen the model's generalization ability through adversarial training, multi-sample dropout and other approaches. Finally, we use the [CLS] token of the last layer of the encoder to perform fine-tuning on the final training dataset and ensemble them using the hierarchical way.

2 System Overview

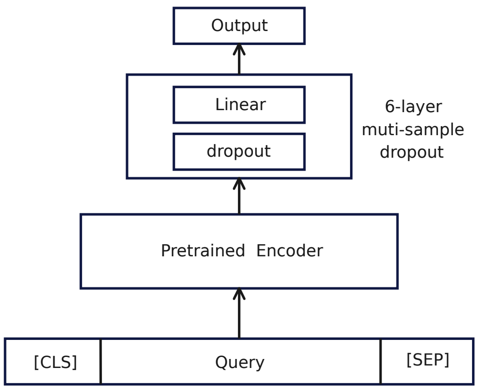


Figure 1: The overall architecture

Figure 1 shows our model architecture. This task is a multi-label text classification task, so we follow the common input format of BERT, that is, using [CLS] and [SEP] as the starting and ending token of the text. In addition to this, we applied data cleaning to the text, replacing "@xxx" with <user>, "#xxx" with <tag>, and "http:xxx" with <url>. It is worth mentioning that we did not do any processing on emojis because we think emojis may be an important feature representing the gap between text semantics and underlying sentiment. These actions are done automatically by the tokenizer. The obtained final token embedding, segment embedding and position embedding constitute the input of the pre-trained model encoder.

In the last layer of the pre-trained encoder, we can acquire the representation of all tokens. In this task, we only select the representation of the first [CLS] token. After that, a layer-norm operation and multi-sample dropout will be utilized on the representation from the encoder. Finally, we use BEC loss as our loss function.

2.1 Pretrained Model

Our submitted architecture integrates three pre-trained language models of different architectures.

BERT-base(BERT) (Devlin et al., 2018): BERT adopts the multi-layer bidirectional transformer encoder to obtain the representation of a query. In the pre-training procedure, BERT conducts two different pre-training objectives. 1. Masked language modelling (MLM) objective. This task predicts a masked token based on a randomly masked input. 2. Next sentence prediction (NSP) objective. The goal of this task is to predict whether the second sentence is the following sentence of the first one.

RoBERTa-base(RoBERTa) (Liu et al., 2019): RoBERTa adopts the same model architecture as BERT and improves the pre-training. It believes that the pre-training of BERT is insufficient, so RoBERTa executes pre-training using longer sentences, more data, and a larger batch size than BERT uses. The author also believes that the NSP task of BERT is redundant and removed NSP from pre-training. Meanwhile, the MLM task is improved at the same time, and the token is dynamically masked during the training process.

BERTweet-base(BERTweet) (Nguyen et al., 2020): BERTweet follows the RoBERTa training procedure, and it is the first large-scale pre-trained language model that uses Twitter texts as a pre-training corpus. Therefore, BERTweet has better performance on tasks related to the tweet-like text.

2.2 Adversiral Training

We also incorporate adversarial training into the training process. The objective of adversarial training is to improve the generalization of the model by perturbing the embedding. For the calculation of this perturbation, we mainly implement two different methods. The Fast Gradient Method (FGM) calculates the disturbance at the moment through the gradient (Miyato et al., 2016), while the Projected Gradient Descent (PGD) executes this process through more steps and additionally adds spherical mapping to prevent the perturbation from being too large (Madry et al., 2017). During the training process, we adopt adversarial training on both the embedding layer and the first layer of the encoder.

2.3 Multi-sample Dropout

Dropout is a common and effective way to increase the generalization of deep neural networks. It can effectively reduce the overfitting of the model by ignoring some neurons in training according to a certain probability. The multi-sample dropout (Inoue, 2019) we use in this paper is an enhanced dropout method. It goes through multiple dropout operations and averages the output obtained by each dropout operation as the final output. In multi-sample dropout, the weights of each dropout layer and classifier layer will be shared, so multi-sample dropout can achieve better results than the original dropout and not bring a significant increase in computational cost.

2.4 Contrastive Loss

Contrastive learning has drawn attention for its' excellent performance. The main idea is to shorten the distance between similar samples(in this task, similar means having the same label) and separate the samples that are not similar. In the SubtaskB, we mainly implement supervised contrastive loss (SupConLoss)(Khosla et al., 2020).

2.5 Ensemble method

For this task, we adopt a hierarchical model ensemble approach. First, we give the models corresponding voting weights based on the performance of each model on the validation set. The weights are calculated as the square root of the inverse of the model's rank among all models. Then we conduct two votes, one for the first two labels (Sarcasm, Irony) and one for the last four labels (Satire, understatement, overstatement, rhetorical question). In this way, we get the final output.

3 Experimental Setup

3.1 Dataset

iSarcasm-2022. Dataset of SemEval-Task6, consisting of tweets text and correspoding sacarsm types. For tweets that are sarcasm, the sarcasm type is given as annotated by language experts, and a single tweet may contain one or more sarcasm types. Along with the sarcastic tweets is the "Rephrase" written by the same poster of the tweet. "Rephrase" contains the same expressive meaning as sarcastic tweets but without sarcasm.

iSarcasm (Oprea and Magdy, 2019). iSarcasm dataset, consisting of tweets texts and correspoding sacarsm types just like iSarcasm-2022. We only leverage the sarcastic tweet which is identityfied as sarcasm. The publisher of sarcastic tweets provides the sarcasm type of iSacarsm. No "Rephrase" is provided.

3.2 Training Details

We fine-tune the pre-train models with batch size 128, sequence length 64,multi-sample dropout of 0.4, threshold 0.2. We set peak learning rate 1e5 for ten epochs and apply layer-wise learning rate Decay for each layer. We set AdamW and Lookahead as our optimizer and set cosine warm-up during the first 0.1 of the updates followed by a linear decay. We conduct validation three times per epoch and perform early stopping. We set the multi-sample layer to six layers and adopt PGD on the embedding and first encoder layers. The training is done on NVidia V100 GPUs. All the F1 result is performance on the test set.

4 Results

Our model performence is shown in Table 1

4.1 Pretrained Model Selection and Data Analysis

We try three transformer-based pre-trained language models in this task, BERT, RoBERTa and BERTweet. From the Table 1 we can see that although RoBERTa achieves better results than BERT, they both perform far worse than BERTweet. The possible reason is due to the difference in the pre-training corpus. The dataset texts of this competition are all tweeted texts of users, and these texts have linguistic features like hashtags and emoji, making them significantly different from standard texts. BERTweet conducts pre-training on tweet texts while BERT and RoBERTa do not have such settings, which leads to BERTweet being better able to handle this particular type of data.

For the fine-tune dataset selection, we extend the dataset provided by the organiser with an additional 777 samples based on the competition requirements that additional data could be used. By analysing the dataset, we found that: 1) The competition dataset reflects a long tail that Understatement, Overstatement, and Rhetorical question are pretty rare. 2) There is an apparent hierarchical relationship between each label. We analysed the combination of labels and found that six labels can be categorised into primary labels (sarcasm, irony) and secondary labels(satire, understatement, Overstatement, rhetorical question). The hierarchical relationship is presented in the Fig 2 . The standard BEC loss and re-weighted BEC loss are tested in this task. The result is shown in the Table 2. Models gain a significantly 10.37% increase in the test set from 0.1417 to 0.1564 of macro-f1 score in the latter setting. We used this system as our baseline.

4.2 Multi-sample Dropout

We experiment with four different multi-sample dropout layer settings, ranging from 2 to 8 layers and the result is shown in Tabel 3. We finally implement 6-layer multi-sample dropout in this task, which achieved 0.1578 in macro-f1 compared to the 0.1564 of baseline.

4.3 Adversiral Training

Tabel 4 display the performance of different adversarial training strategies. We conduct experiments on two common used adversarial training methods and test the effect of the adversarial rate. In the end, we find that PGD is slightly better than FGM as a whole. When the ratio is 0.5, PGD is optimal as adversarial rate 0.5, which increases 1.98% compared to the baseline. If combined with the optimal multi-sample dropout method, the model can obtain 7.10% improvement, reaching a macro-f1 score of 0.1675.

4.4 Contrastive Loss

We experiment with SupCon loss. We can see the result on Tabel 5. There is an indeed decrease when

Model	Macro-F1	F1-SCM	F1-IRN	F1-ST	F1-UST	F1-OST	F1-RQ
`BERT-base-uncased`	0.0766	0.2605	0.0976	0.0000	0.0000	0.0000	0.1013
`RoBERTa-base`	0.0991	0.2921	0.1915	0.0000	0.0000	0.0000	0.1111
`BERTtweet-base`	0.1417	0.4749	0.2151	0.0000	0.0000	0.0000	0.1600
`Our Baseline`	0.1564	0.4760	0.1630	0.0667	0.0000	0.0976	0.1441
`Our Submitted Model`	0.1630	0.4828	0.1863	0.0667	0.0000	0.0870	0.1556
`Our Best Model(single)`	0.1675	0.4586	0.1854	0.1000	0.0000	0.0930	0.1682

Table 1: Performance of final result

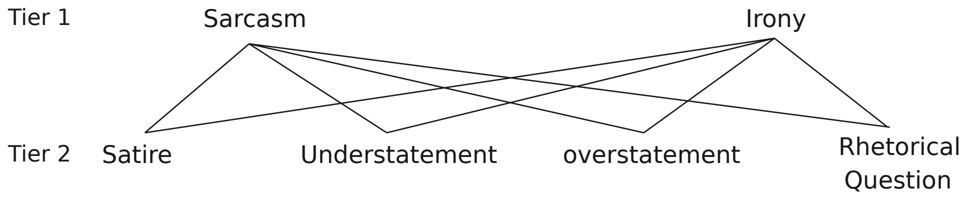


Figure 2: The hirechical relationship of sarcasm types

Model(base)	Loss	Macro-F1
`RoBERTa`	non-weighted	0.0991
`RoBERTa`	re-weighted	0.1034
`BERTweet`	non-weighted	0.1417
`BERTweet(Base)`	re-weighted	0.1564

Table 2: Performance of different pre-trained models applying non-weighted or re-weighted loss

Setting	Macro-F1
`Base`	0.1564
`Base+M-dropout`	0.1578

Table 3: Performance of models applying multi-sample dropout

Setting	M-dropout	Ad-rate	Macro-F1
`Base`	False	0	0.1564
`Base+PGD`	False	0.5	0.1595
`Base+FGM`	False	0.5	0.1593
`Base`	True	0	0.1578
`Base+PGD`	True	0.5	0.1675

Table 4: Performance of models applying different adversarial training strategies

Setting	M-dropout	Macro-F1
`Base`	False	0.1564
`Base+SupCon`	False	0.1529
`Base`	True	0.1578
`Base+SupCon`	True	0.1670

Table 5: Performance of models applying SupCon loss

performing SupCon loss. However, when multi-sample dropout is adopted along with SupCon loss, it shows excellent results, an increase of 6.77% compared to baseline. See Table 5 for performance of contrastive loss applied.

5 Conclusion

We employ the large pre-trained models and fine-tune them for sarcasm category discrimination. We compare the performance of different pre-trained models on Subtask B of SemEval-2022 Task 6. The results show that the difference between the pre-training corpus and the downstream task corpus will significantly affect the performance of the model. We find that the pre-trained model using the default training settings performed poorly on this task, and good model initialization and training strategies can help improve this situation. We also find that there is a hierarchical relationship between the types of sarcasm which we believe is an important feature worth exploiting.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Yitao Cai, Huiyu Cai, and Xiaojun Wan. 2019. Multi-modal sarcasm detection in twitter with hierarchical fusion model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2506–2515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Raymond W Gibbs Jr, Raymond W Gibbs, and Jr Gibbs. 1994. *The poetics of mind: Figurative thought, language, and understanding*. Cambridge University Press.

Hiroshi Inoue. 2019. Multi-sample dropout for accelerated training and better generalization. *arXiv preprint arXiv:1905.09788*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Diana G Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Lrec 2014 proceedings*. ELRA.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *arXiv preprint arXiv:2005.10200*.

Silviu Oprea and Walid Magdy. 2019. isarcasm: A dataset of intended sarcasm. *arXiv preprint arXiv:1911.03123*.

Hongliang Pan, Zheng Lin, Peng Fu, Yatao Qi, and Weiping Wang. 2020. Modeling intra and inter-modality incongruity for multi-modal sarcasm detection. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1383–1392.

Rossano Schifanella, Paloma de Juan, Joel Tetreault, and Liangliang Cao. 2016. Detecting sarcasm in multimodal social platforms. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1136–1145.

Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. *arXiv preprint arXiv:1805.02856*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743.

Tao Xiong, Peiran Zhang, Hongbo Zhu, and Yihui Yang. 2019. Sarcasm detection with self-matching networks and low-rank bilinear pooling. In *The World Wide Web Conference*, pages 2115–2124.

Hamed Yaghoobian, Hamid R Arabnia, and Khaled Rasheed. 2021. Sarcasm detection: A comparative study. *arXiv preprint arXiv:2107.02276*.

stce at SemEval-2022 Task 6: Sarcasm Detection in English Tweets

Mengfei Yuan and **Mengyuan Zhou** and **Lianxin Jiang** and **Yang Mo** and **Xiaofeng Shi**
PALI Inc.
Shenzhen, China
{YUANMENGFEI854, ZHOUMENGYUAN425, JIANGLIANXIN769, MOYANG853, SHIXIAOFENG309}@pingan.com.cn

Abstract

This paper describes the systematic approach applied in "SemEval-2022 Task 6 (iSarcasmEval) : Intended Sarcasm Detection in English and Arabic". In particular, we illustrate the proposed system in detail for SubTask-A about determining a given text as sarcastic or non-sarcastic in English. We start with the training data from the officially released data and then experiment with different combinations of public datasets to improve the model generalization. Additional experiments conducted on the task demonstrate our strategies are effective in completing the task. Different transformer-based language models, as well as some popular plug-and-play proirs, are mixed into our system to enhance the model's robustness. Furthermore, statistical and lexical-based text features are mined to improve the accuracy of the sarcasm detection. Our final submission achieves an F1-score for the sarcastic class of 0.6052 on the official test set (the top 1 of the 43 teams in "SubTask-A-English" on the leaderboard).

1 Introduction

Sarcasm is a sophisticated communication technique to express emotions, attitudes, feelings, and evaluations. Sarcastic and ironic texts typically do not contain words with negative polarity, hostile attitudes, or offensive in their literal sense, but rather express the contradiction or opposite of the literal meanings (Filik et al., 2016; Van Hee et al., 2018; Reyes and Rosso, 2014; Verma et al., 2021). Sarcasm detection can be considered a particular sentiment analysis task, applied to detect texts that are intended to use some exaggeration, understatement, or rhetoric content to express criticism or praise for people or events. Many researchers have conducted different deep learning methods (Poria et al., 2016; Kumar et al., 2020; Zhang et al., 2019), traditional machine learning method (Buschmeier et al., 2014; Hernández-Farías et al., 2015; Yaghoobian et al., 2021), and big data approaches (Bharti et al., 2016; Sarsam et al., 2020; Ortega-Bueno et al., 2019) to improve the accuracy of irony or sarcasm auto-detection.

SemEval-2022 Task 6 (iSarcasmEval) is a sarcasm detection task (Abu Farha et al., 2022). The standard training dataset includes 3468 English tweets and 3102 Arabic tweets. The English training dataset provides 862 sarcastic tweets along with their non-sarcastic rephrases, while the Arabic datasets provides 745 sarcastic samples. For the English dataset, each sarcastic tweet is also labeled as a fine-grained multi-class and multi-label ironic tag such as satire, understatement, overstatement, and rhetorical questions.

SubTask-A is a binary classification task to predict whether a given tweet is sarcastic or not. Table 1 shows one sarcastic tweet and its non-sarcastic version, and one non-sarcastic tweet from the released dataset. We could notice that the raw tweets are pretty noisy and contain user information, URLs, hashtags, etc. Some of the sarcastic tweets also contain sarcastic-related words such as "irony" or "sarcastic" in their hashtags. Many non-sarcastic tweets include confused, unfriendly words or denial attitudes.

In this paper, we demonstrate the following contributions: 1) The discrepancy in prediction performance using different transformer-based language models; 2) The improvement of adding the public dataset and mining effective text features; 3) The enhancement obtained by incorporating various constrative learning loss functions; 4) Model generalization is improved by incorporating the multi-sample dropout layer into the output of pretrained language models. On Subtask-A, our system achieves an F1 score for the sarcasm category of 0.6052 and a Macro F1 score of 0.7675.

2 System Overview

The final submitted result is a contribution from various classification models using the voting mech-

Sarcastic tweet	@PFTompkins Her family should definitely not seek mental health guidance.
Sarcastic tweet rephrase	They should seek guidance.
Non-sarcastic example	I wonder if it's too late for me to re-enroll in University and relive it all just one last time.

Table 1: Sarcastic and non-sarcastic tweet examples

anism. The outcome is a fusion of 15 predictions trained using different transformer-based language models, external datasets, text features, and deep learning-based techniques such as constrastive loss, adversarial training, multi-sample dropout, etc.

The proposed model is trained with a 5-fold cross-validation with a randomly distributed seed. The basic classification model is structured with a multi-sample dropout layer after the pooling layer of the pretrained model. RoBERTa-large, XLM-RoBERTa-large and DeBERTa-v3-large are alternatively adopted in these 15 models. Models are trained using the AdamW optimizer with a learning rate of 1e-05 in the fast gradient method. Four dropout layers with a rate of 0.4 are picked in our multi-sample dropout module. When we make models based on DeBERTa-v3-large, we change the dropout rate to 0.2, which has been suggested by previous studies (He et al., 2020b, 2021).

The cross-entropy loss (L_{Xent}) is used as the classification loss, and the additional XNET loss (L_{NTXent}) with a temperature of 0.2 is selected as the metric learning loss in our system. Equation 1 shows the combination of two types of loss. The weight parameter (w) used to balance the combination of multiple losses is set as 0.1.

$$Loss = (1 - w)L_{Xent} + wL_{NTXent} \quad (1)$$

Additionally, three external datasets are added to the official SemEval-2022 data in the proposed models for further training. Text features are directly concatenated to the training texts in some proposed models as well. The featuring mining, text preprocessing, and the voting method are described below. The scheme of data preparation, training, and prediction processes is demonstrated in Figure 1.

2.1 Pre-processor

The raw English tweets in the official training data contain many noises such as misleading hashtags, usernames, website links, and emojis in different formats. We detected and replaced usernames and links with special tokens. Additionally, we extended some common English abbreviations, such as U, idk, omg, sry, etc. to full-spelled words to keep the whole dataset in the same phase. However, we intend to keep words with unusual capitalization, wrong spelling, and repeated punctuation in raw tweets since people sometimes prefer to express exaggeration and emphasis in this way.

2.2 External Data

Besides the officially released data, we trained the model with three public datasets. The description and the source for additional data are illustrated as below.

(1) The iSarcasm [1] is a public dataset of English tweets. Each tweet is labeled as either sarcastic or non-sarcastic. Each sarcastic tweet is labeled with a fine-grained ironic label as well. We obtained 2279 non-sarcastic and 563 sarcastic tweets using the tweet API (Oprea and Magdy, 2020).

(2) The Multi-modal Sarcasm data [2] contains 33,859 images with descriptions where the sarcastic and non-sarcastic text are uniformly distributed. We used the text information only to train our classification model. In reality, we randomly took 10,000 texts out of the full dataset as the additional dataset. This strategy can speed up the training process and avoid the bias from a large number of additional datasets as well.

(3) The dataset released by SemEval-2018 task3 [3] is also considered in our training dataset (Van Hee et al., 2018). The data format and the task description are pretty similar to our task. This dataset contains 3800 tweets with uniform sarcastic/non-sarcastic labels.

[1] https://github.com/silviu-oprea/iSarcasm

[2] https://github.com/headacheboy/data-of-multimodal-sarcasm-detection

[3] https://competitions.codalab.org/competitions/17468

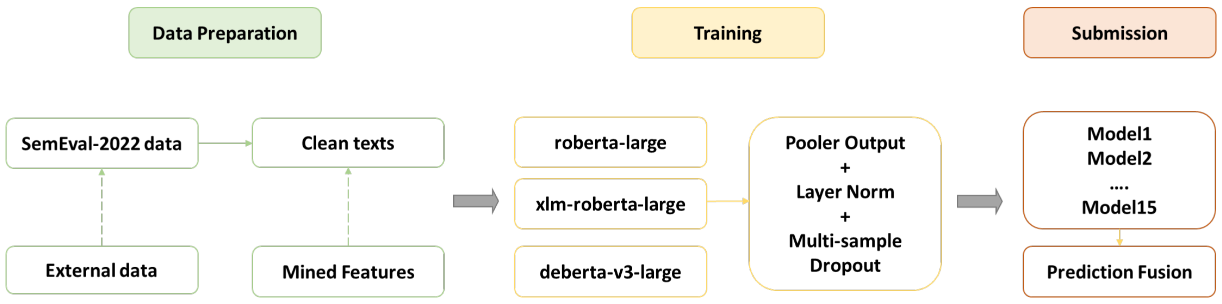


Figure 1: Task experimental progress

2.3 Language Models

We have adopted the RoBERTa-large (Liu et al., 2019) and DeBERTa-v3-large (He et al., 2020b, 2021) as the pretrained models from Hugging Face. We also applied the XLM-RoBERTa-large (Conneau et al., 2019) as a pretrained model for the dataset which includes the Arabic tweets during training.

2.4 Feature Mining

Moreover, we mined different types of statistical and lexical-based features that were previously applied in irony detection. Additional text features can improve the detection of sarcasm in many related tasks (Hernández-Farías et al., 2015; Yaghoobian et al., 2021). All the text features are simply added to the preprocessed tweets using the splitting token "$</s></s>$". Figure 2 demonstrates how we concatenate different features into the original text.

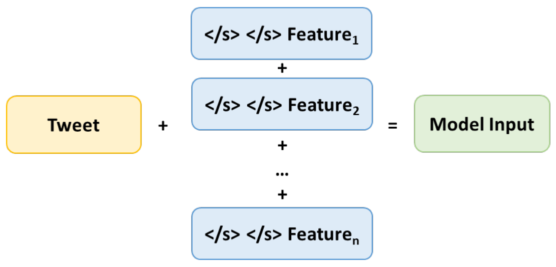


Figure 2: Text and features concatenation

(1) Emoji is a prominent multi-model feature that indicates human emotions after analyzing large amounts of tweet data [4]. On social media, emojis with few characters can easily turn common text into humorous, sarcastic, or ironic expressions.

(2) Parts-of-speech (POS) information is applied as an important feature as well. It is worth mentioning that we mined the POS-based features from the sarcastic tweets and their own rephrases in the official SemEval-2022 dataset. A list of adjectives and adverb words is generated by comparing the differences between the sarcastic tweets and their rephrased versions. For example, some words like "really", "never", "actually", etc. can be considered a symbol of sarcasm and express some contradictory and criticized attitudes.

(3) We also notice that some misspelled words (e.g., "so"->"soooo", "love"->"looove", "sure"->"sureeee") and capitalized words (not located at the beginning of a sentence) can sometimes exaggerate the emotional expression. Those words are detected and added to the tweets as additional text features.

(4) Transitional words and words with negative polarity are also considered as two potential sarcastic features (Tayal et al., 2014). Transitional words can express opposition or contradiction or indicate different meanings in a same sentence, such as "on the other hand" and "neverthelsess" [5]. The polarity of words or lexicon sometimes helps to identify the level of praise or criticism of a text. For the polarity-based feature, we adopted the AFINN dataset [6] (a list of words labelled with a polarity valence) as a reference.

2.5 Ensemble

The final submitted result is fused, utilizing the voting-based mechanism, with the predictions of 15 pretrained models. Voting from different models can usually hinder obvious mis-classifications from a single model (Ruta and Gabrys, 2005; Zhang et al., 2014). Hard voting and soft voting are two classical voting methods in classification tasks.

[4]https://github.com/MathieuCliche/Sarcasm_detector

[5]http://www.csun.edu/~hcpas003/transwords.html;https://wordcounter.net/blog/2016/07/19/101889_transition-words.html

[6]http://github.com/abromberg/sentiment_analysis/blob/master/AFINN/AFINN-111.txt

Hard voting directly fuses different ensembles by picking the highest number of votes. The amount of sarcasm and non-sarcasm votes from different models directly determines the label of a test sample. Soft voting combines all ensembles by adding the probabilities of each prediction and picking the prediction with the highest probability summation. The predicted label would be sarcasm when the mean probability of the sarcasm category is greater than our selected threshold of 0.5. We mixed the hard and soft voting methods for the final submitted prediction. The hard voting method is adopted when the difference between the amount of sarcasm and non-sarcasm is greater than 2. Otherwise, the soft voting method is adopted.

3 Experimental Setup

Four major improvements in F1 score are given by adding the public dataset, multi-sample dropout layer, text features, and tuning the parameters in the contrastive loss function. In this paper, we evaluate different modules and tricks that are adopted in our proposed model to show their effect on the Semeval-2022 official dataset. Table 2 shows the F1 scores for the competition blind test set based on different pretrained models, strategies and datasets.

The DeBERTa-v3-large model outperforms about 5% the other two pretrained models we used in this task. Many other tasks show the outperformance of the DeBERTa model as well. DeBERTa modes proposed two novel tricks to improve the ability to solve many natural language tasks. Compared with the BERT and RoBERTa models, the disentangled attention mechanism is applied to show each word in two vectors, which represent its content and relative position. The disentangled matrices are used to calculate the attention weights between the word content and position. An enhanced mask decoder is adopted to help with model pre-training by using the absolute position to predict the masked tokens. The DeBERTa model also shows a big improvement in how well the model generalizes when the virtual adversarial training method is used.

Multi-sample dropout is a regularization technique which can accelerate training convergence and improve the model generalization compared to the network structure with a traditional dropout layer (Inoue, 2019). Four dropout layers were applied to the pooling layer output from the pretrained model. Table 2 shows that additional multi-sample dropout layer provides a remarkable effect (4% improvement on F1 score) on this classification task.

Moreover, many effective adversarial training and vitual adversarial training methods can improve the model robustness and the regularization on classification tasks (Madry et al., 2017; Miyato et al., 2016; Goodfellow et al., 2014; Zhu et al., 2019; Jiang et al., 2019; Qin et al., 2019; Shafahi et al., 2019). In this task, we adopted Projected Gradient Descent (PGD) and the Fast Gradient Method (FGM) to implement the perturbation on sequence embedding in this task (Madry et al., 2017; Miyato et al., 2016). Table 2 shows a pretty similar F1 score for both methods. The FGM adds small perturbations to the embedding layers to enhance the quality of word embedding. The submitted models are fused by models trained in the fast gradient method, credited with its fast training converges and fewer computation resources.

Additionally, the constrastive loss is considered in our training progress. It maximizes the amount of agreement between different augmented views of the same dataset through adding a contrastive loss in the latent space. (Hadsell et al., 2006; Chen et al., 2020; He et al., 2020a). Many tasks are competitively performed by adding contrastive loss functions, such as triple margin loss, NPair loss, InfoNCE loss, and SupCon loss (Sohn, 2016; Chen et al., 2020; Van den Oord et al., 2018; He et al., 2020a; Khosla et al., 2020). In this task, we considered the NTXent and SupCon losses as the additional contrastive loss. We added the selected contrastive loss to the cross-entropy loss to improve the classification accuracy.

The effect of the contrastive temperature reflects the attention of difficult samples. The smaller temperature pays more attention to the separation of the sample from the most similar one to it (Wang and Liu, 2021). We tuned the contrastive temperature in both NTXent and SupCon loss to train different classification models. NTXent with a contrastive temperature of 0.2 creates the best performance on the competition blind test set according to Table 2.

Furthermore, the weighted voting on multiple models with different pretrained models and random seeds improves final performance on the competition's blind test set. Figure 3 shows the minimum, mean, and maximum F1, precision and recall scores for predictions from 15 single models. The mixed voting method we applied for the final sub-

Dataset	Pre-trained model	Adversarial Training method	Text Features	Multi-sample dropout	Contrastive loss/temp	F1 @ Sarcasm	Macro F1
SemEval2022(EN)	roberta-large	pgd	False	False	NTXent/0.5	0.4125	0.6402
		pgd	False	True	NTXent/0.5	0.4582	0.6794
		fgm	False	True	NTXent/0.5	0.4691	0.6788
		fgm	False	True	SupCon/0.5	0.4788	0.6802
		fgm	False	True	SupCon/0.2	0.4813	0.6833
		fgm	True	True	NTXent/0.2	0.4862	0.6957
	deberta-v3-large	fgm	False	True	NTXent/0.2	0.5370	0.7132
SemEval2022 (EN+AR)	xlm-roberta-large	fgm	False	True	NTXent/0.2	0.3871	0.6420
SemEval2022(EN) + Semeval2018 + iSarcasm	roberta-large	fgm	False	True	NTXent/0.2	0.5570	0.7374
SemEval2022(EN) + Semeval2018 + iSarcasm + multi-model	deberta-v3-large	fgm	False	True	NTXent/0.2	0.5882	0.7445
	xlm-roberta-large	fgm	False	True	NTXent/0.2	0.5615	0.7409
SemEval2022 (EN+AR) + Semeval2018 + iSarcasm + multi-model	xlm-roberta-large	fgm	True	True	NTXent/0.2	0.6029	0.7676

Table 2: F1 scores using different strategies and datasets

mission has obtained the best performance.

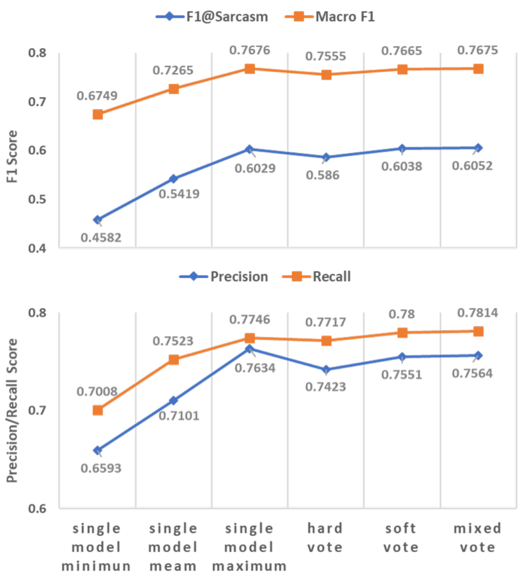


Figure 3: Prediction results from single and fused models

4 Conclusion

According to the performance on the blind test set, the proposed model with the highest F1 score in the sarcastic category applied four layers of multi-sample dropout with a rate of 0.4 following the pooling layer outputting from the XLM-RoBERTa-large model. The model is trained in the fast gradient method using the AdamW optimizer at a learning rate of 1e-05. The combination of the cross-entropy and the NTX contrastive loss is applied during the training process. Additionally, incorporating text features and data from other sources can help improve the prediction's accuracy.

Acknowledgements

This task has been completed with funding from PingAn Life Insurance. All the work stated in this paper was conducted during the Semeval-2022 competition. All the opinions and statements only reflect the authors' views and assertions. Additionally, we wish to express our gratitude to the task organizers and anonymous reviewers for their insightful comments.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Santosh Kumar Bharti, Bakhtyar Vachha, RK Pradhan, Korra Sathya Babu, and Sanjay Kumar Jena. 2016. Sarcastic sentiment detection in tweets streamed in real time: a big data approach. *Digital Communications and Networks*, 2(3):108–121.

Konstantin Buschmeier, Philipp Cimiano, and Roman Klinger. 2014. An impact analysis of features in a classification approach to irony detection in product reviews. In *Proceedings of the 5th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 42–49.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Ruth Filik, Alexandra Ţurcan, Dominic Thompson, Nicole Harvey, Harriet Davies, and Amelia Turner. 2016. Sarcasm and emoticons: Comprehension and emotional impact. *Quarterly Journal of Experimental Psychology*, 69(11):2130–2146.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020a. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020b. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Irazú Hernández-Farías, José-Miguel Benedí, and Paolo Rosso. 2015. Applying basic features from sentiment analysis for automatic irony detection. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 337–344. Springer.

Hiroshi Inoue. 2019. Multi-sample dropout for accelerated training and better generalization. *arXiv preprint arXiv:1905.09788*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2019. Smart: Robust and efficient fine-tuning for pretrained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Avinash Kumar, Vishnu Teja Narapareddy, Veerubhotla Aditya Srikanth, Aruna Malapati, and Lalita Bhanu Murthy Neti. 2020. Sarcasm detection using multi-head attention based bidirectional lstm. *Ieee Access*, 8:6388–6397.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Silviu Oprea and Walid Magdy. 2020. isarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Reynier Ortega-Bueno, Francisco Rangel, D Hernández Farıas, Paolo Rosso, Manuel Montes-y Gómez, and José E Medina Pagola. 2019. Overview of the task on irony detection in spanish variants. In *Proceedings of the Iberian languages evaluation forum (IberLEF 2019), co-located with 34th conference of the Spanish Society for natural language processing (SEPLN 2019). CEUR-WS. org*, volume 2421, pages 229–256.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. *arXiv preprint arXiv:1610.08815*.

Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. 2019. Adversarial robustness through local linearization. *Advances in Neural Information Processing Systems*, 32.

Antonio Reyes and Paolo Rosso. 2014. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems*, 40(3):595–614.

Dymitr Ruta and Bogdan Gabrys. 2005. Classifier selection for majority voting. *Information fusion*, 6(1):63–81.

Samer Muthana Sarsam, Hosam Al-Samarraie, Ahmed Ibrahim Alzahrani, and Bianca Wright. 2020. Sarcasm detection using machine learning algorithms in twitter: A systematic review. *International Journal of Market Research*, 62(5):578–598.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29.

Devendra Kr Tayal, Sumit Yadav, Komal Gupta, Bhawna Rajput, and Kiran Kumari. 2014. Polarity detection of sarcastic political tweets. In *2014 International conference on computing for sustainable global development (INDIACom)*, pages 625–628. IEEE.

Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Palak Verma, Neha Shukla, and AP Shukla. 2021. Techniques of sarcasm detection: A review. In *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 968–972. IEEE.

Feng Wang and Huaping Liu. 2021. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2495–2504.

Hamed Yaghoobian, Hamid R Arabnia, and Khaled Rasheed. 2021. Sarcasm detection: A comparative study. *arXiv preprint arXiv:2107.02276*.

Shiwei Zhang, Xiuzhen Zhang, Jeffrey Chan, and Paolo Rosso. 2019. Irony detection via sentiment-based transfer learning. *Information Processing & Management*, 56(5):1633–1644.

Yong Zhang, Hongrui Zhang, Jing Cai, and Binbin Yang. 2014. A weighted voting classifier based on differential evolution. In *Abstract and Applied Analysis*, volume 2014. Hindawi.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*.

GetSmartMSEC at SemEval-2022 Task 6: Sarcasm Detection using Contextual Word Embedding with Gaussian Model for Irony Type Identification

Diksha Krishnan and **C.Jerin Mahibha**
Meenakshi Sundararajan Engineering College, Chennai
dikshakrish5@gmail.com
jerinmahibha@gmail.com

Durairaj Thenmozhi
Sri Sivasubramaniya Nadar College of Engineering , Chennai
theni_d@ssn.edu.in

Abstract

Sarcasm refers to the use of words that have different literal and intended meanings. It represents the usage of words that are opposite of what is literally said, especially in order to insult, mock, criticise or irritate someone. These types of statements may be funny or amusing to others but may hurt or annoy the person towards whom it is intended. Identification of sarcastic phrases from social media posts finds its application in different domains like sentiment analysis, opinion mining, author profiling and harassment detection. We have proposed a model for the shared task iSarcasmEval - Intended Sarcasm Detection in English and Arabic by SemEval-2022 considering the language English. The Subtask A and Subtask C were implemented using a Convolutional Neural Network based classifier which makes use of ELMo embeddings. The Subtask B was implemented using Gaussian Naive Bayes classifier by extracting TF-IDF vectors. The proposed models resulted in macro-F1 scores of 0.2012, 0.0387 and 0.2794 for sarcastic texts in Subtasks A, B and C respectively.

1 Introduction

In the Internet era, specifying user comments, views and opinions through social media has become very common and these may not be specified directly and can include indirect phrases with implicit meanings (Abu Farha and Magdy, 2020). The posts may also represent undesirable characteristics using positive words and they may not be formal in nature. It is common to use abbreviations, uncommon, ambiguous and multilingual words in social media posts and no predefined structures are explicitly defined for sarcastic messages (Sarsam et al., 2020). So the identification of sarcastic posts cannot be considered as a direct process. Exhaustive understanding of the contextual meaning of the posts is considered as an important factor in identifying sarcastic posts. Sarcasm detection is considered as a challenging task associated with sentiment analysis. Sarcasm detection also plays an important role in analysing the voice of the customer based on which major decisions will be taken.

Sarcasm requires some shared knowledge between speaker and audience and it is considered as a profoundly contextual phenomenon. Using sarcastic phrases in sentences is a common way of ironic or satirical speech for the common man. That being said, social media platforms, such as Twitter, Facebook and YouTube contain millions of tweets and comments that include sarcastic phrases, thus making it a field of study under the domain of NLP. For example, a user can change a supposedly negative comment using positive words as in the sentence, *"It is awesome to go to bed at 3 am #not"*. In such cases it becomes important to ensure that the right sentiment is drawn out of the sentence through proper analysis (Vu et al., 2018).

The shared task iSarcasmEval was part of SemEval 2022 (Abu Farha et al., 2022) and there were three subtasks associated with it. Two of the subtasks - Subtask A and Subtask B, were conducted for both English and Arabic languages, while Subtask C only pertains to English. We as a team participated in all the three subtasks associated with the language English.

Subtask A: Sarcastic vs. Non-Sarcastic

The first subtask was a straightforward Binary Classification problem, in which the model had to predict whether a given phrase is sarcastic or not. For example, *"I work 40 hours a week for me to be this poor." is sarcastic, whereas "Her husband is serving a three-year sentence for fraud."* is non-sarcastic.

Subtask B: Different types of irony

The second subtask was a Multi-Label Classification problem, where the aim was to identify which type of irony a given sentence falls under from six specific labels: sarcasm, irony, satire, understatement, overstatement and rhetorical question.

Subtask C: Differentiation between sarcastic

phrase and its rephrase

The final subtask is another Binary Classification problem, wherein, given a sarcastic phrase and its rephrase, the model needs to identify which of the two is sarcastic. For example, the phrase *"Taxes are just the best and I cannot wait to pay more"* is sarcastic, whereas its rephrase - *"I dislike paying taxes.*" is non-sarcastic (Goutte et al., 2014).

Considering all the three subtasks the training of the proposed model was done using the training data set provided for the corresponding task and language. This model was then tested with a testing data set provided by the shared task, based on which the task was evaluated.

2 Related Works

Sarcasm detection can be posed as a text classification task in which the given text needs to be identified as sarcastic or not. In simple systems it could be considered as a binary classification task where the presence of sarcasm is only detected. But if the irony associated with the sarcastic message needs to be identified then it is considered as a multi label classification task. Different machine learning and deep learning models could be used for identifying sarcasm in text and it is not evident that a particular algorithm provides the best result for any data. The features of the data set like the number of instances in the data set, distribution of data in the training data set are important factors on which the performance of the algorithm relies. So it becomes necessary to analyse the data set, to choose the model for implementing the classification task.

It had been shown that Support Vector Machine provided the best performance for sarcasm detection considering posts from twitter (Sarsam et al., 2020). A combination of Convolutional Neural Network (CNN) and SVM had also offered high prediction accuracy. Even though feature rich SVM model perform well, Avinash Kumar and et al. (Kumar et al., 2020) had shown better performance by combining multi-head attention mechanism with bidirectional long-short memory (BiLSTM) to detect sarcastic messages.

Use of deep learning models with BiLSTM to detect sarcasm from Arabic texts had been illustrated by (Abu Farha and Magdy, 2020). A sarcasm detector had been implemented by fine tuning BERT and considering both affective and contextual features (Babanejad et al., 2020). The features of sarcastic text had been captured using a multi-level memory network to take care of sentiment semantics and the contrast between it and the situation in each sentence (Ren et al., 2020). Different transformer-based language models were used for sarcasm detection in Arabic language text and had found that MARBERT and AraELECTRA performed well and AraGPT2 had shown poor performance (Abu Farha and Magdy, 2021).

Pre-trained transformer models when ensembled with Recurrent Convolutional Neural Network to form hybrid neural architecture to detect sarcastic messages had resulted in better performance when compared to other relevant state-of-the-art methodologies (Potamias et al., 2020). A hybrid model implemented using bidirectional long short-term memory with a softmax attention layer and convolution neural network when used for sarcasm detection had resulted in better performance (Jain et al., 2020). Sarcasm detection had been implemented by taking into account the contextual information using a dual channel Convolutional Neural Network and the user's expression habit had been identified using attention mechanisms (Du et al., 2022).

In addition to the above approaches, various novel approaches like statistical approaches, graph based approaches, fuzzy logic based approaches and pseudo labelling approaches had been used for identifying sarcastic messages. A complex-valued fuzzy network had been used to identify the text with sarcasm by leveraging the mathematical formalisms of quantum theory and fuzzy logic (Zhang et al., 2021). Sarcasm detection had been implemented by taking the contextual information of a sentence into account in a sequential manner using the concept of pseudo-labeling (Kumar Jena et al., 2020), (Kalaivani and Thenmozhi, 2020). Long-range literal sentiment inconsistencies had been taken into account in sarcasm detection by constructing an affective graph and a dependency graph for each sentence and had then used an Affective Dependency Graph Convolutional Network (ADGCN) framework for the classification process (Lou et al., 2021). Statistical approach had been proposed for sarcasm detection by combining TF-IDF features with the important features related to sentiments and punctuations that are identified using chi-square test (Gupta et al., 2020).

Social media posts may have both text and images associated with it and identifying sarcasm in

Task	Category	Instances
Subtask A	Sarcastic	867
	Non Sarcastic	2601
Subtask B	Sarcasm	713
	Irony	155
	Satire	25
	Under Statement	10
	Over Statement	40
	Rhetorical question	101
Subtask C	Rephrase	867

Table 1: Data Distribution

such case had been implemented using a multimodal framework using Coupled-Attention Networks (CANs) which captures and integrates information from both text and image for the classification task (Zhao et al., 2021). As the imbalanced nature of the data set affects the performance of the model oversampling had been done to convert the data set into a balanced one and had shown that SMOTE and BorderlineSMOTE–1 techniques when used for oversampling had resulted in the improvement of performance (Banerjee et al., 2020). It could be summarized from the related works that identifying sarcasm from social media posts is an emerging research area that requires more insights. Different techniques like traditional machine learning models such as SVM and Random Forest, Convolutional Neural Network based models, Transformer models and Ensemble models could be used for this purpose. It could be found that the performance of the approach depends on the dataset on which the model is being trained (Abdulraheem et al., 2015). It is hard to identify a particular approach that can detect sarcastic messages under any circumstances which provides the best performance.

3 Data set

The data set that we used to implement sarcasm detection was the training and the test dataset that was provided by the organisers of the shared task. Each instance of the training dataset had the following informations attached to it:

1. Label specifying the sarcastic nature of the text

2. Rephrase text that convey the same message of the text non-sarcastically

3. Label specifying the category of ironic speech which includes sarcasm, irony, satire, understatement, overstatement and rhetorical question

There were 3468 instances in the training data set of which 867 instances were under the sarcastic category and remaining 2601 instances were under the non sarcastic category. This shows the unbalanced nature of the data set. The test data had 1400 instances for which the predictions had to be done using the proposed model. The distribution of the data in the training dataset is shown in Table 1. As separate data set was not provided for the evaluation purpose, under both category 80% of the training data instances were used for training purpose and 20% of the instances from the training data set was used for the evaluation purpose.

4 System Description

The proposed methodology uses ELMo embedding based Convolutional Neural Network model for implementing the Subtasks A and C with an embedding layer followed by two dense layers. The Subtask B has been implemented using TF-IDF based Gaussian Naive Bayes classifier.

4.1 ELMo Model

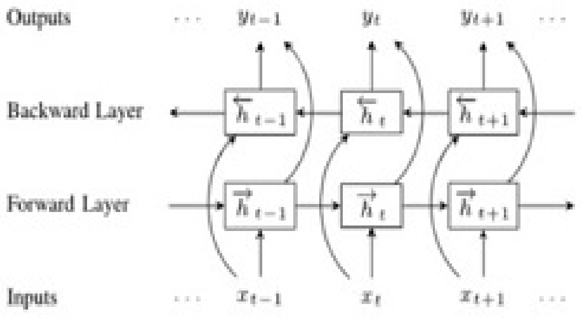

Outputs
$\cdots$ y_{t-1} y_t y_{t+1} $\cdots$
Backward Layer
$\overleftarrow{h}_{t-1}$ $\overleftarrow{h}_t$ $\overleftarrow{h}_{t+1}$
Forward Layer
$\overrightarrow{h}_{t-1}$ $\overrightarrow{h}_t$ $\overrightarrow{h}_{t+1}$
Inputs
$\cdots$ x_{t-1} x_t x_{t+1} $\cdots$

Figure 1: Architecture of ELMo Model

It stands for Embeddings from Language Models and is a novel way to represent words in vectors or embeddings (Peters et al., 2018). In ELMo the syntax and semantics of the word and the linguistic context associated with it are modelled as a deep contextualised word representation. Huge text corpus had been used to pretrain the model and is constructed using deep bidirectional models[1]. It is implemented with 4096 units and the input embedding transform using 2048 convolutional filters.

[1]https://allenai.org/allennlp/software/elmo

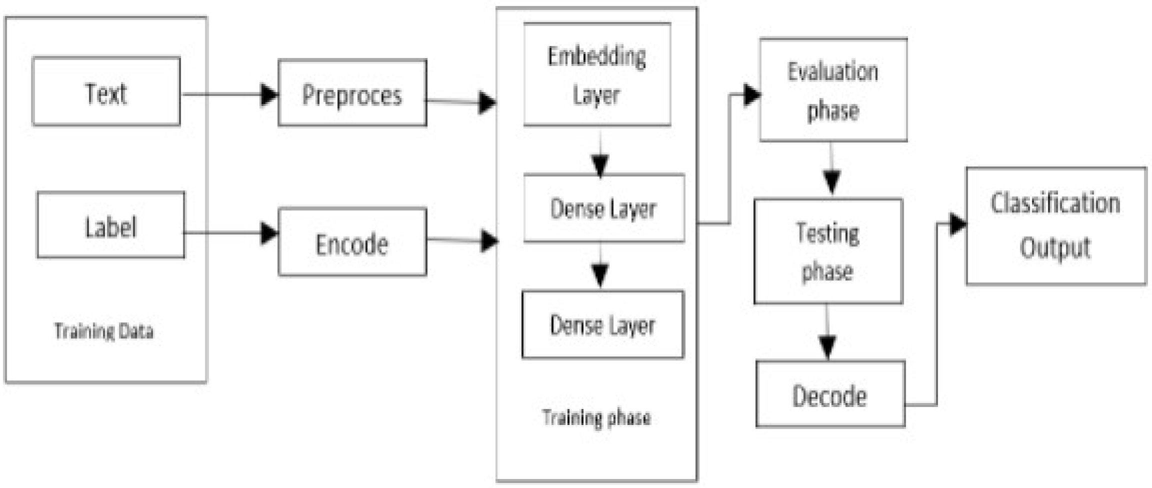


Figure 2: Proposed Architecture

The context of the word usage forms the base for the word representation. ELMo word representations take the entire input sentence into equation for calculating the word embeddings. The architecture of ELMo model[2] is shown in Figure 1.

ELMo is a bidirectional language model. The forward LM is a deep LSTM that goes over the sequence from start to end to predict the token and the backward LM is a deep LSTM that goes over the sequence from end to start to predict the token.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Figure 3: Likelihood of Gaussian Naive Bayes

4.2 Gaussian Naive Bayes Classifier

It follows Gaussian Normal Distribution and is a variant of Naive Bayes which are a group of supervised machine learning classification algorithms based on the Bayes theorem. The concept of conditional probability is used while using this for the classification problem[3]. The likelihood of the features in Gaussian Naive Bayes Classifier is represented by Figure 3. Predictions using this classifier is done by providing new input values for the parameters which will result in an estimated probability by the Gaussian function.

The task of determining whether the given text is sarcastic or non-sarcastic represents a binary classification problem, which is Subtask A. The task of determining the sarcastic one from two texts that convey the same meaning also represents a text classification problem which is the problem statement associated with Subtask C. The proposed model uses an ELMo model to implement the above two tasks. Subtask B is a binary multi-label classification task in which the correct ironic speech category has to be identified from the given set of labels. Figure 2 shows the architecture of the proposed model.

The first step associated with all the subtasks is to prepare the data set, which involves preprocessing of the text within. This is carried out by removing any escape sequences and stop words associated with the text, generating the associated tokens and lemmatizing the same. The idea behind preprocessing is to remove the parts from the text which do not contribute to the actual intent of the text.

For implementing Subtask A and C which were binary classification problem, the labels were normalized by using a one hot encoding scheme, which transformed the labels into a categorical value for which embedding is done and the encoded labels are returned[4]. The encoded data is used to train the model which is a Convolutional Neural Network with an embedding layer followed by two dense layers. The embedding layer gener-

[2]https://andy-nguyen.medium.com/create-a-strong-text-classification-with-the-help-from-elmo-e90809ba29da

[3]https://jakevdp.github.io/PythonDataScienceHandbook

[4]https://scikit-learn.org/stable/modules/generated

Metric	Score
F1-Sarcastic	0.2012
F-Score	0.5101
Precision	0.5137
Recall	0.5196
Accuracy	0.705

Table 2: Subtask A Scores

ates the ELMo embeddings which is implemented with the help of tensorflow hub. The first dense layer has 256 units and makes use of relu activation function. The output is generated from the second dense layer with two units as it performs Binary Classification. The activation function used by the last layer is softmax. As Subtask B involved multi label classification, TF-IDF vectors were generated for the input text and the classification was carried out using a Gaussian Naive Bayes classifier[5], which uses the concept of Bayes theorem.

The training data set provided as a part of the shared task was used for training the model. As a separate data set was not provided for validation, 20% of the training instances were selected and used for the evaluation process. Finally the testing phase was implemented using the testing data set provided for the shared task.

5 Results

The metrics that were considered for the evaluation of all the three subtasks were precision, recall, accuracy and macro-F1 score. Precision represents the ratio of the number of correct positive results to the number of positive results predicted by the classifier. The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples are detected. Classification accuracy is the ratio of number of correct predictions to the total number of input samples. F1 score is an overall measure of a model's accuracy that combines precision and recall. A high F1 score means that the classification has resulted with low number of false positives and low false negatives. The proposed model resulted in an F1 score for sarcastic texts as 0.2012 based on which Subtask A was evaluated and we were ranked 36 on the leaderboard. Table 2 shows the values that were obtained for various metrics like precision, accuracy and recall considering Subtask A.

Subtask B was evaluated based on the macro-F1 score of the model and the proposed model had resulted in a macro-F1 score of 0.0387. We ranked 19 on the leaderboard under this category. Table 3 shows the F1 scores that were obtained for different type of sarcastic texts like Irony, Satire, Understatement, Overstatement and Rhetorical Question.

Metric	Score
Macro F	0.0387
F1-Sarcasm	0.2321
F1-Irony	0.0000
F1-Satire	0.0000
F1-Understatement	0.0000
F1-Overstatement	0.0000
F1-Rhetorical Question	0.0000

Table 3: Subtask B Scores

Metric	Score
Accuracy	0.3400
F-Score	0.2794

Table 4: Subtask C Scores

Accuracy was the metric that was used for the evaluation of Subtask C, and our model achieved an accuracy of 0.34 with rank 15 on the leaderboard. The F1 score that we obtained for this subtask was 0.2794 and these are tabulated in Table 4.

6 Conclusions

Sarcasm detection has become an important area of research as it is interlinked with different areas of application that includes sentiment analysis, opinion mining, offensive and hate speech detection. Having this in mind SemEval-2022 had come up with the task of Sarcasm detection which was represented by three subtasks namely sarcasm detection, identifying the type of irony associated with the sarcastic text and identifying whether the text or its rephrase is sarcastic. We have applied ELMo embedding based Convolutional Neural Network model for implementing the binary Subtasks A and C. Gaussian Naive Bayes classifier based on TF-IDF vectors was used to implement Subtask B. All the three subtasks were implemented considering the language English. The performance of the models used, were not up to the mark and it is found from the task overview that the transformer models when applied over this tasks provides better results.

Dataset for sarcasm detection could be created with contextual information which can help in ef-

[5]https://scikit-learn.org/stable/modules/naive$_{b}ayes.html$

fectively detecting sarcasm. Usage of hybrid approaches where different machine learning and deep learning models are combined can also facilitate efficient detection of sarcasm from text. Often it could be observed that sarcasm is not in the text, but could be detected from the intonation or facial expression, which has made mulitmodal sarcasm detection also as a promising research area.

References

Ajiboye Abdulraheem, Ruzaini Abdullah Arshah, and Hongwu Qin. 2015. Evaluating the effect of dataset size on predictive model using supervised learning technique. *International Journal of Software Engineering Computer Sciences (IJSECS)*, 1:75–84.

Ibrahim Abu Farha and Walid Magdy. 2020. From Arabic sentiment analysis to sarcasm detection: The ArSarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 32–39, Marseille, France. European Language Resource Association.

Ibrahim Abu Farha and Walid Magdy. 2021. Benchmarking transformer-based language models for Arabic sentiment and sarcasm detection. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 21–31, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Nastaran Babanejad, Heidar Davoudi, Aijun An, and Manos Papagelis. 2020. Affective and contextual embedding for sarcasm detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 225–243, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Arghasree Banerjee, Mayukh Bhattacharjee, Kushankur Ghosh, and Sankhadeep Chatterjee. 2020. Synthetic minority oversampling in addressing imbalanced sarcasm detection in social media. *Multimedia Tools and Applications*, 79(47):35995–36031.

Yu Du, Tong Li, Muhammad Salman Pathan, Hailay Kidu Teklehaimanot, and Zhen Yang. 2022. An effective sarcasm detection approach based on sentimental context and individual expression habits. *Cognitive Computation*, 14(1):78–90.

Cyril Goutte, Michel Simard, and Marine Carpuat. 2014. CNRC-TMT: Second language writing assistant system description. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 192–197, Dublin, Ireland. Association for Computational Linguistics.

Rahul Gupta, Jitendra Kumar, Harsh Agrawal, and Kunal. 2020. A statistical approach for sarcasm detection using twitter data. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 633–638.

Deepak Jain, Akshi Kumar, and Geetanjali Garg. 2020. Sarcasm detection in mash-up language using softattention based bi-directional lstm and feature-rich cnn. *Applied Soft Computing*, 91:106198.

A Kalaivani and D Thenmozhi. 2020. Sarcasm identification and detection in conversion context using bert. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 72–76.

Avinash Kumar, Vishnu Teja Narapareddy, Veerubhotla Aditya Srikanth, Aruna Malapati, and Lalita Bhanu Murthy Neti. 2020. Sarcasm detection using multi-head attention based bidirectional lstm. *IEEE Access*, 8:6388–6397.

Amit Kumar Jena, Aman Sinha, and Rohit Agarwal. 2020. C-net: Contextual network for sarcasm detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 61–66, Online. Association for Computational Linguistics.

Chenwei Lou, Bin Liang, Lin Gui, Yulan He, Yixue Dang, and Ruifeng Xu. 2021. *Affective Dependency Graph for Sarcasm Detection*, page 1844–1849. Association for Computing Machinery, New York, NY, USA.

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

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas . Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Lu Ren, Bo Xu, Hongfei Lin, Xikai Liu, and Liang Yang. 2020. Sarcasm detection with sentiment semantics enhanced multi-level memory network. *Neurocomputing*, 401:320–326.

Samer Muthana Sarsam, Hosam Al-Samarraie, Ahmed Ibrahim Alzahrani, and Bianca Wright. 2020. Sarcasm detection using machine learning algorithms in twitter: A systematic review. *International Journal of Market Research*, 62(5):578–598.

Thanh Vu, Dat Quoc Nguyen, Xuan-Son Vu, Dai Quoc Nguyen, Michael Catt, and Michael Trenell. 2018. Nihrio at semeval-2018 task 3: A simple and accurate neural network model for irony detection in twitter.

Yazhou Zhang, Yaochen Liu, Qiuchi Li, Prayag Tiwari, Benyou Wang, Yuhua Li, Hari Mohan Pandey, Peng Zhang, and Dawei Song. 2021. Cfn: A complex-valued fuzzy network for sarcasm detection in conversations. *IEEE Transactions on Fuzzy Systems*, 29(12):3696–3710.

Xuan Zhao, Jimmy Huang, and Haitian Yang. 2021. Cans: Coupled-attention networks for sarcasm detection on social media. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Amrita_CEN at SemEval-2022 Task 6: A Machine Learning Approach for Detecting Intended Sarcasm using Oversampling

Aparna K Ajayan, Krishna Mohanan, Anugraha S, Premjith B, and Soman K.P
Centre for Computation Engineering and Networking (CEN)
Amrita School of Engineering, Amrita Vishwa Vidyapeetham, India
b_premjith@cb.amrita.edu

Abstract

This paper describes the submission of the team Amrita_CEN to the shared task on iSarcasm Eval: Intended Sarcasm Detection in English and Arabic at SemEval 2022. The sarcasm detection task was formulated as a classification problem and modelled using machine learning classifiers. We used K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Naïve Bayes, Logistic Regression, Decision Tree, and the Random Forest ensemble method. In addition, the class imbalance problem in the dataset was addressed using a feature engineering technique. We submitted the predictions by SVM, Logistic Regression and Random Forest ensemble based on the performance during training.

1 Introduction

Sarcasm is an ironic form showing a disparity between the actual and intended meaning of the text affecting the decision-making process. These are reflected in our day-to-day communication with each other happening in social media forums. Twitter exhibits rich sarcasm phenomena, thereby encouraging automatic sarcasm detection methods and removing such tweet data. Due to the sociocultural aspects of sarcastic communication, the majority of the sarcasm detection work has been focused only on the English language (Oprea and Magdy, 2020b), and only a limited amount of work was done in other languages such as Arabic (El Mahdaouy et al., 2021).

Identification of sarcastic comments from social media contexts is essential since the author and the receiver are at various places. Therefore, exchanging conversations may sometimes lead to a negative meaning of the text that even the author has not meant to convey. Moreover, the data stream for sarcasm does not exhibit any static structure like specific tags in the form of #sarcasm, and #irony (Ptáček et al., 2014) (Khodak et al., 2018). This event can lead to noisy labels due to several reasons, as outlined by (Oprea and Magdy, 2020b). Other works reported on the topic mainly depend on manual labelling, provided with manually annotated sarcasm labels. In (Oprea and Magdy, 2020b) the authors pointed out that manual labelling represents the author annotation in contrast with the intention of the authors.

The sarcasm prediction on Twitter that influences Machine Intelligence is a challenging task (Khare et al., 2022). It can be achieved with the help of the Natural Language Processing (NLP) approach, and many recent works on automatic sarcasm detection have focused on Twitter data as it primarily requires an understanding of the human expressions, language, and emotions expressed via textual or non-textual content (Kumar et al., 2021). Therefore, the goal of the SemEval shared task is to facilitate the development of machine learning models that can detect sarcasm from tweets. The shared task consists of two subtasks:

- Subtask A: For a given text, determine whether it is sarcastic or non-sarcastic.
- Subtask B (English only): A binary multi-label classification task for a given a text, determine which ironic speech category it belongs.

In this paper, we describe the machine learning models designed for solving the problems given in iSarcasm shared tasks (Abu Farha et al., 2022). The performance of the models was evaluated using the F1-score. The models submitted achieved the following scores: 0.4966 in English, 0.6127 in Arabic and 0.0567 F1-score in subtasks A and B, respectively.

2 Literature Review

The majority of the published works developed for the text sarcasm detection used datasets that were annotated using a weak supervision method, where the texts were regarded as sarcastic only if they met

preset criteria, including specific tags like sarcasm and irony (Oprea and Magdy, 2020a) (Ptáček et al., 2014) (Khodak et al., 2018). In (Oprea and Magdy, 2020b), S.V Opera and W Magdy reported that labelling using a weak supervision method could lead to noisy labels. Other works on this topic were based on manual labelling, where the human annotators are given the role of labelling the texts (Filatova, 2012) (Riloff et al., 2013) (Abercrombie and Hovy, 2016). The disadvantage of such a labelling procedure is that it represents the perception of the annotator, which may differ from the author's intention (Oprea and Magdy, 2020b).
In addition to the above-mentioned method, a significant majority of works on sarcasm detection were centered exclusively on the English language (Oprea and Magdy, 2019) (Campbell and Katz, 2012) (Riloff et al., 2013) (Joshi et al., 2016) (Amir et al., 2016) (Rajadesingan et al., 2015) (Bamman and Smith, 2015). It is because of its sociocultural aspects on sarcastic communication (Oprea and Magdy, 2020b), leading to the uncertainty that, the models trained on English could generalize to other languages. All the reported works on sarcasm detection in other languages such as Arabic (Karoui et al., 2017) (Ghanem et al., 2019) (Abbes et al., 2020) (Farha and Magdy, 2020) were relied on the afore-mentioned labelling techniques.

3 Dataset and Task Description

The dataset comprises tweets in English and Arabic. There are two subtasks in English and one in Arabic. Tweets in the English dataset were categorized into two: Sarcastic and Non-sarcastic. It contains 3,467 instances of tweets and ten columns containing the attributes (id, tweet, sarcastic, rephrase, sarcasm, irony, satire, understatement, overstatement, rhetorical question). The objective of task-1 is to determine whether a given text is sarcastic or not. Task-2 is a multi-label classification that aims to classify a tweet into different ironic speech categories, such as Sarcasm, Irony, Satire, Understatement, Overstatement, and Rhetorical questions. The shared task-1 in Arabic focused on categorizing a tweet into sarcastic or non-sarcastic, similar to task-1 in English. The Arabic dataset contains 2,601 instances of tweets and five attributes (id, tweet, sarcastic, rephrase, dialect). Table 1 describes the datasets used for task-1 and task-2 in English and task-1 in Arabic.

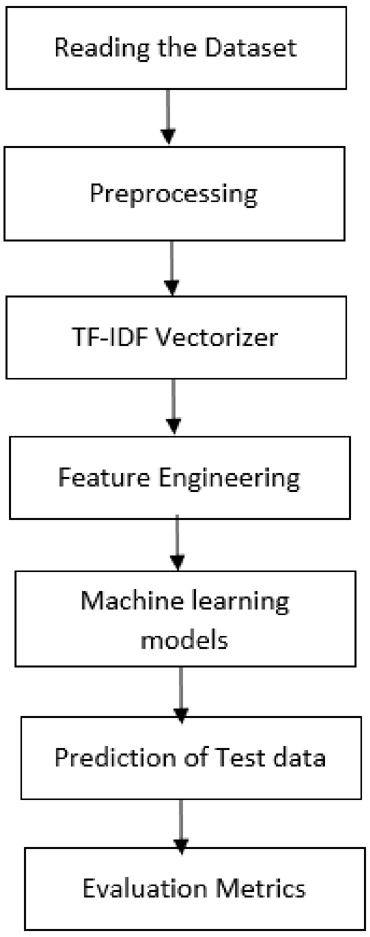


Figure 1: Workflow of the Model

4 System Overview

This section discusses the overview of the models submitted to the shared tasks. The flow of the model building is illustrated in Figure 1.

4.1 Data preprocessing

The shared task was provided with two kinds of input

(a) *Task-1*: text file contain tweets provided with its label, rephrased form and also the irony of the same for both English and Arabic language.

(b) *Task-2*: English text file considered for task 1 is used for irony identification in csv format.

The "Tweet" column from the datasets (Tasks 1 and 2) contains tweets, which must be preprocessed before extracting features for model creation. The preprocessing steps include tokenization, lemmatization, stop word removal and represented tweets

Dataset properties	Task-1 English	Task-2 English	Task-1 Arabic
No. of rows	3,467	3,467	2,601
No. of classes	2	6	2
No. of words	22,623	22,623	38,885
Vocabulary size	5,509	5,509	16,226
Maximum tweet length	72	72	31

Table 1: Description of the dataset used for Task-1 and Taks-2 in English and Task-1 in Arabic

as vectors using the Term Frequency-Inverse Document Frequency (TF-IDF) algorithm (HB et al., 2016).The preprocessing of the tweets was carried out by using the functions available in the NLTK][1] library, whereas the sklearn TfidfVectorizer() [2] helps to vectorize the tweets.

4.1.1 Tokenization

Tokenization is the first step that we executed in preprocessing. Here, the tweet from the user is split into tokens for the ease of feature extraction.

4.1.2 Lemmatization

Lemmatization refers to correctly identifying the base form of a word and converting it into the meaningful base form considering the context.

4.1.3 Stopword removal

Stop word removal is performed to remove the most commonly occurring words in the tweet, such as pronouns and articles. A similar operation was performed on Arabic data by collecting a publicly available stopword list.

4.2 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF is a feature extraction method for vectorizing a sentence or tweet. The TF-IDF vector can be obtained for a sentence by computing Equation 1 for each word in that sentence.

$$TF-IDF(t,D) = TF(t,D) \times IDF(t) \quad (1)$$

Where the Term Frequency

$$TF(t) = \frac{N(t)}{T} \quad (2)$$

and Inverse Document Frequency

$$IDF(t) = log\frac{n}{df(t)} \quad (3)$$

where,
t is the word in a tweet, $N(t)$ is the number of times word t occurs in a document, T is the number of words in a document, n is the total number of sentences/tweets in the dataset, and $df(t)$ is the number of documents in which the term t appears.

4.3 SMOTE

SMOTE (Synthetic Minority Oversampling Technique) (Chawla et al., 2002) is an oversampling method for solving the class imbalance problem in the dataset. It resolves the problem by increasing the number of data points in the minority class with synthetically generated random data points. It is achieved by randomly selecting one or more k nearest neighbours of each minority class. The process can be initiated using the following steps:

1. Given the minority class S, for each $y \in S$, the nearest k-neighbours of y are obtained using Euclidean distance of y and every other elements in S.

2. Sampling rate T is given according to the proportion of imbalance. For each $y \in S$, T elements are selected randomly from nearest k-neighbours. And the set S_1 is made.

3. For every $y_k \in S_1$, $k = 1, 2, 3..., T$, the formula for generating new example (y')is,

$$y' = y + rand(0,1) * |y - y_k| \quad (4)$$

The SMOTE algorithm was implemented using the SMOTE function available in the imblearn Python package[3].

4.4 Model development

We utilized K-Nearest neighbour (KNN) (Guo et al., 2003), Support Vector Machine (SVM) (Soman et al., 2009), Naïve Bayes (Huang and Li,

[1]https://www.nltk.org/
[2]https://scikit-learn.org/stable/modules/generated/sklear.feature_extraction.text.TfidfVectorizer.html

[3]https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html

2011), Decision Tree (Priyam et al., 2013) and Random Forest (Premjith et al., 2019) ensemble method for developing the models for various subtasks in iSarcasm. The procedures for model development for different tasks are given in the following subsections.

4.4.1 Sub task1: Sarcasm Identification

We built a binary classifier to determine whether the given tweet is sarcastic or not. Therefore, for the same purpose, we applied machine learning classifiers to the processed train data. For English tweet data, encouraging results were obtained by Decision tree and logistic regression. The decision tree is a particular type of probability tree that makes the decision about the process (Rahaman et al., 2021), and Logistic Regression is used for predicting the categorical dependent variable using a given set of independent variables (Sarsam et al., 2020). SVM and Random forest classifiers obtained the best performance for Arabic data. The Random Forest classifier reduces the bias due to overfitting and class imbalance between tweets. Bouazizi and Ohtsuki (Bouazizi and Ohtsuki, 2016) used logistic regression to label the data as sarcastic or non-sarcastic.

4.4.2 Sub task2: ironic speech category Identification

A multi-label classifier was developed for this task to determine the ironic speech category of the tweets. We applied a multi labelled classifier strategy with fitting one classifier per target, allowing multiple target variable classifications. The primary purpose behind this class is to extend estimators enabling estimation of a series of target functions mentioned in the dataset, which are trained using a single predictor matrix to predict a series of responses. We implemented a classification model using Logistic Regression, and a decision tree for the same as mentioned above (Sarsam et al., 2020)-(Rahaman et al., 2021).

4.5 Evaluation Metrics

The trained models were evaluated using macro F1-score, Precision, Recall and Accuracy. Accuracy is given by the ratio of the total number of correct predictions to the measure of total predictions done by the model, regardless of correct or incorrect predictions. Precision defines the actual positive among the predicted positive. The recall is a measure of the correctly classified total number of positives. Moreover, F1-score is the harmonic mean of precision and recall. Macro-average is defined as the average of precision, recall, and F1-score in different classes.

5 Experimental Setup

We implemented the models using Python version 3. The training data is split into train and validation sets for confirming the best performing model. In the Arabic sarcasm identification model using the SVM classifier (subtask 1), we used a range of gamma values $(0.1, 1, 10, 100)$ and c regularization parameter values $(0.1, 1, 10, 100)$ and changed the kernel type to RBF, linear and polynomial to see how the accuracy and F1-score vary. In random forest classifier different, n_estimators value $(10, 100, 1000)$ and the maximum features are given to $sqrt$, $log2$ to see the changes (Premjith and Kp, 2020). The English tweet irony detection model (subtask 2) is a multi-class classification problem and implemented using a multioutput classifier set to multilabel.

The model performance was analyzed using macro F1-score obtained using the sklearn metrics along with the accuracy, precision and recall (Pedregosa et al., 2011) of the trained model.

6 Result

All the subtasks were evaluated on the macro-average F1-scores of each information unit. We fixed the best performing models by using cross-validation. The Random Forest classifier and SVM obtained the best F1-scores for English task 1 and Arabic, respectively. In subtask 2, Logistic Regression gave the higher F1-score. We were officially ranked 23rd in task1 English with an F1-score of 0.4966 and accuracy of 56.71% using the Random Forest classifier and ranked 20th in Arabic with 0.6127 of F1-score and 79.21% accuracy using SVM binary classifier. In subtask 2, we were ranked 14th with a macro F1-score of 0.0567 using the Logistic Regression model. The obtained result from our model among all participating teams are shown in table 2, 3 below.

7 Conclusion

This paper presents the submission of Amrita_CEN towards the SemEval 2022 Task 6 competition named " iSarcasmEval - Intended Sarcasm Detection in English and Arabic ". A total of six machine learning algorithms were used, including five

Metrics	Task-1 English	Task-1 Arabic
F1-Sarcastic	0.3052	0.3490
F1-score	0.4966	0.6127
Precision	0.5550	0.6050
Recall	0.6121	0.6246
Accuracy	0.5671	0.7921

Table 2: Result for Subtask 1 English and Arabic

Metrics	Task-2 English
Macro-average F-score	0.0567
F1-score Sarcasm	0.2180
F1-score irony	0.0293
F1-score satire	0.0461
F1-score understatement	0.0074
F1-score overstatement	0.0245
F1-score rhetorical question	0.0150

Table 3: Result for Subtask 2 English

classical ML models and one ensemble technique. The class imbalance problems were dealt with by oversampling technique called SMOTE, and for evaluation, macro F1-score were considered for both the subtasks. The model trained using Random forest, SVM and logistic regression performed well among the subtasks given, and the results were submitted using the same.

8 Acknowledgements

We wish to express our sincere gratitude to our facualty Dr Sowmya V for her help throughout this work and the research underlying it. Your guidance, willingness to share the vast knowledge and expertise made us to understand this work and its manifestations in great depth and helped us to complete the assigned task on time.

References

Intissar Abbes, Yousra Hallem, and Nadia Taga. 2020. Second-hand shopping and brand loyalty: The role of online collaborative redistribution platforms. *Journal of Retailing and consumer Services*, 52:101885.

Gavin Abercrombie and Dirk Hovy. 2016. Putting sarcasm detection into context: The effects of class imbalance and manual labelling on supervised machine classification of twitter conversations. In *Proceedings of the ACL 2016 student research workshop*, pages 107–113.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mário J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 167–177, Berlin, Germany. Association for Computational Linguistics.

David Bamman and Noah Smith. 2015. Contextualized sarcasm detection on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 9, pages 574–577.

Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488.

John D Campbell and Albert N Katz. 2012. Are there necessary conditions for inducing a sense of sarcastic irony? *Discourse Processes*, 49(6):459–480.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Abdelkader El Mahdaouy, Abdellah El Mekki, Kabil Essefar, Nabil El Mamoun, Ismail Berrada, and Ahmed Khoumsi. 2021. Deep multi-task model for sarcasm detection and sentiment analysis in Arabic language. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 334–339, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Ibrahim Abu Farha and Walid Magdy. 2020. From arabic sentiment analysis to sarcasm detection: The arsarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 32–39.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Lrec*, pages 392–398. Citeseer.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Véronique Moriceau, and Paolo Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in arabic tweets. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 10–13.

Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. 2003. Knn model-based approach in classification. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 986–996. Springer.

Barathi Ganesh HB, M Anand Kumar, and KP Soman. 2016. Distributional semantic representation in health care text classification. In *FIRE (Working Notes)*, pages 201–204.

Yuguang Huang and Lei Li. 2011. Naive bayes classification algorithm based on small sample set. In *2011 IEEE International conference on cloud computing and intelligence systems*, pages 34–39. IEEE.

Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1006–1011, Austin, Texas. Association for Computational Linguistics.

Jihen Karoui, Farah Benamara, Véronique Moriceau, Viviana Patti, Cristina Bosco, and Nathalie Aussenac-Gilles. 2017. Exploring the impact of pragmatic phenomena on irony detection in tweets: A multilingual corpus study. In *15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 1, pages 262–272.

Arpita Khare, Amisha Gangwar, Sudhakar Singh, and Shiv Prakash. 2022. Sentiment analysis and sarcasm detection of indian general election tweets.

Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. A la carte embedding: Cheap but effective induction of semantic feature vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22, Melbourne, Australia. Association for Computational Linguistics.

Akshi Kumar, Shubham Dikshit, and Victor Hugo C Albuquerque. 2021. Explainable artificial intelligence for sarcasm detection in dialogues. *Wireless Communications and Mobile Computing*, 2021.

Silviu Oprea and Walid Magdy. 2019. Exploring author context for detecting intended vs perceived sarcasm. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2859, Florence, Italy. Association for Computational Linguistics.

Silviu Oprea and Walid Magdy. 2020a. iSarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1279–1289, Online. Association for Computational Linguistics.

Silviu Vlad Oprea and Walid Magdy. 2020b. The effect of sociocultural variables on sarcasm communication online. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–22.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

B Premjith and Soman Kp. 2020. Amrita_cen_nlp@ wosp 3c citation context classification task. In *Proceedings of the 8th International Workshop on Mining Scientific Publications*, pages 71–74.

B Premjith, KP Soman, M Anand Kumar, and D Jyothi Ratnam. 2019. Embedding linguistic features in word embedding for preposition sense disambiguation in english—malayalam machine translation context. In *Recent Advances in Computational Intelligence*, pages 341–370. Springer.

Anuja Priyam, GR Abhijeeta, Anju Rathee, and Saurabh Srivastava. 2013. Comparative analysis of decision tree classification algorithms. *International Journal of current engineering and technology*, 3(2):334–337.

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers*, pages 213–223.

Arifur Rahaman, Ratnadip Kuri, Syful Islam, Md Javed Hossain, and Mohammed Humayin Kabir. 2021. Sarcasm detection in tweets: A feature-based approach using supervised machine learning models. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 12(6).

Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 97–106.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

Samer Muthana Sarsam, Hosam Al-Samarraie, Ahmed Ibrahim Alzahrani, and Bianca Wright. 2020. Sarcasm detection using machine learning algorithms in twitter: A systematic review. *International Journal of Market Research*, 62(5):578–598.

KP Soman, R Loganathan, and V Ajay. 2009. *Machine learning with SVM and other kernel methods*. PHI Learning Pvt. Ltd.

High Tech team at SemEval-2022 Task 6: Intended Sarcasm Detection for Arabic texts

Alami Hamza[1], **Abdessamad Benlahbib**[2], **Ahmed Alami**[3],
[1] Innov-Tech Laboratory, Departement of Engineering, High Technology School, Rabat, Morocco
[2] LISAC Laboratory, Faculty of Sciences Dhar EL Mehraz (F.S.D.M),
Sidi Mohamed Ben Abdellah University (U.S.M.B.A)
[3] Ibn Tofail University, National School of Applied Sciences, Kenitra, Morocco
hamza0alami@gmail.com, abdessamad.benlahbib@usmba.ac.ma,
alami.alami1996@gmail.com

Abstract

This paper presents our proposed methods for iSarcasmEval shared task. The shared task consists of three different subtasks. We participate in both subtask A and subtask C. The purpose of the subtask A was to predict if a text is sarcastic while the aim of subtask C is to determine which text is sarcastic given a sarcastic text and its non-sarcastic rephrase. Both of the developed solutions used BERT pre-trained models. The proposed models are optimized on simple objectives and easy to grasp. However, despite their simplicity our methods ranked 4 and 2 in iSarcasmEval subtask A and subtask C for Arabic texts.

1 Introduction

Nowadays, social media users provide a huge amount of text, images and videos. This large amount of data contains useful information (users ideas, opinions, events, etc) for various domains such as stock predictions, marketing, or politics. In order to benefit from these data, new fields of study have been introduced including sentiment analysis, opinion mining, author profiling, and harassment detection (Liu, 2012; Rosenthal et al., 2014; Maynard and Greenwood, 2014; Van Hee et al., 2018). Natural Language Processing (NLP) algorithms are used extensively in these fields to extract useful information. For instance, to determine whether a given product has a positive or negative sentiment in the market, we can apply NLP techniques to analyse a list of twitter posts to infer a sentiment about the product.

According to Oxford dictionary, sarcasm is "*The use of irony to mock or convey centempt*". Sarcastic text convey negative implied sentiment, however it can have positive, negative, or no surface sentiment. Sarcasm is commonly used in social media, thus it introduces errors in various tasks such as sentiment analysis and opinion mining. This is explained in the work of Rosenthal et al. (2014), it shows a significant drop in sentiment polarity classification performance when processing sarcastic tweets, compared to non-sarcastic ones. In this context, the task *iSarcasmEval: Intended Sarcasm Detection In English and Arabic* (Abu Farha et al., 2022) is organized by SemEval 2022. The main tasks consists of three subtask:

- Subtask A: Given a text, determine whether it is sarcastic or non-sarcastic.
- SubTask B (English only): A binary multi-label classification task. Given a text, determine which ironic speech category it belongs to, if any.
- SubTask C: Given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which is the sarcastic one.

In this paper, we describe our contribution to iSarcasmEval shared task, Arabic language only. For subtask A, we built a BERT-based neural network (Devlin et al., 2019; Antoun et al., 2020) classifier to determine whether a tweet is sarcastic or not.

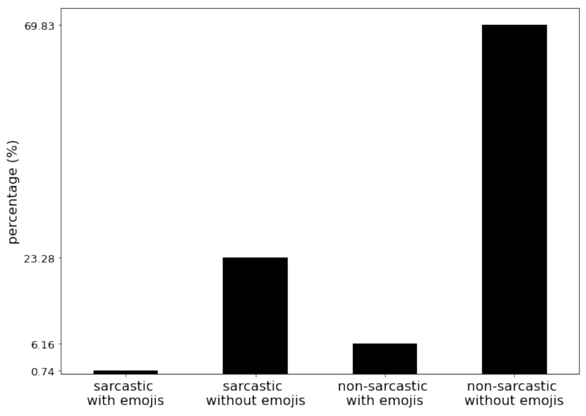


Figure 1: The train dataset distribution according to 4 classes including sarcastic texts that contains emojis, sarcastic texts without emojis, non-sarcastic texts with emojis, and non-sarcastic texts without emoji.

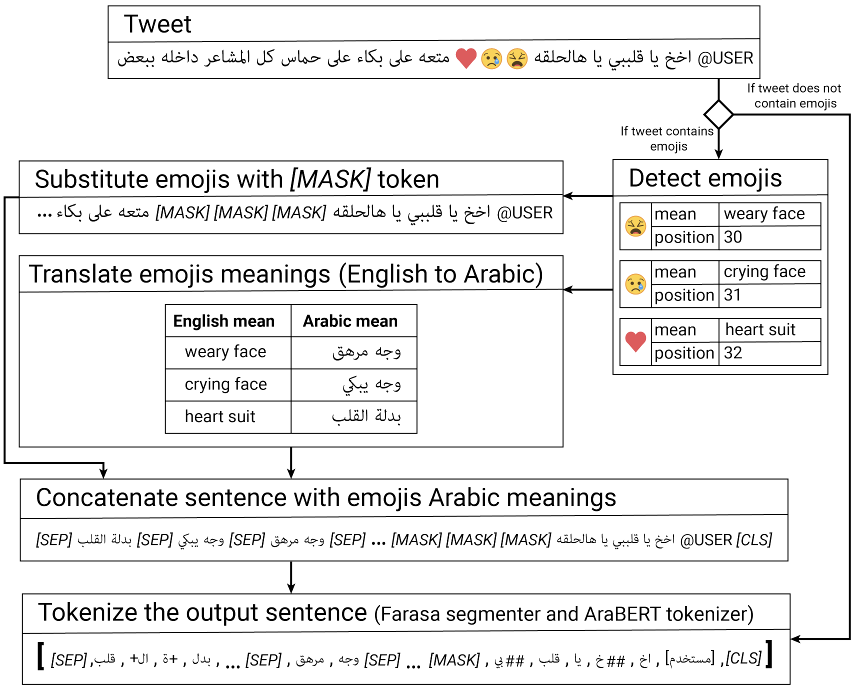

Tweet
If tweet does not contain emojis
If tweet contains emojis
Detect emojis
mean weary face
position 30
mean crying face
position 31
mean heart suit
position 32
Substitute emojis with *[MASK]* token
Translate emojis meanings (English to Arabic)
English mean
Arabic mean
weary face
وجه مرهق
crying face
وجه يبكي
heart suit
بدلة القلب
Concatenate sentence with emojis Arabic meanings
Tokenize the output sentence (Farasa segmenter and AraBERT tokenizer)

Figure 2: Text preprocessing proposed by Alami et al. (2020).

Our model obtained the fourth best performance in the subtask A. For subtask C, we built also a BERT-based classifier to detect the sarcastic text from two text that convey the same meaning. We scored the second best performance in the subtask C. The results are promising and there is much room for improvement.

The rest of the paper is organized as follows: Section 2 presents our method overview; Section 3 provides performance evaluation; Section 4 concludes the paper and provides future work.

2 Method Overview

In this section, we first describe how we split data to evaluate our models. Next, we explain preprocessing steps. Next, we discuss our models for each subtask, and the experimental setup we used. We also provide illustrations and examples, when necessary.

2.1 Dataset split

The organizers of iSarcasmEval provided Arabic texts annotated with their sarcasm labels. The train set contains 3102 samples where 75.98% (2357 samples) are non-sarcastic and 24.02% (745 samples) are sarcastic. The test set consists of 1400 samples. All the samples are annotated also with their dialect. We build a validation set from train set based on emojis. We first split the train data into 4 classes including sarcastic texts that contains emojis, sarcastic texts without emojis, non-sarcastic texts with emojis, and non-sarcastic texts without emojis. Fig. 1 illustrates the distribution of this 4 classes in the train dataset. We notice that only 6.9% of train samples contain emojis while 20.86% test samples include emojis. Considering this we use 4 splits to validate our models:

- Split A: The validation set contains all the sarcastic samples with emojis, 10% sarcastic samples without emojis, 10% non-sarcastic samples with emojis, and 10% non-sarcastic samples without emojis.
- Split B: The validation set contains 50% sarcastic samples with emojis, 10% sarcastic samples without emojis, 10% non-sarcastic samples with emojis, and 10% non-sarcastic samples without emojis.
- Split C: The validation set does not contain any sarcastic samples with emojis and contains 10% sarcastic samples without emojis, 10% non-sarcastic samples with emojis, and 10% non-sarcastic samples without emojis.
- Split D: The validation set contains 20% of train samples. We applied stratified split to have the same distribution of classes as the train set.

Table 1: Performance evaluation of different models for sarcasm prediction

	Split A			**Split B**			**Split C**			**Split D**			**Score**
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	
AraBERTv02-twitter	84.76	64.10	72.99	76.92	73.53	75.19	75.00	82.76	78.69	78.66	86.58	82.43	**77.32**
CAMEL-MIX	86.67	66.67	75.36	78.95	66.18	72.00	74.24	84.48	79.03	77.07	81.21	79.08	76.36
AraBERTv02-twitter / Emojis	90.74	62.82	74.24	55.93	48.53	51.97	75.41	79.31	77.31	77.22	81.88	79.48	70.75

2.2 Preprocessing

Our preprocessing step consists of tokenization. We apply the pre-trained BERT tokenizer which is based on wordpiece model (Schuster and Nakajima, 2012). For comparison purposes, we applied the same preprocessing process applied by Alami et al. (2020). The main idea is to integrate the meaning of emojis whitin the initial text. Fig. 2 presents the preprocessing step used in (Alami et al., 2020).

2.3 SubTask A

The objective of this task is to predict whether a text is sarcastic or not. We fin tune various BERT-based models pre-trained with Arabic large corpora. These models are used to extract valuable features from raw text. These features are then used with a softmax classifier to predict the label of the input text. All models are optimized to minimize the cross entropy loss.

2.4 SubTask C

The aim of this task is to predict the sarcastic text given two texts with the same meaning. Like the model used in subtask A, we fine tune BERT-based models for this specific task. The input of these models is the concatenation of the two texts separated by the special token *[SEP]*. Features are extracted with BERT-based models, then a softmax layer is applied to compute the probabilities of the events: first text is sarcastic and second text is sarcastic. All models are optimized to minimize the cross entropy loss.

2.5 Experimental Setup

We implemented our models using HuggingFace (Wolf et al., 2020). We used AraBERTv02-twitter (Antoun et al., 2020) and CAMEL-Mix (Inoue et al., 2021) as the pre-trained language models. To train our models, we used a batch size of 8, a learning rate 10^{-5}. We used the AdamW optimizer (Loshchilov and Hutter, 2017). We ran the experiments on a Google colaboratory environment [1].

[1]https://colab.research.google.com/

3 Performance Evaluation

In this section, we present the performance of various models trained on both subtasks A and C.

3.1 Subtask A

First, we compared the performances of two models: a model fine tuned with AraBERTv02-twitter and a model fine tuned with CAMEL-MIX. Table 1 shows the obtained results according to the 4 splits we previously discussed in subsection 2.1. We compute the overall score of a model by averaging all the f1 scores of the sarcastic class obtained from different splits. The model fine tuned with AraBERTv02-twitter scored the best results.

To investigate the impact of the substitution of emojis with their meanings. We evaluated the performances of a model based on AraBERTv02-twitter and take as input text preprocessed as proposed in Alami et al. (2020). Table 1 shows that emojis processing didn't improve the overall score.

Therefore, we submit the predictions obtained using AraBERTv02-twitter with the test set. We scored the fourth best score in the leaderboard (46.84% f1 score for sarcastic class).

3.2 Subtask C

Since the AraBERTv02-twitter model obtained the best result in subtask A, we trained the same base model to predict the sarcastic text given two text that convey the same meaning. We augmented the dataset by applying a simple rule which consist of switching the positions of the sarcastic text with the non-sarcastic text and replace the label with 0. We ranked 2 in the leaderboard with (88.5% accuracy).

4 Conclusion

We developed two methods for sarcasm prediction. The first one aim to predict if a text is sarcastic or not. This method is based on AraBERTv02-twitter pretrained model which extract valuable features from raw text. We achieved the fourth top performance in the iSarcasmEval subtask A for Arabic with a 46.84% f1 score for the sarcastic class. The second model has the objective to detect the sarcastic text given two texts that convey the same meaning. We trained a BERT-based model that take as input two text and predict the sarcastic one. We ranked 2 in the leaderboard with 88.5% accuracy. In future work, we plan to improve the performance of our models by using linguistic rules and some external datasets.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Hamza Alami, Said Ouatik El Alaoui, Abdessamad Benlahbib, and Noureddine En-nahnahi. 2020. LISAC FSDM-USMBA team at SemEval-2020 task 12: Overcoming AraBERT's pretrain-finetune discrepancy for Arabic offensive language identification. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2080–2085, Barcelona (online). International Committee for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

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

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in Arabic pre-trained language models. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, Kyiv, Ukraine (Online). Association for Computational Linguistics.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

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

Diana Maynard and Mark Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. SemEval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Cynthia Van Hee, Gilles Jacobs, Chris Emmery, Bart Desmet, Els Lefever, Ben Verhoeven, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2018. Automatic detection of cyberbullying in social media text. *PLOS ONE*, 13(10):1–22.

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

CS-UM6P at SemEval-2022 Task 6: Transformer-based Models for Intended Sarcasm Detection in English and Arabic

Abdelkader El Mahdaouy **Abdellah El Mekki** **Kabil Essefar**†
Abderrahman Skiredj† **Ismail Berrada**†
School of Computer Science, Mohammed VI Polytechnic University, Morocco
abdelkader.elmahdaouy@um6p.ma, abdellah.elmekki@um6p.ma
{firstname.lastname}@um6p.ma†

Abstract

Sarcasm is a form of figurative language where the intended meaning of a sentence differs from its literal meaning. This poses a serious challenge to several Natural Language Processing (NLP) applications such as Sentiment Analysis, Opinion Mining, and Author Profiling. In this paper, we present our participating system to the intended sarcasm detection task in English and Arabic languages. Our system[1] consists of three deep learning-based models leveraging two existing pre-trained language models for Arabic and English. We have participated in all sub-tasks. Our official submissions achieve the best performance on sub-task A for Arabic language and rank second in sub-task B. For sub-task C, our system is ranked 7th and 11th on Arabic and English datasets, respectively.

1 Introduction

Sarcasm is an important aspect of human natural language. It is characterized by the occurrence of a discrepancy between the intended and the literal meaning of utterance (Wilson, 2006). The prevalence of this phenomenon may jeopardize the performance of many NLP applications, such as Sentiment Analysis, Opinion Mining, and Emotion Detection, among others (Maynard and Greenwood, 2014; Rosenthal et al., 2014; Van Hee et al., 2018). Indeed, sarcasm detection has been the subject of many systematic investigation, where several shared tasks have been organized and a number of datasets have been introduced (Van Hee et al., 2018; Ghanem et al., 2019; Ghosh et al., 2020; Abu Farha et al., 2021). The existing datasets are either labeled by a human annotator or using distant supervision signals such as the presence of a set of predefined hashtags (Oprea and Magdy, 2020). However, these labeling methods might be sub-optimal for intended sarcasm detection as the author's sarcastic intent may differ from an annotator's perceived meaning (Oprea and Magdy, 2019, 2020). Besides, most existing research works have focused on English language (Van Hee et al., 2018; Ghosh et al., 2020), while few studies have been introduced for other languages such as Arabic (Ghanem et al., 2019; Abu Farha et al., 2021).

To overcome the aforementioned limitations, Abu Farha et al. (2022) have organized the iSarcasmEval shared task for intended sarcasm detection in English and Arabic languages. In contrast to previous research work, the introduced dataset, for intended sarcasm detection, is labeled by the authors themselves. The authors are then asked to provide non-sarcastic rephrases that covey the same intended meaning of their sarcastic texts. Further, the iSarcasmEval's organizers have relied on linguistic experts to categorize sarcastic texts into sarcasm, irony, satire, understatement, overstatement, and rhetorical questions (Leggitt and Gibbs, 2000).

In this paper, we present our participating system to iSarcasmEval shared task (Abu Farha et al., 2022). Our system rely on three transformer-based deep learning models. In all our models, we use existing Pre-trained Language Model (PLM) to encode the input text and apply a single attention layer to the contextualized word embedding of PLM (Barbieri et al., 2021; Abdul-Mageed et al., 2021). For all our models, we use the same classifier architecture composed of one hidden layer and one classification layer. The classifier is fed with the concatenation of the pooled output of the PLM as well as the attention layer output. We model the sub-task A as a binary classification (first model) and as a multi-class classification (second model). Motivated by the small size of the datasets and similarly to GAN-BERT architecture (Croce et al., 2020), the third model uses a conditional generator that tries to generate fake samples that are similar to the PLM's embedding of the real input data. The

[1]The source code of our system is available at https://github.com/AbdelkaderMH/iSarcasmEval

discriminator of the third model is trained to discriminate between fake and real samples as well as classify the real ones correctly. For sub-task B, we only employ the GAN based model (third model), while for sub-task C, we utilize all models trained on Task A and we compare the probabilities of the sarcastic class of the left and the right text. We train our models using several loss functions, including the focal loss (Lin et al., 2017). Besides, for sub-task A, we train our models with and without the non-sarcastic rephrase texts.

For the official submissions, we employ hard voting ensemble of our trained models. Our system achieve promising results as we rank 1st, 15th, 2nd, 7th, and 11th on sub-task A AR, sub-task A EN, sub-task B, sub-task C AR, and sub-task C EN, respectively.

2 Background

2.1 Task description

The organizers of iSarcasmEval shared task have provided training data and testing data for intended sarcasm detection in English and Arabic languages (Abu Farha et al., 2022). The datasets are collected from Twitter and labeled for intended sarcasm detection by the authors of the tweets. For English, the training data consists of 3,468 samples out of which 867 samples are sarcastic. For Arabic, the training data contains 3,102 samples, where 745 samples are sarcastic. The organizers also provide the non-sarcastic rephrase of sarcastic texts for both languages and the dialect of the given Arabic tweets. The shared task consists of the flowing sub-tasks:

- **Sub-task A** is a binary classification task, where the aim is to determine if a tweet is sarcastic or not. This sub-task consists of two sub-tasks A EN and A AR. The test data contains 1,400 for each language.

- **Sub-task B** is a multi-label classification task, where the aim is to assign a given tweet into the sarcasm, irony, satire, understatement, overstatement, and rhetorical question categories of ironic speech (Leggitt and Gibbs, 2000). This task is provided for English language only and the test data contains 1,400 samples.

- **Sub-task C** aims to identify the sarcastic tweet and the non-sarcastic rephrase given two texts that convey the same meaning. This sub-task consists of two sub-tasks C EN and C AR. The test sets of both languages consist of 200 samples.

2.2 Related work

In recent years, there has been a growing number of research works focusing on fine-tuning the existing PLMs on NLP tasks. These PLMs are based on the transformer architecture and are trained using self-supervised learning objectives such as Masked Language Modeling (MLM) amongst others (Devlin et al., 2019). Several multilingual and monolingual PLM variants are introduced (Devlin et al., 2019; Conneau et al., 2020; Antoun et al., 2020). For domain-specific data, domain adaptive fine-tuning of existing PLMs using MLM or domain adaptation have been shown to improve the performance of NLP applications (Rietzler et al., 2020; Barbieri et al., 2021; El Mekki et al., 2021a). Nevertheless, when the domain-specific data is sufficiently large, these transformers can be trained from scratch (Abdul-Mageed et al., 2021; Inoue et al., 2021).

For sarcasm detection, several research studies have been introduced based on fine-tuning the existing PLMs for English and Arabic languages (Ghanem et al., 2019; Ghosh et al., 2020; Abu Farha et al., 2021). El Mahdaouy et al. (2021) have shown that incorporating attention layers on top of the contextualized word embedding of the PLM improves the performance of multi-task and single-task learning models for both sarcasm detection and sentiment analysis in Arabic. The main idea consists of classifying the input text based on the concatenation of the PLM's pooled output and the output of the attention layer. This Architecture has yielded promising results on other tasks such as detecting and rating humor, lexical complexity prediction, and fine-grained Arabic dialect identification (Essefar et al., 2021; El Mamoun et al., 2021; El Mekki et al., 2021b).

Although transformer-based architectures have shown state-of-the-art performance on many NLP tasks, their task-specific fine-tuning requires a reasonable amount of labeled data. Nevertheless, in real-world applications, one may not always have enough labeled training data. Motivated by the performance of Semi-Supervised Generative Adversarial Networks (SS-GAN) in computer vision (Odena, 2016), Croce et al. (2020) have introduced GAN-BERT. The latter extends BERT fine-tuning procedure by training a generator and a discriminator. The generator is trained to generate fake samples embeddings that are similar to the real input text embeddings, whereas, the discriminator is trained to detect fake examples and categorize the real text inputs.

3 System overview

The submitted system to the iSarcasmEval shared task relies on three transformer-based models for intended sarcasm detection. In order to encode the input text, we utilize the Twitter-XLM-Roberta-base (Barbieri et al., 2021) and MARBERT (Abdul-Mageed et al., 2021) for English and Arabic texts, respectively. The former is a variant of XLM-RoBERTa PLM that is adapted to Twitter data using MLM objective, while the latter is a variant of BERT PLM that is pre-trained from scratch on Arabic tweet corpora. In the following subsection, we describe the components of our system.

3.1 Preprocessing

The tweet preprocessing component spaces out emojis and substitutes user's mention and URL with their special tokens of the PLM's tokenizer. For Twitter-XLM-Roberta-base, URLs and user's mentions are replaced by 'http' and '@user'. For MARBERT, they are replaced by 'user' and 'url' special tokens. To leverage the dialect information for Arabic data, we replace the dialect string with its full Arabic name and pass the input text to MARBERT's tokenizer as follows:

- [SEP] dialect [SEP] preprocessed text [SEP]

3.2 Deep Learning Models

Our three deep learning models are described as follows:

- **Model 1** consists of a transformer encoder, one attention layer, and a classifier. Following the work of (El Mahdaouy et al., 2021), we apply attention to the contextualized word embedding of the encoder. The classifier is composed of one hidden layer and one classification layer for binary classification. The classifier is fed with the concatenation of the PLM's pooled output and the attention layer's output.

- **Model 2** is similar is to **Model 1** and the task is modeled as multi-class classification problem. In other words, the classification layer of this model consists of two hidden units. Figure 1 illustrates the overall architecture of Model 1 and Model 2.

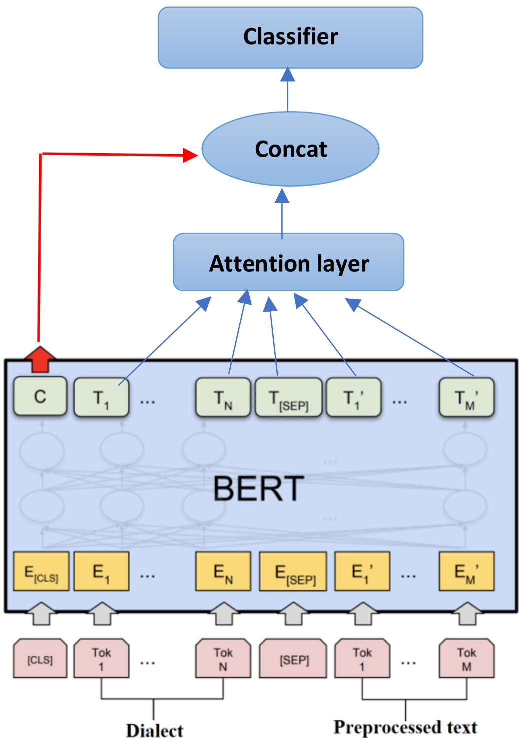


Figure 1: The overall architecture of Model 1 and Model 2.

- **Model 3** is similar to GAN-BERT model (Croce et al., 2020), whereas, we employ a conditional generator that generates fake embeddings from a random noise and the class category. The model 3 consits of three components, a BERT-based encoder with an extra attention layer on top of the contextualized word embedding, a generator, and a discriminator. The encoder represents the input sentences using the CLS token embedding and the output of the attention layer. The generator is trained to fool the discriminator by generating fake embedding representations that are similar to the input text embedding. It consists of two hidden layers and one output layer. Each hidden layer is flowed by a dropout layer and the relu activation layer. The discriminator is trained to discriminate between fake examples and real ones and to classify real input texts into sarcastic and non-sarcastic labels. The discriminator is also composed of two hidden layers and one classification layer. Similarly to the generator, the hidden layers are followed by one dropout layer and the relu activation layer.

3.3 Training objectives

For training our models, we utilizes several loss functions, including the Focal Loss (Lin et al., 2017). The aim is to assess the performance of the following training objectives under class imbalance:

- For **Model 1**, we investigate the Binary Cross-Entropy loss (BCE), the Weighted Binary Cross-Entropy loss (W. BCE), and the Binary Focal Loss (BFL). For the W. BCE loss, the positive class weight is set to:

$$\frac{batch\_size - positive\_count}{positive\_count + \epsilon}$$

- For **Model 2 and 3**, we investigate the Cross-Entropy loss, the Weighted Cross-Entropy loss (W. CE), and the Focal Loss (FL). For the W. CE, the positive and the negative class weights are computed as follows:

$$\begin{cases} pos\_weight = \frac{batch\_size - positive\_count}{positive\_count + \epsilon} \\ neg\_weight = \frac{batch\_size - negative\_count}{negative\_count + \epsilon} \end{cases}$$

4 Experimental setup

All our models are implemented using the PyTorch[2] framework and the open-source Transformers[3] libraries. Experiments are conducted on a PowerEdge R740 Server having 44 cores Intel Xeon Gold 6152 2.1GHz, a RAM of 384 GB, and a single Nvidia Tesla V100 with 16GB of RAM. 20% of the training set is used for the model validation. All our models are trained using Adam optimizer with a linear learning rate scheduler. Based on our preliminary results, obtained on the validation set, the learning rate, the number of epochs, and the batch size are fixed to 1×10^{-5}, 10, and 16 respectively. For the focal loss, the hyper-parameters γ and α (the weight of the negative class) are set to 2 and 0.8 respectively. All models are evaluated using the Accuracy as well as the macro averaged Precision, Recall, and F1 measures. Besides, we train our models with and without rephrase texts for sub-task A. For sub-task B, our models are trained on sarcastic tweets.

[2] https://pytorch.org/
[3] https://huggingface.co/transformers/

5 Results

In this section, we present the obtained results of our models as well as our official submissions. It is worth mentioning that for sub-task C, we employ the trained models on sub-task A and we use their output probabilities to discriminate between the sarcastic text and the non-sarcastic rephrase. Besides, for our official submissions, we use the hard vote ensemble of our trained models. For each loss function, the best performance obtained is highlighted in **bold** font, while the overall best performance for each task is highlighted in ***bold-italic*** font.

5.1 Sub-task A

Table 1 summarizes the obtained results. The results show that training the models on tweets as well as the non-sarcastic rephrases improve the performance for both languages, especially Arabic where an important performance increment is yielded. Moreover, the performance of the evaluated models depends on the employed loss function. Although Model 1 and Model 2 are simple, they achieve better results than the GAN-based model (Model 3). The best performances on sub-task A are obtained using Model 2 in conjunction with the FL loss (0.6217) and Model 1 in conjunction with BCE loss (0.3833) for Arabic and English respectively. In our official submission, ensembling the models that are trained with and without the rephrase data harms the results. Our submitted system achieves the best performance on Arabic and ranked 15th on the English.

5.2 Sub-task B

Table 2 presents our obtained results for sub-task B. Since we use the sarcastic tweets only for training, we only train the Model 3. The results show that the best performance is obtained using the BCE loss. The FL and W. BCE loss functions have not improved the results. This might be explained by the fact that we did not tune the hyper-parameters α and $gamma$ of FL loss. Besides, in the W. BCE, the positive classes are assigned larger importance weights in comparison to the negative ones (see Section 3.3). Our official submission yields the second-best results on this sub-task.

5.3 Sub-task C

Table 3 summarizes the obtained results on sub-task C test sets of Arabic and English languages. In accordance with the results of sub-task A, the

Table 1: The obtained results on the test set of sub-task A for both Arabic and English. For our non official submissions, we report the macro F-1 score of the sarcastic class only.

	Sub-Task A Arabic						**Sub-Task A English**					
	Tweet only			Tweet + rephrase			Tweet only			Tweet + rephrase		
	BCE/CE	BFL/FL	W. BCE/CE	BCE	BFL/FL	W. BCE/CE	BCE/CE	BFL/FL	W. BCE/CE	BCE/CE	BFL/FL	W. BCE/CE
Model 1	0.5253	**0.5621**	0.4565	**0.6135**	0.5787	**0.5793**	0.3485	**0.3605**	0.3421	***0.3833***	0.3313	**0.3714**
Model 2	0.5307	0.4481	0.4892	0.5949	***0.6217***	0.5505	**0.3574**	0.3375	0.3144	0.3770	**0.3619**	0.3090
Model 3	**0.56**	0.5271	**0.4937**	0.5339	0.5488	0.5345	0.3470	0.3448	**0.3478**	0.3517	0.3517	0.3557

	Official Submission						**Official Submission**					
Ensembling	F-1 sarcastic	F-score	Precision	Recall	Accuracy	**Rank**	F-1 sarcastic	F-score	Precision	Recall	Accuracy	**Rank**
	0.5632	0.7188	0.6948	0.8362	0.8050	**1**	0.3713	0.6171	0.6058	0.6458	0.7750	**15**

Table 2: The obtained results on the test set of sub-task B.

		F-1 Macro	F-1 sarcasm	F-1 irony	F-1 satire	F-1 understatement	F-1 overstatement	F-1 rhetorical question
Model 3	BCE	**0.0924**	**0.2331**	0.1676	0.0530	0	0	**0.1008**
	BFL	0.0877	0.2298	**0.1733**	0.025	0	0	0.0983
	W. BCE	0.0681	0.2302	0.0705	**0.0581**	0	0	0.0501
Ensembling	**Rank**	**Official Submission**						
	2	0.0875	0.2314	0.1622	0.0392	0	0	0.0923

Table 3: The obtained results on the test set of sub-task C for both Arabic and English. For our non-official submissions, we report the accuracy score only.

	Sub-Task C Arabic						**Sub-Task C English**					
	Tweet only			Tweet + rephrase			Tweet only			Tweet + rephrase		
	BCE/CE	BFL/FL	W. BCE/CE	BCE	FL	W. BCE/CE	BCE/CE	BFL/FL	W. BCE/CE	BCE/CE	BFL/FL	W. BCE/CE
Model 1	**0.68**	**0.69**	0.61	0.815	**0.82**	0.83	**0.69**	**0.68**	**0.67**	0.695	**0.7**	**0.685**
Model 2	0.65	0.575	0.59	**0.835**	0.815	***0.85***	0.655	**0.68**	0.65	0.685	0.67	0.655
Model 3	0.67	0.68	**0.685**	**0.835**	0.815	0.835	0.655	**0.68**	0.625	***0.71***	0.685	**0.685**

	Official Submission			**Official Submission**		
Ensembling	Accuracy	F-1 Score	**Rank**	Accuracy	F-1 Score	**Rank**
	0.7800	0.7688	**7**	0.6950	0.6481	**11**

models that are trained on the tweets and the non-sarcastic rephrases yield better performances. The best-obtained accuracy scores are 0.85 and 0.71 in comparison to 0.78 and 0.69 obtained by our official submission for Arabic and English respectively. Hence, ensembling the models that are trained with and without the rephrase data harms the performance of our official submission. Our official submission is ranked 7th and 11th on Arabic and English respectively.

6 Conclusion

In this paper, we present our participating system in the iSarcasmEval shared task for intended sarcasm detection in English and Arabic. Our system relies on three deep learning-based models that leverage two existing pre-trained language models for Arabic and English. We participate in all sub-tasks, investigate several training objectives, and we study the impact of including non-sarcastic rephrase in the training data. The results show that ensembling models that are trained with and without rephrases have a negative impact on the official results. Our official submissions achieve the best performance on sub-task A for the Arabic language and rank in the second position on sub-task B.

Acknowledgments

Experiments presented in this paper were carried out using the supercomputer Toubkal, supported by Mohammed VI Polytechnic University (`https://www.um6p.ma`), and facilities of simlab-cluster HPC & IA platform.

References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages

7088–7105, Online. Association for Computational Linguistics.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ibrahim Abu Farha, Wajdi Zaghouani, and Walid Magdy. 2021. Overview of the WANLP 2021 shared task on sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 296–305, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Francesco Barbieri, Luis Espinosa-Anke, and Jose Camacho-Collados. 2021. A Multilingual Language Model Toolkit for Twitter. In *arXiv preprint arXiv:2104.12250*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

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

Abdelkader El Mahdaouy, Abdellah El Mekki, Kabil Essefar, Nabil El Mamoun, Ismail Berrada, and Ahmed Khoumsi. 2021. Deep multi-task model for sarcasm detection and sentiment analysis in Arabic language. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 334–339, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Nabil El Mamoun, Abdelkader El Mahdaouy, Abdellah El Mekki, Kabil Essefar, and Ismail Berrada. 2021. CS-UM6P at SemEval-2021 task 1: A deep learning model-based pre-trained transformer encoder for lexical complexity. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 585–589, Online. Association for Computational Linguistics.

Abdellah El Mekki, Abdelkader El Mahdaouy, Ismail Berrada, and Ahmed Khoumsi. 2021a. Domain adaptation for Arabic cross-domain and cross-dialect sentiment analysis from contextualized word embedding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2824–2837, Online. Association for Computational Linguistics.

Abdellah El Mekki, Abdelkader El Mahdaouy, Kabil Essefar, Nabil El Mamoun, Ismail Berrada, and Ahmed Khoumsi. 2021b. BERT-based multi-task model for country and province level MSA and dialectal Arabic identification. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 271–275, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Kabil Essefar, Abdellah El Mekki, Abdelkader El Mahdaouy, Nabil El Mamoun, and Ismail Berrada. 2021. CS-UM6P at SemEval-2021 task 7: Deep multi-task learning model for detecting and rating humor and offense. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1135–1140, Online. Association for Computational Linguistics.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Véronique Moriceau, and Paolo Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in arabic tweets. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, FIRE '19, page 10–13, New York, NY, USA. Association for Computing Machinery.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 1–11, Online. Association for Computational Linguistics.

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in Arabic pre-trained language models. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 92–104, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

John S. Leggitt and Raymond W. Gibbs. 2000. Emotional reactions to verbal irony. *Discourse Processes*, 29(1):1–24.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. *CoRR*, abs/1708.02002.

Diana Maynard and Mark Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

Augustus Odena. 2016. Semi-supervised learning with generative adversarial networks.

Silviu Oprea and Walid Magdy. 2019. Exploring author context for detecting intended vs perceived sarcasm. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2859, Florence, Italy. Association for Computational Linguistics.

Silviu Oprea and Walid Magdy. 2020. iSarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1279–1289, Online. Association for Computational Linguistics.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2020. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4933–4941, Marseille, France. European Language Resources Association.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. SemEval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743. Language in Mind: A Tribute to Neil Smith on the Occasion of his Retirement.

TechSSN at SemEval-2022 Task 6: Intended Sarcasm Detection using Transformer Models

Rajalakshmi Sivanaiah, Angel Deborah S, Sakaya Milton R,
Mirnalinee T T, Ramdhanush Venkatakrishnan
Department of Computer Science and Engineering
Sri Sivasubramaniya Nadar College of Engineering
Chennai - 603110, Tamil Nadu, India
{rajalakshmis, angeldeborahs}@ssn.edu.in,
{miltonrs, mirnalineett,ramdhanush2010105}@ssn.edu.in

Abstract

Irony detection in the social media is an upcoming research which places a main role in sentiment analysis and offensive languague identification. Sarcasm is one form of irony that is used to provide intended comments against realism. This paper describes a method to detect intended sarcasm in text (SemEval-2022 Task 6). The TECHSSN team used Bidirectional Encoder Representations from Transformers (BERT) models and its variants to classify the text as sarcastic or non-sarcastic in English and Arabic languages. The data is preprocessed and fed to the model for training. The transformer models learn the weights during the training phase from the given dataset and predicts the output class labels for the unseen test data.

1 Introduction

Sarcasm is a form of verbal irony that occurs when there is a discrepancy between the literal and intended meanings of an utterance. This is often used to express the opposite meaning of the words spoken. This is used frequently while making fun of someone or something, and is used in a variety of contexts, like casual conversation, memes, or even public speaking, to convey a variety of meanings, providing a certain level of depth and sophistication to the communication of the language.

Sarcasm is present in all overcontemporary social media networks and may reduce the efficiency of systems that perform operations on these sarcastic data such as sentiment analysis, opinion mining, author profiling, and harassment detection (Liu, 2012; Rosenthal et al., 2014; Maynard and Greenwood, 2014; Van Hee et al., 2018). It generates misleading conclusions, due to its nature to imply different meaning than what is intended on the surface. Even in SemEval, (Rosenthal et al., 2014) shows that there is a significant drop in system performance when processing sarcastic text data, in comparison to non-sarcastic data. These systems are used in industry, driving marketing, administration, and investment decisions (Medhat et al., 2014). This clearly shows that developing models to find and detect sarcasm is becoming more important by the day.

The iSarcasmEval Task 6 for SemEval 2022 (Abu Farha et al., 2022) is comprised of three SubTasks: To classify the input text as sarcastic or not, in English (SubTaskA English) and Arabic (SubTask A Arabic), and further classify sarcastic text into categories (SubTask B), and given two phrases with same meaning, identify the sarcastic one (SubTask C English, SubTask C Arabic). Of these, the TechSSN team has attempted to solve SubTaskA English, SubTask A Arabic, and SubTask B.

2 Related Work

A lot of the previous sarcasm detection datasets have been annotated using a weak supervision method. In weak supervision, text data is classified as sarcastic only if it meets a certain set of conditions that are decided upon prior to the collection and analysis of the data. This includes using tags (e.g. #sarcasm, #irony) (Ptacek et al., 2014; Khodak et al., 2018) to perform the above mentioned classification. However, this can result in noisy labels for many reasons, as demonstrated by (Oprea and Magdy, 2020).

Other work makes use of manual labelling, where sarcasm labels are provided by human annotators (Filatova, 2012; Riloff et al., 2013a; Abercrombie and Hovy, 2016). But, this can mean that labels are subjective in nature, i.e. labels may reflect annotator perception, which may differ from the meaning intended by the author, as pointed out by (Oprea and Magdy, 2020).

Moreover, a lot of sarcasm detection work applies only to the English language and, because of the socio-cultural aspects of sarcastic communication (Oprea and Magdy, 2020), it is doubtful that the models trained to detect sarcasm in the English language could do the same task with the same effectiveness on other languages such as Arabic (where most of the sarcasm detection is carried out using the above-mentioned weak supervision).

We have participated for irony and sarcasm detection SemEval task in (Sivanaiah et al., 2018) and used MultiLayer Perceptron model to find the ironic and sarcastic tweets. We have used CNN, RNN, LSTM, BERT and COLBERT models for offensive language detection in earlier SemEval workshop tasks (Sivanaiah et al., 2021), (Sivanaiah et al., 2020), (Sivanaiah et al., 2019) in which BERT models provides better results than other machine learning and deep learning models.

3 Methodology

3.1 Dataset

We used the dataset provided by the organizers of the Task to train and build the model. The dataset has fields for sarcasm, irony, satire, understatement, overstatement, rhetorical questioning, all of which are binary, and other categories of sarcastic text, along with a field for a non-sarcastic rephrase. It contains about 3467 entries of which about 866 are sarcastic and the rest are not sarcastic. The Arabic dataset has fields for sarcasm, rephrased text, and the regional dialect. It has about 3102 entries of which about 746 entries are sarcastic and the rest are not sarcastic. The test dataset entries for Task A English is 1400, Task A Arabic is 1400, Task B is 1616. In addition, the Task B dataset has fields for sarcastic rating and regional dialect.

3.2 Data Pre-processing

First, the raw data is tokenized. This means that each sentence is tokenized or split into sub-words for the BERT model. This is done using thecompute_input_arrays method that is available under the BertTokenizer class. This method makes use of a pre-trained 'BERT-base-uncased' model to tokenize the input sentences. The maximum sequence length is set as 200 as BERT requires inputs to be in a fixed size and shape. After trimming the input, the pre-trained model combines segments and creates appropriate masks for the given data. The input representation for the model is shown in Figure 1. The token embeddings, segmentation embeddings and position embeddings are summarized together to form the input embeddings.

3.3 Models and training

We have used pre-trained models for each Subtask. We have used BERT and its modifications such as Contextualized Late Interaction over BERT (CoLBERT) (Khattab and Zaharia, 2020) and A Robustly Optimized BERT Pretraining Approach (RoBERTa) (Liu et al., 2019). BERT, simply put, is a stack of encoders part of the transformer architecture. It uses attention on the decoder side, and self-attention on the encoder side. Base BERT has 768 hidden units, 12 attention heads, and 110M parameters. Similarly, Large BERT has 1024 hidden units, 16 attention heads, and 340M parameters. The BERT model takes a classification token, followed by a sequence of words as input. The input is then passed through several layers of encoder stack (12 in Base BERT, 24 in Large BERT). Each of the many layers applies self-attention, sends the output through a feedforward network of hidden layers, and then sends the output to the next encoder layer.

The procedures for pretraining and finetuning the model is shown in Figure 2. Same architecture structure is used in pre-training anf fine-tuning, differs only in the output layers. Both the encoder and decoder stream tasks are initialized with the same pre-trained model parameters. All parameters are fine-tuned in tuning phase. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

RoBERTa is a modification to the original BERT, and needs about the same amount of parameters that Base BERT requires (110M). It takes more training time than BERT, about 4-5 times more than BERT, but can provide more accurate results and predictions compared to BERT. CoLBERT is faster than many other BERT-based models and uses a pre-trained BERT model to handle late interactions. The model is trained with the training data provided by the organizers.

4 Results and Discussions

The test dataset was provided by the SemEval-2022 organizers and was given to different models for each Subtask and the results are listed in Tables 1 to 3.

ColBERT model gives better accuracy for the English language than Arabic. For subtask B we have used multilabel classifier to predict the output.

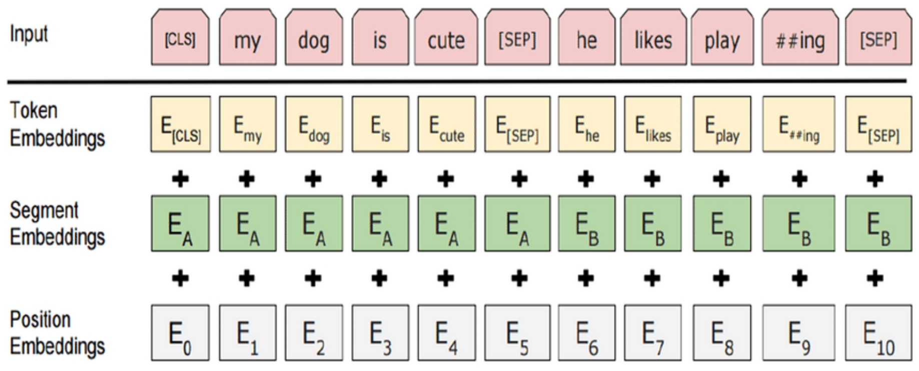

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
E[CLS] Emy Edog Eis Ecute E[SEP] Ehe Elikes Eplay E##ing E[SEP]
Segment Embeddings
EA EA EA EA EA EA EB EB EB EB EB
Position Embeddings
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E10

Figure 1: Input Representation – *source:(Devlin et al., 2018)*

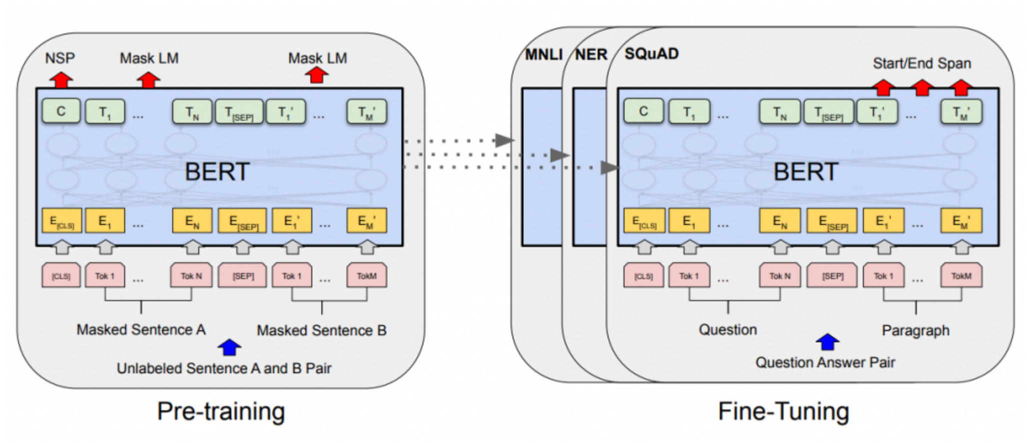


Figure 2: Pretraining and Fine tuning BERT – *source:(Devlin et al., 2018)*

Models	**F1-Score**	**Accuracy**
BERT	0.2558	0.2936
ColBERT	0.2637	0.7407

Table 1: Subtask A - English Results

Models	**F1-Score**	**Accuracy**
BERT	0.2292	0.3707

Table 2: Subtask A - Arabic Results

Models	**F1-Macro**
RoBERTa	0.0596

Table 3: Subtask B Results

5 Conclusion

It is obvious that the detection of sarcasm and its various categories is important as it is pivotal in computations like sentiment analysis, opinion mining, author profiling, or harassment checking. This can become increasingly difficult and tedious to compute, as sarcasm is extremely subjective as its nature itself is implying and contradictory, and the amount of data to be analyzed is getting larger and complex by the day. It is also a big step for Natural Language Processing (NLP) as it can tremendously help in the creation of more sophisticated virtual assistants and chatbots, which can emulate conversations that are closer to human interactions and life-like.

SemEval-2022 Task 6 is comprised of three SubTasks of which the TechSSN team has participated in Subtask A English, Subtask A Arabic and Subtask B. Deep learning models like BERT, RoBERT, and CoLBERT were used to carry out the tasks successfully. The team was able to obtain the 27th rank in Task A – English, 29th rank in Task A – Arabic, and 17th rank in Task B. Results show that BERT based models perform better on average than other conventional models like Logistic Regression Decision Tree, Support Vector Machines etc. Because of BERT's multilingual nature, sarcasm detection can be carried out in other languages across the globe. We would like to investigate further and apply these models to other languages. The accuracy could potentially be improved by using more advanced and efficient pre-processing techniques.

References

Gavin Abercrombie and Dirk Hovy. 2016. Putting sarcasm detection into context: The effects of class imbalance and manual labelling on supervised machine classification of twitter conversations. In *Proceedings of the ACL 2016 student research workshop*, pages 107–113.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Lrec*, pages 392–398. Citeseer.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. A la carte embedding: Cheap but effective induction of semantic feature vectors. *arXiv preprint arXiv:1805.05388*.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

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

Diana G Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Lrec 2014 proceedings*. ELRA.

Walaa Medhat, Ahmed Hassan, and Hoda Korashy. 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113.

Preslav Nakov, Sara Rosenthal, Svetlana Kiritchenko, Saif M Mohammad, Zornitsa Kozareva, Alan Ritter, Veselin Stoyanov, and Xiaodan Zhu. 2016. Developing a successful semeval task in sentiment analysis of twitter and other social media texts. *Language Resources and Evaluation*, 50(1):35–65.

Silviu Oprea and Walid Magdy. 2019. Exploring author context for detecting intended vs perceived sarcasm. *arXiv preprint arXiv:1910.11932*.

Silviu Vlad Oprea and Walid Magdy. 2020. The effect of sociocultural variables on sarcasm communication online. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–22.

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers*, pages 213–223.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

Rajalakshmi Sivanaiah, Angel Deborah S, S Milton Rajendram, and Mirnalinee T T. 2018. SSN MLRG1 at SemEval-2018 task 3: Irony detection in English tweets using MultiLayer perceptron. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 633–637, New Orleans, Louisiana. Association for Computational Linguistics.

Rajalakshmi Sivanaiah, Angel Deborah S, S Milton Rajendram, Mirnalinee TT, Abrit Pal Singh, Aviansh Gupta, and Ayush Nanda. 2021. TECHSSN at SemEval-2021 task 7: Humor and offense detection and classification using ColBERT embeddings. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1185–1189, Online. Association for Computational Linguistics.

Rajalakshmi Sivanaiah, Angel Suseelan, Logesh B, Harshini S, Geetika B, Dyaneswaran S, S Milton Rajendram, and Mirnalinee T T. 2019. TECHSSN at SemEval-2019 task 6: Identifying and categorizing offensive language in tweets using deep neural networks. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 753–758, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Rajalakshmi Sivanaiah, Angel Suseelan, S Milton Rajendram, and Mirnalinee T.t. 2020. TECHSSN at SemEval-2020 task 12: Offensive language detection using BERT embeddings. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2190–2196, Barcelona (online). International Committee for Computational Linguistics.

Cynthia Van Hee, Gilles Jacobs, Chris Emmery, Bart Desmet, Els Lefever, Ben Verhoeven, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2018. Automatic detection of cyberbullying in social media text. *PloS one*, 13(10):e0203794.

I2C at SemEval-2022 Task 6: Intended Sarcasm in English using Deep Learning Techniques

Adrián Moreno Monterde
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
adrian.moreno521@alu.uhu.es

Laura Vázquez Ramos
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
laura.vazquez005@alu.uhu.es

Victoria Pachón Álvarez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
vpachon@dti.uhu.es

Jacinto Mata Vázquez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
mata@uhu.es

Abstract

Sarcasm is often expressed through several verbal and non-verbal cues, e.g., a change of tone, overemphasis in a word, a drawn-out syllable, or a straight looking face. Most of the recent work in sarcasm detection has been carried out on textual data. This paper describes how the problem proposed in *Task 6: Intended Sarcasm Detection in English* (Abu Arfa et al. 2022) has been solved. Specifically, we participated in *Subtask B: a binary multi-label classification task*, where it is necessary to determine whether a tweet belongs to an ironic speech category, if any. Several approaches (classic machine learning and deep learning algorithms) were developed. The final submission consisted of a BERT based model and a macro-F1 score of 0.0699 was obtained.

1 Introduction

Existing social media analysis systems are limited by their inability to accurately detect and interpret figurative language. Sarcasm is often used by individuals to express opinions on complex matters and regarding specific targets (Carvalho et al. 2009).

Early computational models for verbal and irony and sarcasm detection have relied on shallow methods exploiting conditional token count regularities. But lexical clues alone are insufficient to discern sarcasm intent. Appreciating the context of expression is critical for this; even for humans (Wallace et al. 2014). Indeed, the exact same sentence can be interpreted as literal or sarcastic, depending on the speaker. Consider the sarcastic tweet in Figure 1 (ignoring for the moment the attached *#sarcasm* hashtag). Without knowing the author's political inclination, it would be difficult to conclude with certainty whether the tweet was intended as sarcastic or not.

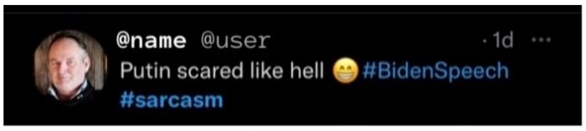

@name @user · 1d ···
Putin scared like hell 😁 #BidenSpeech #sarcasm

Figure 1: An illustrative tweet.

This task is about the binary classification of tweets in English based on the category of ironic speech. As a Multilabel-Classification, a tweet can belong to multiple categories or none. This research is important because social networking sites have changed our lives in recent years. For companies, this social reality has turned into an obligation to set up communication and marketing channels through social networks. Besides, companies require feedback from consumers, through comments or messages that mark the acceptance or rejection of each of the proposals, products or services.

For this competition 2 models/strategies were used:

- The first model is a binary multilabel classifier using Bayesian networks.
- The second model and final submit on the competition consisted of six different binary classifiers using BERT. In other words, one for each evaluable label: sarcasm, irony, satire,

understatement, overstatement and rhetorical question.

In this task we have analysed the sarcastic behaviour of people on social networks, in this case twitter, and the ways in which people express it. The dataset proposed by the organization was very unbalanced, so we had to apply several data balancing techniques in order to achieve good results.

The rest of this paper is organized as follows: in section 2 we explain the dataset, the meaning of the labels and data distribution. Therefore, we refer to other research that has helped us in our approach to this one. In section 3, several techniques and methods applied to our models to improve their performance and metrics are described. In section 4 we explain the libraries used and their usefulness. Finally, in section 5, the scores obtained with our proposed approaches are presented.

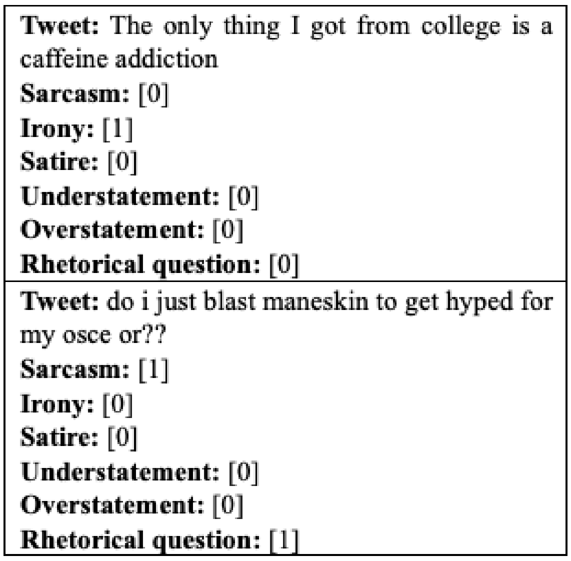
2 Background

As mentioned above, this paper is focused on Subtask B: binary multilabel classification. The original dataset has 3467 tweets in English with a maximum size of 280 characters (tweet limit). As a matter of fact, only the first 867 tweets will be useful for our task because the rest of the tweets do not have the labels that we are going to work with. Furthermore, the columns “Unnamed:0”, “rephrase” and “sarcastic” have been removed because they are useless for our task perfomance.

For a better analysis and understanding of the multiple labels that we have in our dataset it is important to mention the *ironic speech* exposed in (Leggitt and Gibbs 2000):

1. *Sarcasm*: tweets that contradict the state of affairs and are critical towards an addressee
2. *Irony*: tweets that contradict the state of affairs. but are not obviously critical towards an addressee.
3. *Satire*: tweets that appear to support an addressee but contain underlying disagreement and mocking.
4. *Understatement*: tweets that undermine the importance of the state of affairs they refer to
5. *Overstatement*: tweets that describe the state of affairs in terms that are obviously exaggerated.
6. *Rhetorical question*: tweets that include a question whose invited inference (implication) is obviously contradicting the state of affairs.

In Figure 2, two samples of the dataset with the information used in our approaches can be seen.


Tweet: The only thing I got from college is a caffeine addiction
Sarcasm: [0]
Irony: [1]
Satire: [0]
Understatement: [0]
Overstatement: [0]
Rhetorical question: [0]

Tweet: do i just blast maneskin to get hyped for my osce or??
Sarcasm: [1]
Irony: [0]
Satire: [0]
Understatement: [0]
Overstatement: [0]
Rhetorical question: [1]

Figure 2: Example of two rows

When it comes to the multiple labels (categories) of sarcasm, Table 1 shows the distribution of the tweets into these categories. As can be seen, the dataset is imbalanced so, in the next section, we explain how the dataset was balanced for a better performance.

Category	Number of tweets
Sarcasm	713
Irony	155
Satire	25
Understatement	10
Overstatement	40
Rhetorical question	101

Table 1: Distribution of tweets in each category

This challenge has been approached by different researchers. In (Davidov et al., 2010), experiments with semi-supervised sarcasm identification on a Twitter dataset (5.9 million tweets) were carried out using 50 Twitter tags and 15 emojis as sentiment labels. They used a 5-fold cross validation on their classifier getting a F1-score of 0.55.

In addition, in (Tsur et al.,2010), they propose a semi supervised system for sarcasm recognition over 66,000 products reviews from Amazon. They used the same strategy as in the previous mention and obtained and F-score of 0.83 on the product reviews dataset.

More recently, other approaches have been developed to solve the task of sarcasm detection. In (Ashwita et al. 2021), the authors experimented by varying the amount of context used along with the response (text to be classified) and found that including the last utterance in the dialogue along with the response improved the performance of their system.

In (Khatri, P, Pranav, y M, Dr. Anand Kumar 2020), a model using machine learning techniques with BERT and GloVe embeddings to detect sarcasm in tweets was proposed.

3 System Overview

This section describes the two types of models that were submitted and the techniques and methods applied to each model to improve their performance and metrics.

3.1 Data augmentation

One of the main problems with the dataset is the small number of tweets to train our models (only 867 tweets). To solve this, a data augmentation technique was applied. In particular, a synonym augmenter (Wordnet, English) (McCrae et al. 2019) was used to create a new tweet but only swapping one random word by its synonym and keeping their labels. An example of this technique can be seen in Table 2.

Original tweet	The quick brown fox jumps over the lazy dog
New tweet	The quick gray fox jumps over the lazy dog

Table 2: Example of data augmentation

We suggest applying this technique only once because our model could overfitting the data and could yield overrated results of the metrics.

3.2 External databases

Another technique applied in the proposed models for the data augmentation was the manual insertion of tweets and labels (Oprea and Magdy 2019). Most of the tweets inserted belong to minority labels (we can see the minority classes on Table 1) such as satire, overstatement or understatement. Finally, once the new tweets were manually added, the dataset consisted of 904 tweets.

3.3 Text Processing

We have applied three versions of text processing to clean and simplify the text based on the work described in (Alzahrani and Jolonian 2021).

Text processing v1.0: this is the most basic pre-process. For this version, the following guidelines were applied:

- Conversion of all characters to lowercase.
- Extent of all possible contractions in English (e.g., what's → what is).
- Removal of emojis.
- Removal of special characters.
- Removal of multiple spaces between characters.

Text processing v2.0: this is the intermediate version. In addition to the features described at v1.0, the following features were added:

- Removal of emojis made from keyboard characters
- Removal of mentions
- Removal of links

Text processing v3.0: this is the full version. In addition to the features presented in v2.0, a removal of stopwords in English was added.

3.4 First model: Bayesian networks

The first model that was developed involves the classic algorithm of Bayesian networks (Heckerman and Wellman 1995) to study the pattern of behavior that the categories of sarcasm may present in our dataset.

We used a naive Bayes classifier (NBC) which assumes that the attributes are independent of each other. That is to say, the probability can be obtained by calculating the product of the individual conditional probabilities of each attribute given the class node as it can be seen on Figure 3.

In this model, an input (a single tweet) is provided, and it returns a vector of size six (one for each tag) with the predicted label.

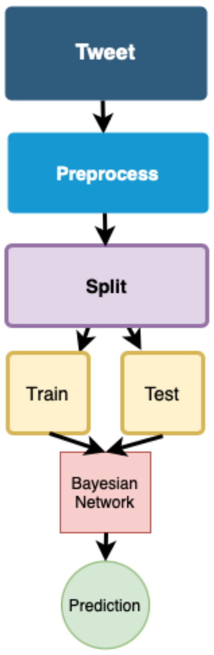


Figure 3: Steps followed on the first model

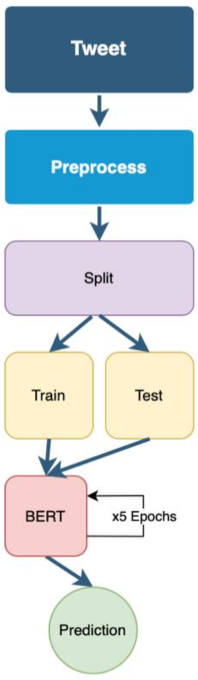


Figure 4: Steps followed on the second model

3.5 Second model: BERT

Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based machine learning technique for natural language processing (NLP) pretraining developed by (Devlin et al. 2018). *BERT-base-uncased* model is pretrained from unlabeled data extracted from BooksCorpus (Bandy and Vincent 2021) which have 800M of words and from English Wikipedia with 2,500M of words.

BERT uses Transformers (Wolf et al. 2019) as an attention mechanism that learns contextual relations between words (or sub-words) within a text. Transformers includes two separate mechanisms: an encoder that reads the text input and a decoder that produces a prediction of the label.

For this model, the binary relevance (BR) strategy (Luaces et al. 2012) was used, which splits the learning process of the dataset into a sets of binary classification tasks, in other words, one per label. The main disadvantage of this strategy is that BR ignores any label dependency and could fail in predicting some combination of labels that presents any dependency.

We have trained our second model with a batch of 32 instances and 5 epochs. Figure 4 represents the strategy of this model.

4 Experimental Setup

To obtain the above models, some libraries were used:

- For the data augmentation, the *nltk* library (Wang and Hu 2021, 1041-1049).
- For padding the sequences of the inputs id's, the *keras* library.
- *Sklearn* library for the metrics and splitting the dataset (Hao and Ho 2019).
- *Pandas* library for working with dataframes (Stepanek 2020).
- *Transformers* library for everything related with BERT.

For all the experiments, tweets were preprocessed and then they were randomly split using a stratified method (80% training and 20% testing). That means that the proportion of values in the samples produced is the same as the proportion of values provided for the parameter to stratify.

During the training phase of the competition, we have only focused on the macro-F1 score as the key metric of the Subtask B.

5 Results

Two submissions were sent using Bayesian Networks: the first one with the text processing v1.0 and the second one with the text processing v3.0. Table 3 shows the results obtained during the training phase.

Metrics	v1.0. Text processing	v3.0. Text processing
F1 Sarcasm	0.89	0.84
F1 Irony	0.26	0.20
F1 Satire	0.38	0.51
F1 Overstatement	0.17	0.48
F1 Understatement	0.47	0.47
F1 Rhetorical Question	0.12	0.20
Macro F1-Score	0.38	0.45

Table 3: Results obtained using Bayesian Networks

Metrics	v1.0	v2.0	v3.0	v4.0	v5.0
F1 Sarcasm	0.45	0.78	0.81	0.80	0.70
F1 Irony	0.45	0.72	0.77	0.77	0.64
F1 Satire	0.49	0.63	0.66	0.64	0.57
F1 Understatement	0.50	0.60	0.63	0.62	0.52
F1 Overstatement	0.49	0.65	0.69	0.71	0.60
F1 Rhetorical Question	0.84	0.90	0.91	0.89	0.87
Macro F1-Score	0.54	0.71	**0.75**	0.74	0.65

Table 4: Results obtained using BERT

Regarding the final submission using BERT-base-uncased, different approaches were used. Table 4 shows the results obtained during the training phase. The approaches were:

- **v1.0.** Nothing extra applied
- **v2.0.** Previous versions + Data Augmentation in minority class only.
- **v3.0.** Previous versions + insert data of an external database.
- **v4.0.** Previous versions + v2.0 of text processing.
- **v5.0.** Previous version + v3.0 of text processing.

Taking a look at Table 3 and Table 4, can be seen that in v3.0 of Table 4, the best macro F1-score is obtained. So that was our final submission.

According to the official metrics, as was mentioned before, we achieved a macro F1-score of 0.0699 and we were ranked 10th among 22 teams that participated on this subtask.

Analyzing our systems, we can state that the main problem found in the subtask was the lack of data towards unbalanced data at the dataset, which is why we have been constantly applying data augmentation on the minority classes and even inserting data from an external database. Applying these two techniques, a big improvement in the performance of our systems can be seen.

Furthermore, our research shows that any kind of preprocessing technique is mostly useless because any character, capital letter, overextended word or symbol, could be the determining factor in recognizing ironic speech.

6 Conclusion

In this paper our approach to solve *Task 6 (iSarcasmEval) – Subtask B: Given a text, determine which ironic speech category it belongs to, if any; in English*, has been described.

Our best result was reached with a deep learning algorithm (BERT) model, with which we achieved a macro F1-score of 0.0699. We obtained the 10th position in the ranking.

For future works, an improved version of our BERT model could be developed by training with a bigger dataset. It is also possible to look for new preprocessing techniques that enable the removal of information that is useless to the meaning of the tweet but still maintain the ironic speech patterns (if any).

References

Abu Farha, Ibrahim, Silviu Oprea, Steven Wilson, Walid Magdy. "SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic". In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics, 2022.

Carvalho, Paula, Luís Sarmento, Mário J. Silva, and Eugénio de Oliveira. 2009. "Clues for Detecting Irony in User-Generated Contents: Oh...!! It's 'so Easy' ;-)." In *Proceeding of the 1st International*

CIKM Workshop on Topic-Sentiment Analysis for Mass Opinion - TSA '09. New York, New York, USA: ACM Press.

Wallace, Byron C., Do Kook Choe, Laura Kertz, and Eugene Charniak. 2014. "Humans Require Context to Infer Ironic Intent (so Computers Probably Do, Too)." In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 512–16. Stroudsburg, PA, USA: Association for Computational Linguistics.

Leggitt, John S., and Raymond W. Gibbs. 2000. "Emotional Reactions to Verbal Irony." *Discourse Processes* 29 (1): 1–24. https://doi.org/10.1207/s15326950dp2901_1.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twit- ter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 107–116, Stroudsburg, PA, USA. Association for Computational Linguistics.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. ICWSM - a great catchy name: Semi-supervised recog- nition of sarcastic sentences in online product reviews. In William W. Cohen and Samuel Gosling, editors, *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010, Washington, DC, USA, May 23-26, 2010*. The AAAI Press.

Oprea, Silviu, and Walid Magdy. 2019. "ISarcasm: A Dataset of Intended Sarcasm." *ArXiv [Cs.CL]*. https://paperswithcode.com/paper/isarcasm-a-dataset-of-intended-sarcasm.

Alzahrani, Esam, and Leon Jololian. 2021. "How Different Text-Preprocessing Techniques Using the BERT Model Affect the Gender Profiling of Authors." *ArXiv [Cs.CL]*. http://arxiv.org/abs/2109.13890.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." CoRR abs/1810.04805.

Bandy, Jack, and Nicholas Vincent. 2021. "Addressing 'Documentation Debt' in Machine Learning Research: A Retrospective Datasheet for BookCorpus." *ArXiv [Cs.CL]*. http://arxiv.org/abs/2105.05241.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, et al. 2019. "HuggingFace's Transformers: State-of-the-Art Natural Language Processing." ArXiv [Cs.CL]. http://arxiv.org/abs/1910.03771.

Luaces, Oscar, Jorge Díez, José Barranquero, Juan José del Coz, and Antonio Bahamonde. 2012. "Binary Relevance Efficacy for Multilabel Classification." *Progress in Artificial Intelligence* 1 (4): 303–13. https://doi.org/10.1007/s13748-012-0030-x.

Wang, Meng and Fanghui Hu. 2021. "The Application of NLTK Library for Python Natural Language Processing in Corpus Research." Theory and Practice in Language Studies 11 (9): 1041-1049. doi:10.17507/tpls.1109.09.

Hao, Jiangang and Tin Kam Ho. 2019. Machine Learning made Easy: A Review of Scikit-Learn Package in Python Programming Language. Vol. 44. Los Angeles, CA: SAGE Publications. doi:10.3102/1076998619832248. https://journals.sagepub.com/doi/full/10.3102/1076998619832248

Stepanek, Hannah. 2020. Thinking in Pandas : How to use the Python Data Analysis Library the Right Way. Berkeley, CA: Apress. doi:10.1007/978-1-4842-5839-2. https://library.biblioboard.com/viewer/087e187a-b660-11ea-a44d-0a7fc7c4e64f.

McCrae, John Philip, Alexandre Rademaker, Francis Bond, Ewa Rudnicka, and Christiane Fellbaum. 2019. "English WordNet 2019 – An Open-Source WordNet for English." In *Proceedings of the 10th Global Wordnet Conference*, 245–52.

Ashwitha, Shruthi, Shruthi, Makarand Upadhyaya, Abhra Pratip Ray, y Manjunath. 2021. "Sarcasm Detection in Natural Language Processing". Materials Today: Proceedings 37: 3324–31. https://doi.org/10.1016/j.matpr.2020.09.124.

Khatri, Akshay, P, Pranav, y M, Dr. Anand Kumar. 2020. "Sarcasm detection in tweets with BERT and GloVe embeddings". https://doi.org/10.48550/ARXIV.2006.11512.

Heckerman, David, and Michael P. Wellman. "Bayesian networks." *Communications of the ACM*, 1995, 38(3), 27-31

NULL at SemEval-2022 Task 6: Intended Sarcasm Detection Using Stylistically Fused Contextualized Representation and Deep Learning

Mostafa Rahgouy
Department of Computer Science
Auburn University, Alabama, USA
mzr0108@auburn.edu

Hamed Babaei Giglou
Department of Computer Science
University of Tabriz, Tabriz, Iran
h.babaei98@ms.tabrizu.ac.ir

Taher Rahgooy
Department of Computer Science
University of West Florida, Florida, USA
trahgooy@students.uwf.edu

Cheryl D Seals
Department of Computer Science
Auburn University, Alabama, USA
sealscd@auburn.edu

Abstract

The intended sarcasm cannot be understood until the listener observes that the text's literal meaning violates truthfulness. Consequently, words and meanings play an essential role in specifying sarcasm. Enriched feature extraction techniques were proposed to capture both words and meanings in the contexts. Due to the overlapping features in sarcastic and non-sarcastic texts, a CNN model extracts local features from the combined class-dependent statistical embedding of sarcastic texts with contextualized embedding. Another component BiLSTM extracts long dependencies from combined non-sarcastic statistical and contextualized embeddings. This work combines a classifier that uses the combined high-level features of CNN and BiLSTM for sarcasm detection to produce the final predictions. The experimental analysis presented in this paper shows the effectiveness of the proposed method.

1 Introduction

Sarcasm detection is a specific case of sentiment analysis where the focus is on detecting sarcasm in text. Therefore, the task is to detect if a given text is sarcastic or not. According to (Hacker, 2011), sarcasm refers to the use of words that mean the opposite of what you want to say, especially to insult someone, show irritation, or provide humor. However, (Oprea and Magdy, 2020; Wilson, 2006) define sarcasm as a form of irony marked by a discrepancy between the literal and intended meanings of an utterance, through which the speaker usually manifests a hostile, derogatory, or contemptuous attitude. Generally, sarcasm tweets/texts are the utterances of a positive statement with harmful intent, and since the intent is hard to detect not only for computers but also for humans, it has attracted a considerable body of research in the natural language processing field to study opinions given by the users of online social media platforms such as Twitter, Facebook, Reddit, and Instagram. In the SemEval 2022 at Task 6: iSarcasmEval (Abu Farha et al., 2022), the goal is to identify intended sarcasm in both English and Arabic languages. This task consists of 3 subtasks, and we will focus on SubTask A defined as: *SubTask A: Given a text, determine whether it is sarcastic or non-sarcastic*.

Since sarcasm is an utterance of a statement with negative intent (e.g., *He has the best taste in music.*") and an admiring tone (e.g., "*She always makes dry jokes.*"). We hypothesized that contextualized representation might not be enough to represent intent when considering the sentiment analysis challenge. Therefore, we combined the representation of stylistically statistical and contextualized word embeddings using deep neural networks (DNNs) as a high-level feature extractor and classifier for the task. In this work, we studied the effect of the combination of stylistic and contextualized representation. The rest of the article is organized as follows. We first describe sarcasm studies in Section 2. Section 3 presents the proposed methodology. Sections 4 and 5 describe experimental setups and results. Moreover, in section 6 we made the discussion. Finally, in section 7 we conclude the article.

2 Related Works

Sarcasm detection research has seen a significant surge in interest in the past few years (Potamias et al., 2020; González et al., 2020; Babanejad et al., 2020; Gregory et al., 2020; Kumar et al., 2021; Ahuja and Sharma, 2021). At early stages, most works considered content as the only source for identifying sarcasm (Yaghoobian et al., 2021). To differentiate between sarcasm and non-sarcasm handcrafted rules and features have been applied (Veale and Hao, 2010; Bharti et al., 2015; Riloff et al., 2013). For instance, (Barbieri et al., 2014) used seven sets of lexical features to detect sarcasm by its inner structure. Frequency and Structure (i.e., length, punctuation, emoticons) are two examples of sarcasm features. However, the model based on these stylistic features shows promising results but lacks external knowledge (i.e., common sense). Our proposed method incorporates stylistic features with XLM-RoBERTa model to cope with the limitations of prior methods. Most recently, dominant approaches are Context-based, utilizing background knowledge and contextual dependencies as prior knowledge about the event mentioned in a sarcastic context (Li et al., 2020; Agrawal et al., 2020; Potamias et al., 2020). (Wallace et al., 2014) provided empirical evidence that to make judgments concerning ironic intent annotators require additional contextual information. Hazarika et al. is one of a few works that tried to capture the stylometric and personality features of the users and used user embeddings to encode stylometric and personality features of users combined by a content-based feature extractor (i.e., CNN)(Hazarika et al., 2018). This sarcasm detection model shows promising results on a large Reddit corpus. The improvement in results by this work inspired us to investigate the effectiveness of incorporating content-based features for SemEval-2022 Task 6: iSarcasmEval, but the problem we faced was the lack of data assigned to each user. As a result, instead of a users' unique features, we investigate the stylometric features of each class. In other words, we tried to find general styles of users who tried to write sarcastic texts by finding the distribution of their frequent words and other features provided in section 3. Results provided in section 5 empirically demonstrate the effectiveness of this incorporation. The effectiveness of transfer learning has given rise to a diversity of approaches, methodology, and practice, and they are also gaining attention in Sarcasm detection. BERT, RoBERTa, and XLNet are examples that deliver significant improvements. For example, (Potamias et al., 2020) proposed Recurrent CNN RoBERTa (RCNN-RoBERTa), which obtained %79 on the SARC dataset. Based on its effectiveness and multilingualism, we utilized the RoBERTa model for this research.

3 Model Description

In this section, we describe the details of our proposed deep learning model. The goal is to predict whether the given text is sarcastic or non-sarcastic. Figure 1 depicts an overview of our proposed method.

First, preprocessing (Appendix A.1) is applied to the raw texts. Next, character-based lower dimensionality statistical embedding (Char-LDSE) (Rangel et al., 2017; Giglou et al., 2021) captures the stylistic information about sarcastic and non-sarcastic data in form of class-dependent features based on the term frequency of tokens in sarcastic and non-sarcastic classes. The calculated LDSE features for sarcastic class in combination with contextualized representation are fed into Convolutional Neural Networks (CNNs) (LeCun et al., 1999) to extract local information of the texts. In another setting, the LDSE for non-sarcastic class in combination with contextualized representation is fed into Bidirectional Long Short-Term Memory (BiLSTM) (Schuster and Paliwal, 1997) to learn the long-dependent information in the texts. The features from CNN and BiLSTM layers combined for enhancing the feature extraction ability of the model. Extracted high-level features are fed into the classifier. In the following, we describe each component elaborately.

3.1 Representation

Preprocessed texts are fed into Char-LDSE and contextualized representations. Where it outputs an embedding for models.

Char-LDSE: First, a character n-gram matrix with TFIDF weight is created. Using TFIDF matrix the LDSE weighted probability of terms per class was obtained. As a result, $P(sarcastic)$ and $P(non-sarcastic)$ embeddings are calculated for sarcastic and non-sarcastic, respectively. The union of n-grams with TFIDF weighing on the training set is applied to achieve the input matrix of LDSE. We utilized character-level n-grams features with an n-gram range of $(2, 3)$, and word-level unigrams features for building input matrix to LDSE. For English character-level and word-level settings, and for Arabic, the only character-level setting is applied. As a result, we obtain the following matrix:

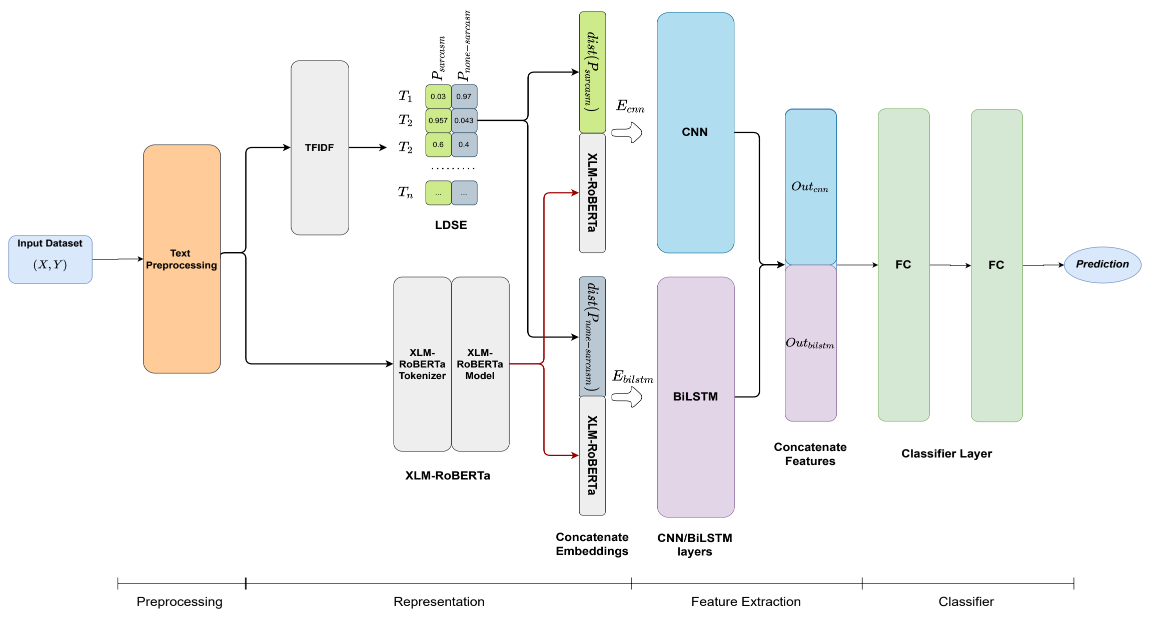


Figure 1: Architecture of Proposed Model

$$M = \left[\begin{array}{ccc|c} W_{11} & ... & W_{1n} & \beta(iS_{C_0}) \\ W_{21} & ... & W_{2n} & \beta(iS_{C_0}) \\ ... & ... & ... & ... \\ W_{(m-2)1} & ... & W_{(m-2)n} & \beta(iS_{C_1}) \\ W_{(m-1)1} & ... & W_{(m-1)n} & \beta(iS_{C_1}) \\ W_{m1} & .... & W_{mn} & \beta(iS_{C_1}) \end{array}\right]$$

Where each row in the matrix M represents a intended Sarcasm (iS) iS_i, each column represents vocabulary term T and W_{ij} represents its TFIDF weight and β represents the assigned class (C_1 - sarcastic, C_0 - non-sarcastic) of the iS text. Also, m and n represent the number of the training set and vocabulary size, respectively. To obtain the class-dependent term T weight embedding LDSE the following $LDSE(T, c)$ formula has been calculated:

$$LDSE(T, c) = \frac{\sum_{is \in \beta(iS_c)/c=\beta(iS_c)} W_{is,T}}{\sum_{is \in \beta(iS_c)} W_{is,T}}$$

$$\forall is \in iS, \; c \in \{C_0, C_1\}$$

Next, we calculated LDSE for each class for term T:

$$P_{sarcastic} = LDSE(T, c)$$

$$P_{non-sarcastic} = LDSE(T, c)$$

At the end, using $P_{sarcastic}$ and $P_{non-sarcastic}$ we extract the following representations:

$$feat(P_{sarcastic}) \; , \; feat(P_{non-sarcastic})$$

Where $feat(P)$ contains the set of features presented as followings:

$$feat(P) = \{max, min, std, avg, Q_1, ..., Q_{100}\}$$

Where $feat(P) \in \mathbb{R}^{104}$ is the class-dependent LDSE features. Accordingly, max, min, std, and avg are maximum, minimum, standard deviation, and average of weights P, respectively. Moreover, the $\{Q_1, ..., Q_{100}\}$ is the Q-th quantile of P's.

Contextualized Embedding: The XLM-RoBERTa (Conneau et al., 2020) is a pre-trained language model with the Masked Language Modeling (MLM) objective. The XLM-RoBERTa is a multilingual version of RoBERTa (Liu et al., 2019), which is pre-trained on 2.5TB of CommonCrawl data containing 100 languages. The model learns an inner representation of 100 languages that can then be used to extract features useful for downstream tasks. The function $CE \in \mathbb{R}^{768}$ is a base version of contextualized XLM-RoBERTa word embeddings. CE takes iS texts, uses XLM-RoBERTa tokenizer and XLM-RoBERTa model for word embedding generations.

$$CE(iS) := XLMRoBERTa(iS)$$

Concatenate Embeddings: In representations combination, we took the following approach. For CNN layer we combined $feat(P_{sarcastic})$ with $CE(iS)$, and for BiLSTM layer we combined $feat(P_{non-sarcastic})$ with $CE(iS)$. We obtained the following equations.

$$f\colon feat(P_{sarcastic}) \oplus CE(iS) \rightarrow E_{cnn}$$

$$f\colon feat(P_{non-sarcastic}) \oplus CE(iS) \rightarrow E_{bilstm}$$

Where $E_{cnn}, E_{bilstm} \in \mathbb{R}^{872}$, which will be the inputs of CNN and BiLSTM models in feature extraction module.

3.2 Feature Extraction

The CNN is applied on E_{cnn} and BiLSTM is applied on E_{bilstm} to extraction high-level features. Next, outputs of CNN and BiLSTM models are concatenated as an input to the classifier.

CNN Feature Extractor: CNN is a feature extraction technique with strong adaptability and is good at mining data local characteristics. Our CNN's has architecture as follows:

1. Two CNN layer with $Conv2D : conv2d \rightarrow maxpooling1d \rightarrow activation$ scheme.
2. Concatenate output of the two CNN layers: $Out_{cnn} = Conv2D_1 \oplus Conv2D_2$.
3. Dropout layer.

Where $Out_{cnn} \in \mathbb{R}^{200}$ with output channel number of 100 in both CNN layers. The first layer in CNN uses a kernel size of $(4, 872)$, padding of $(2, 0)$, and stride of 3. However, the second layer employs a kernel size of $(3, 872)$, padding of $(1, 0)$, and stride of 2. Next, for an activation function, the Gaussian Error Linear Unit (GELU) (Hendrycks and Gimpel, 2020) that nonlinearity weights inputs by their percentile, rather than gates inputs by their sign as in ReLUs (Agarap, 2019) is applied. It has been used in most of the transformers.

BiLSTM Sequence Learner: For a better representation of contextual information, single BiLSTM layer was employed. The BiLSTM is composed of two LSTM network that is capable of reading input texts in both directions, forward and backward. The forward LSTM processes information from left to right, and backward LSTM processes information vice versa and hidden state can be shown as:

$$\overrightarrow{h}_t = LSTM(E_{bilstm}, \overrightarrow{h}_{t-1})$$

$$\overleftarrow{h}_t = LSTM(E_{bilstm}, \overleftarrow{h}_{t+1})$$

Next, output of BiLSTM can be summarized as following where the max-pooling is applied to output h_t to retrieves a maximum value of each feature in the BiLSTM layer.

$$h_t = \overrightarrow{h}_t \oplus \overleftarrow{h}_t$$

$$Out_{bilstm} = MaxPool1d(h_t)$$

Where $Out_{bilstm} \in \mathbb{R}^{200}$.

Concatenate Features: As a final high-level feature, the Out_{cnn} and Out_{bilstm} outputs are concatenated as follows, Where $Out_{cnn \oplus bilstm} \in \mathbb{R}^{400}$.

$$Out_{cnn \oplus bilstm} = Out_{cnn} \oplus Out_{bilstm}$$

3.3 Classifier

The two fully connected layers use $Out_{cnnbilstm}$ for classification objectives. The first layer takes 400 neurons as input and outputs 200 neurons. Next, GELU activation is applied. Finally, the second layer takes 200 neurons and outputs 2 neurons (the prediction module).

4 Experimental Setup

Dataset: For sarcastic or non-sarcastic detection tasks, the collected dataset where the sarcasm labels for texts are provided by authors themselves, thus eliminating labeling proxies. For hyperparameter tuning and model selection we split the training set into train and dev sets with a 10% split rate for both English and Arabic languages. More information about the datasets (train, dev, and test sets) has been presented in Appendix A.2. The dataset is highly imbalanced.

Training Setup: The preprocessing (Appendix A.1) is applied to text, first. Next, using train and dev sets we made hyperparameters tuning (Appendix A.3). Due to the imbalanced setting of the data we made a data argumentation (Appendix A.4) as one of our experiments. The experimental design shows the positive effect of data augmentation. More information about preprocessing, hyperparameter setting, and data augmentation are described in the appendix section. As an evaluation metric to this task (SubTask A of iSarcasmEval), the main metrics for SubTask A is F1-score for the sarcastic class.

5 Results

Main Quantitative Findings: The table 1 presents the final results on the test set for English and Arabic languages. The main quantitative findings are:

Language	Accuracy	Precision	Recall	F1-Score	F1-Sarcastic	FP	FN	Rank
English	0.6136	0.5290	0.5558	0.4992	**0.2599**	436	105	28
Arabic	0.6671	0.5835	0.6600	0.5667	**0.3581**	396	70	18

Table 1: Evaluation results on test set for SubTask A.

- For English, the proposed model achieved 28th place among 43 teams, and 18th place among 32 teams for the Arabic.
- According to type I error (FP), the model for both languages wrongly predicts the high rate of non-sarcastic samples as sarcastic data. Regarding this phenomenon, the task suffers from type I error mostly.
- For the English language, among 200 sarcastic samples model captures 95 samples correctly, however, for the Arabic language, 130 samples were predicted correctly as sarcastic. About 65% TPR (True Positive Rate) of sarcastic samples in Arabic, but 47.5% of sarcastic samples in English were detected. It shows the task is more sensitive in English rather than Arabic.

Quantitative Analysis: We have used experimentations on the dev set to conclude the CNN-BiLSTM model as a final system for intended sarcasm detection at iSarcasmEval. The experiments are presented in Appendix A.5, and the table 6 shows experimental results. The quantitive analysis is presented as follows:

- The character-based n-gram with an n-gram range of $(2, 3)$ was implemented as a baseline representation with logistic regression, linear SVM, and multi-layer perceptron (MLP) as baseline models. The experimental results on both English and Arabic languages show neural networks (MLP) are performing well on this task.
- The experimentation of *LDSE + MLP* shows that stylistic representations are promising, but they are not enough.
- To find a candidate representation in combination with LDSE, as a performance booster, the XLM-RoBERTa masked language model is been considered. The *XLM-RoBERTa + MLP* shows that contextualized representation is very capable.
- The *Combination + MLP* model is combined LDSE and XLM-RoBERTa representation. The achieved result is promising for Arabic but it is 2% higher for English, this shows stylistic features in combination with contextualized representation perform well. The more complex model may boost the performance.
- The *Proposed Model (1)* uses preprocessing followed by the enriched representation with CNN-BiLSTM model. It is more capable than *Combination + MLP* model.
- Due to the unequal class distribution in the dataset, we employed weights in the cross-entropy loss function. The *Proposed Model (2)* results show its positive effects.
- We experimented with data augmentation techniques (Appendix A.4) and discovered that due to the word importance in achieving high-quality features for LDSE, data augmentation works more considerably well for this task (*Proposed Model (2) + Aug* model).
- The data augmentation probability shifts show the effects of LDSE representations (table 4). And considering *Proposed Model (2)* and *Proposed Model (2) + Aug*, We can observe that data augmentation affects mostly the stylistic features and results in boosting the performance. It represents that high-level feature extraction from a multi-space domain such as LDSE and XLM-RoBERTa play an important role in sarcasm detection.

6 Discussion

Type-1 Error: In either language, non-sarcastic samples are predicted wrongly as sarcastic. Type-1 error is the major source of error. Analysis of English FP samples indicated that most of the samples were in response to a post or reply to another person and indeed, with no information about the text context nor the writer profile, this indicates a lack of primordial clues for sarcasm detection. To make an improvement in reducing the error two approaches can be taken. First, consider more information about the text context such as the tread or post. Second, using a finer feature representation, since the lack of primordial clues leads to low-quality features. Moreover, the analysis revealed the following findings in the data which can be taken into account to improve the results by reducing the error (all of the findings come from the analysis of 436 FP samples for the English test set).

- We have found that samples in the dataset may be labeled wrongly as sarcasm so correcting them may boost the performance (e.g *"10 dec 2021, 4:46 am, All in on $PYR (28,07$)"* or *"pointing-downmedium-skin-tone-emoji this"*)

- A few samples were in single words (e.g *"Rubbish"*) which in a few cases such as *"pointing-downmedium-skin-tone-emoji this"* and *"Followed"* they didn't convey any meaning, tackling them in modelin could be more challenging since they don't have enought information for models.

- In a few cases preprocessing causes a low quality text generation (e.g *"10 dec 2021, 4:46 am, All in on $PYR (28,07$)"* will be converted into *'dec am all in on"* which is less valuable for modeling).

- Sometimes, emojis make the text sarcasm, so using them as features could be useful for this task since most of the comments in FP samples had emojis (almost 60% of them). In the whole preprocessing we converted emojis into texts, but in this way we thread the emojis similar to text, however, threading them in a different way may boost the performance.

Weighted Loss and TFIDF Data Augmentation Improvements: According to table 6, and model *Proposed Model (2)*, adding weighted loss cross-entropy increased results on the dev set by 1% in English, and 2% in Arabic. Similar to weighted loss cross-entropy, the model *Proposed Model (2) +Aug* in table 6 shows TFIDF data augmentation improved results on the dev set by 1% in English, and 4% in Arabic. Overall, weighted loss cross-entropy and TFIDF data augmentation together improved results by 2% in English, and 6% in Arabic. So using these techniques is promising in boosting the sarcasm detection models.

Task Sensitiveness in Languages: We proposed a class-dependent LDSE that considers character n-grams in stylistic information calculations. Character n-gram contains information on the more important tokens and the less important ones as well. So it captures the differences using token counts. However, The stress patterns of most Arabic dialects are broadly similar and appear regularly. Changes happen frequently in English, as word stress can change the lexical variety and meaning of the word (Watson et al., 2011). This affects both LDSE and contextualized representation.

Because LDSE is doing a probability-based calculation for tokens using a character n-gram matrix, the similarity in dialects will lead to a high-quality n-gram matrix in Arabic rather than English. Table 6 experimentation on the TFIDF matrix reveals this fact precisely. Accordingly, as a result of changes in the meaning, neural network feature extractors may have found some variant features which leads to misclassification (Type 1 error). The experimentations with *XLM-RoBERTa + MLP* (table 6) show that neural network models are performing better on Arabic contextualized representation rather than English. Experiments clearly show why this task is more sensitive in the English language than in Arabic.

7 Conclusion

This paper presented our approach for SemEval-2022 Task 6: iSarcasmEval: Intended Sarcasm Detection In English and Arabic. We investigated this problem by employing statistical and contextualized representations with deep learning techniques. We conducted the experimental and statistical analysis and presented the CNN-BiLSTM framework. The proposed studies in this paper show that considering stylistic features with deep learning frameworks is promising for intended sarcasm detection in both English and Arabic languages.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Abien Fred Agarap. 2019. Deep learning using rectified linear units (relu).

Ameeta Agrawal, Aijun An, and Manos Papagelis. 2020. Leveraging transitions of emotions for sarcasm detection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1505–1508.

Ravinder Ahuja and SC Sharma. 2021. Transformer-based word embedding with cnn model to detect sarcasm and irony. *Arabian Journal for Science and Engineering*, pages 1–14.

Nastaran Babanejad, Heidar Davoudi, Aijun An, and Manos Papagelis. 2020. Affective and contextual embedding for sarcasm detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 225–243.

Francesco Barbieri, Horacio Saggion, and Francesco Ronzano. 2014. Modelling sarcasm in twitter, a novel approach. In *proceedings of the 5th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 50–58.

Santosh Kumar Bharti, Korra Sathya Babu, and Sanjay Kumar Jena. 2015. Parsing-based sarcasm sentiment recognition in twitter data. In *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1373–1380. IEEE.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Hamed Babaei Giglou, Taher Rahgooy, Jafar Razmara Mostafa Rahgouy, and Zahra Rahgooy. 2021. Profiling Haters on Twitter using Statistical and Contextualized Embeddings—Notebook for PAN at CLEF 2021. In *CLEF 2021 Labs and Workshops, Notebook Papers*. CEUR-WS.org.

José Ángel González, Lluís-F Hurtado, and Ferran Pla. 2020. Transformer based contextualization of pre-trained word embeddings for irony detection in twitter. *Information Processing & Management*, 57(4):102262.

Hunter Gregory, Steven Li, Pouya Mohammadi, Natalie Tarn, Rachel Draelos, and Cynthia Rudin. 2020. A transformer approach to contextual sarcasm detection in twitter. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 270–275.

Hacker. 2011. Merriam-webster.com (8 may 2011).

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. *arXiv preprint arXiv:1805.06413*.

Dan Hendrycks and Kevin Gimpel. 2020. Gaussian error linear units (gelus).

Avinash Kumar, Vishnu Teja Narapareddy, Pranjal Gupta, Veerubhotla Aditya Srikanth, Lalita Bhanu Murthy Neti, and Aruna Malapati. 2021. Adversarial and auxiliary features-aware bert for sarcasm detection. In *8th ACM IKDD CODS and 26th COMAD*, pages 163–170.

Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. 1999. Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*, page 319, Berlin, Heidelberg. Springer-Verlag.

Meimei Li, Chen Lang, Min Yu, Yue Lu, Chao Liu, Jianguo Jiang, and Weiqing Huang. 2020. Scx-sd: Semi-supervised method for contextual sarcasm detection. In *International Conference on Knowledge Science, Engineering and Management*, pages 288–299. Springer.

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

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Silviu Oprea and Walid Magdy. 2020. isarcasm: A dataset of intended sarcasm.

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas-Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Francisco Rangel, Marc Franco-Salvador, and Paolo Rosso. 2017. A low dimensionality representation for language variety identification.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Tony Veale and Yanfen Hao. 2010. Detecting ironic intent in creative comparisons. In *ECAI 2010*, pages 765–770. IOS Press.

Byron C Wallace, Laura Kertz, Eugene Charniak, et al. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–516.

Janet CE Watson et al. 2011. Word stress in arabic.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743. Language in Mind: A Tribute to Neil Smith on the Occasion of his Retirement.

Hamed Yaghoobian, Hamid R. Arabnia, and Khaled Rasheed. 2021. Sarcasm detection: A comparative study.

Dataset	Sarcastic	Non-Sarcastic	Overall
English			
Train	772	2349	3121
Dev	95	252	347
Test	200	1200	1400
Arabic			
Train	678	2113	2791
Dev	67	244	311
Test	200	1200	1400

Table 2: English & Arabic dataset stats

Parameter	English	Arabic
Batch Size	8	8
Learning Rate	0.001	0.001
Epoch	10	10
Loss Function	CrossEntropy	CrossEntropy
Weighted Loss	Yes	Yes
Optimizer	Adam	Adam
Weight Decay	1e-9	1e-8
Dropout	0.6	0.4

Table 3: Hyperparameter setting

	English	Arabic
$Avg(P_{sarcastic})$	0.244	0.148
$Avg(P_{sarcastic})$+Aug	**0.327**	**0.213**
$Avg(P_{none-sarcastic})$	0.756	0.852
$Avg(P_{none-sarcastic})$+Aug	**0.673**	**0.787**
LDSE tokens	17691	16941
LDSE tokens + Aug	17961	17073
sarcastic samples	772	678
synthetic samples	760	678
sarcastic samples + Aug	1532	1356
overall training data + Aug	**3881**	**3469**

Table 4: Data augmenter stats

A Appendices

A.1 Preprocessing

Text preprocessing is often the first step in the pipeline of an NLP system. We consider the following preprocessing techniques to be applied to an input text:

- Lowercasing
- Punctuation removal
- Special character removal
- Demojify[1](converting emoji into texts)
- URL, #, @ removal
- Number removal

These preprocessing techniques were applied for both English and Arabic languages.

A.2 Train, Dev, and Test Sets

Each text in the datasets has a label specifying its sarcastic nature (sarcastic or non-sarcastic), provided by its author. The stats of the datasets are presented in table 2.

A.3 Hyperparameter Setting

For training DNNs there are multiple hyperparameters to be fine tuned. Batch size, learning rate, epoch, loss function, weight decay, dropout, wighted loss, and optimizer. The hyperparameters are described in the table 3 for both English and Arabic languages.

A.4 Data Augmentation

Data augmentation is a way to generate synthetic data for improving model performance without manual effort. Since we are dealing with an imbalanced binary classification problem and due to the importance of terms likelihood in LDSE we made a TFIDF data augmenter that learns the word preferences for samples in sarcastic class only, then generates new synthetic samples. For generating new samples we have followed the following steps:

1. Getting sarcastic samples from training data for data augmentation.
2. Preprocessing selected samples.
3. Training TFIDF data augmenter using *nlpaug* (Ma, 2019) a python library.
4. Augment sarcastic samples using data augmenter.
5. Generate new samples for sarcastic samples only.

The overall stats of the data augmentation are presented in the table 4. According to the mean average probability of terms in sarcastic and non-sarcastic, the data augmentation shifts the average probabilities in both classes. Since we were using the same set of data for argumentation so we observe the proper number of changes in tokens. Samples from data augmenter for both languages are presented in the table 5.

A.5 Experiments

The table 6 presents the experimental results for hyperparameter tunings. According to the table

[1]https://pypi.org/project/demoji/

ORG	mentally i m hiding in the walk in
AUG	mentally hiding jewelry the watto in
ORG	i love hour panic attacks
AUG	love limited spoiling attacks
ORG	العيون البصاصة تندب فيها رصاصة
AUG	العيون البصاصة تندب تضرب الأرض
ORG	عامل زى ابو الفراجين معندوش ...
AUG	عامل الحق متعالم الفراجين وغيره ...

Table 5: Data augmentation samples (ORG: original sample, AUG: augmented sample)

Model	English	Arabic
TFIDF + LR	0.16	0.56
TFIDF + LinearSVM	0.31	0.63
TFIDF + MLP	0.32	0.62
LDSE + MLP	0.27	0.58
XLM-RoBERTa + MLP	0.40	0.72
Combination + MLP	0.42	0.72
Proposed Model (1)	0.44	0.74
Proposed Model (2)	0.45	0.76
Proposed Model (2) + Aug	0.46	0.80

Table 6: Experimental results on dev set (F1-sarcastic is reported)

6, *Combination + MLP* refers to LDSE + XLM-RoBERTa Representation, *Proposed Model (1)* is the CNN-BiLSTM model and preprocessing, *Proposed Model (2)* is model with preprocessing, and weighted loss cross-entropy. *Proposed Model (2) + Aug* is a model with preprocessing, weighted loss cross-entropy and TFIDF data augmentation.

UoR-NCL at SemEval-2022 Task 6: Using ensemble loss with BERT for intended sarcasm detection

Emmanuel Osei-Brefo
University of Reading
White Knights Campus
Reading-UK
e.osei-brefo@student.reading.ac.uk

Huizhi Liang
University of Newcastle
Newcastle
UK
huizhi.liang@newcastle.ac.uk

Abstract

Sarcasm has gained notoriety for being difficult to detect by machine learning systems due to its figurative nature. In this paper, Bidirectional Encoder Representations from Transformers (BERT) model has been used with ensemble loss made of cross-entropy loss and negative log-likelihood loss to classify whether a given sentence is in English and Arabic tweets are sarcastic or not. From the results obtained in the experiments, our proposed BERT with ensemble loss achieved superior performance when applied to English and Arabic test datasets. For the validation dataset, our model performed better on the Arabic dataset but failed to outperform the baseline method (made of BERT with only a single loss function) when applied on the English validation set.

1 Introduction

In recent times, Social media platforms such as Twitter have turned out to be one of the most influential media for the expression of views, emotions, and information. To characterize sarcasm on Twitter; (Parmar et al., 2018) proposed the following: (a) strife between negative situation and positive sentiment, (b) strife between positive situation and negative sentiment, (c) Tweet starts with an interjection word, (d) likes and dislikes contradiction, (e) tweet conflicting ubiquitous facts, (f) tweets with positive sentiment and antonym pair, and (g) tweet contrasting facts that are time-sensitive.

With a huge amount of content being churned on social media and the need to analyze and detect sarcasm closely, text classification methods have been widely introduced to deal with these sophisticated tasks. Sarcasm has been shown to pose a major difficulty for sentiment analysis models (Liu, 2010), mainly because sarcasm acts as a form of verbal irony which enables the concealment of the true intention of denigration and negativity under a pretence of open a respectful representation such as; “The only thing I got from college is a caffeine addiction". As a result, the ability to detect sarcasm is crucial for understanding the real intents and beliefs of people (Maynard and Greenwood, 2014).

To address these issues and limitations, (Abu Farha et al., 2022) have organized the iSarcasmEval shared task for intended Sarcasm Detection In English and Arabic where different tweets in English and Arabic languages have to be classified as either sarcastic or not.

Several models for detection of sarcasm have been proposed in literature which incorporate statistical, machine learning, and rule-based approaches but predominantly (Mandal and Mahto, 2019). However, these techniques are not able to effectively perceive the figurative and ironic meaning of words (Joshi et al., 2016). Also, these methods require manually engineered features and are unable to understand the patterns in passive voice sentences (Bajwa and Choudhary, 2006). However, instead of utilizing manually engineered features, the incorporation of transformer models has the ability to automatically learn the important features.

In this paper, we present our participating system to the iSarcasmEval shared task (Abu Farha et al., 2022). In this work, we seek to apply a novel transformer-based approach to detect sarcastic statements in both English and Arabic languages and propose a novel approach to fine-tune the BERT model for sarcasm detection of both Arabic and English tweets using ensemble loss. BERT is a model that is gaining increasing popularity due to its outstanding performance for multiple NLP tasks.

Recently, the language model called BERT has been widely used in several tasks. The pre-trained model can generate word/token or sentence representations that are enriched with prior knowledge. Then, they can be fine-tuned specifically for many downstream tasks such as sarcasm detection. The traditional BERT model only uses one loss function whilst our proposed model which combines a cross-entropy loss with a negative log-likelihood function leads to a better penalization of incorrect predictions that the model is confident about more than incorrect predictions the model is less confident about. This leads to better detection of sarcasm due to an improved F-1 sarcastic.

One of the main contributions of this work is the use of ensemble loss with BERT (Devlin et al., 2019) to address the task of sarcasm detection of tweets in both English and Arabic languages with improved sarcasm detection ability.

The rest of the paper is organized as follows: The Related Work in section 2 discusses the research works which are closely related to the current study. Section 3 contains the system description part and highlights the description of the proposed approach and its working methodology. The dataset used for experiment setup is provided in the Experiments section in section 4, whilst the results and Discussion segment can be found under section 4.1 and contains the experimental results and the performance metrics. The conclusion and future work for this research work are all captured in section 5.

2 Related Work

The conversation aspect in sarcasm detection was investigated by (Ghosh et al., 2017), and sarcastic and non-sarcastic tweets were collected to create a dataset for their experiments. BERT, which is a transformer-based machine learning model has been exploited to provide a deep contextual representation of words and used for sarcasm detection. They are pre-trained language models and have been also been widely utilized in many NLP domains. These models have been proven to be effective since they are enriched with knowledge from the pre-training resources (Avvaru et al., 2020). (Moores and Mago, 2022) recently surveyed automated sarcastic detection on Twitter and found out that there was a shift towards the use of deep learning methods due to the benefit of using a model with induced instead of discrete features combined with the innovation of transformers. BERT and GloVe embeddings were combined with machine learning classifiers to detect sarcasm in tweets in the work of (Khatri and P, 2020). Semi-supervised techniques were used by (Tsur et al., 2010) to detect sarcasm Sentences in Online Product Reviews whilst (Amir et al., 2016) also deployed the use of automatic learning and exploiting word embeddings to recognize sarcasm. Other approaches such as Bi-Directional Gated Recurrent Neural Network (Bi-Directional GRNU) have also been used to detect sarcasm (Zhang et al., 2016). Bert has been used with a crowd layer on tweets with noisy labels to achieve improved results (Osei-Brefo et al., 2021).

3 System description

For each sample in the English and Arabic dataset, a tweeted text and a given token were first concatenated in the following format:

[CLS] + Tweeted text + [SEP] sarcastic sentence [SEP]

where '[CLS]' token was added for classification and two '[SEP]' tokens were used to identify the sarcastic nature of the tweet. Each sentence was initially organized, after which the '[CLS]' and '[SEP]' tokens were added to their start and end. The tokenized sentence was then truncated or padded to a maximum length of 64. Each generated token was then mapped to its individual IDs to create an attention mask for each sentence. A hidden size of 768 was utilized for the BERT, whilst 120 and 2 were respectively used as the hidden size of the classifier and the number of labels. A one-layer feed-forward classifier was then instantiated after which the last hidden state of the token '[CLS]', was extracted for classifying whether a statement was sarcastic or not.

Binary Classification for detecting sarcasm To fine-tune the models for sub-task A, a fully-connected layer was added on top of the pooled output (the sequence embedding from the pre-trained model). This layer had an output size of 2 with a softmax activation function which outputs the probability of each class as either 1 or 0. For sub-task A, the baseline model used was a Bert model with the same parameters as our proposed model except that it was made up of a single cross-entropy loss function. A cross-entropy loss and Negative log-likelihood loss functions were combined in different weights proportion to form the ensemble loss used as a loss function for fine-tuning. These proportional weight functions were 0.8/0.2 and 0.2/0.8 for the cross-entropy loss and Negative log-likelihood loss functions respectively as can be seen in Tables 1 and 2 as **BERT Ensemble loss A** and **BERT Ensemble loss B** respectively. The final prediction of each sentence was made by selecting the class with the maximum probability. Due to the difficulties inherent nature of detecting sarcasm by machines, a higher threshold of about 70 % was used as the minimum probability beyond which a tweet can be classified as being sarcastic for both the English and Arabic datasets.

Ensemble loss function Two loss functions were combined together in such a way that the benefits derived from each loss function were embedded in the model. These loss functions were cross-entropy loss and negative log-likelihood loss. The weights of these combinations were 0.8 to 0.2 and 0.2 to 0.8 for BERT Ensemble loss A and BERT Ensemble loss B respectively. On the other hand, the Baseline model loss was made up of only the cross-entropy loss.

4 Experiments

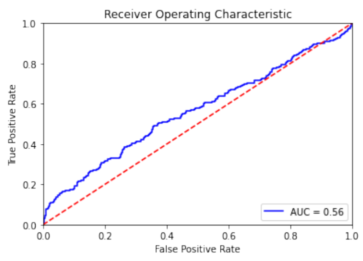
Experiments were conducted on the respective English and Arabic training and test sets provided by the task organizers. The training set of the English language had 3467 samples whilst that of the Arabic language had 3120 data samples. During the development stage, the training datasets were divided into 90% and 10% splits sets respectively for the training and validation datasets. The test sets for both languages on the other hand had a total of 1,400 data samples used in the evaluation phase.

All the models were fine-tuned by using the Adam optimizer for 12 epochs with the batch size 32 and the learning rate of 0.000005 and 0.000000001 as the default epsilon value. For sub-task A, the models were evaluated by Accuracy, Recall, false positives, F-1 sarcastic, Accuracy, and F1-macro.

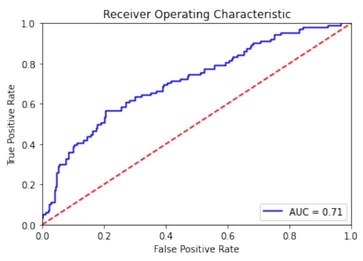
4.1 Results and Discussion

Table 1 shows the results of the baselines and our approaches in both sub-task A using the Validation dataset. Table 2 also shows the results obtained using the test data sets

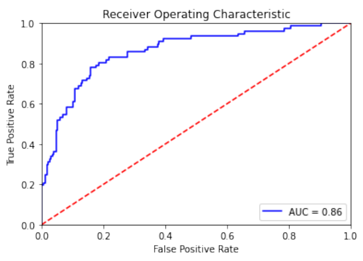
Our proposed model with the ensemble loss had different performance strengths on the validation dataset and the test datasets. Their relative performance also varied depending on the particular Language they were applied to. Among the 5 metrics used, the most effective metric that can be used to determine how accurate a model is able to detect sarcastic statements is the F-1 sarcastic metric. For the validation dataset, the baseline model outperformed our proposed model in all the 5 metrics used when applied to the English Language dataset. For the Arabic dataset, our model outperformed the baseline in all the other 3 metrics used with the exception of the false positive and f1-macro metrics where our model could not outperform the baseline. It is worth noting that our model had an F-1 sarcastic metric of 83.53% when it was applied to the Arabic language test dataset. For the test dataset, our proposed model outperformed the baseline model for both the English languages with an



(a) Roc curve illustrating the performance of the baseline Bert model on test dataset English tweets

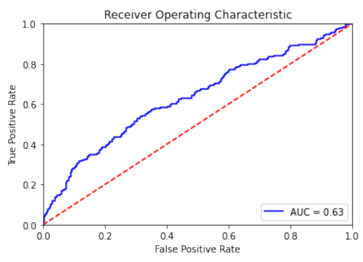


(b) Roc curve illustrating the performance of the Bert model on validation dataset English tweets

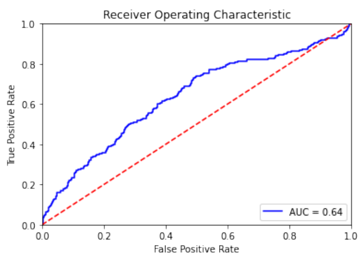


(c) Roc curve illustrating the performance of the Bert model on Test dataset Arabic tweets

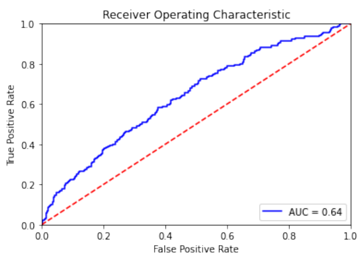
Figure 1: Comparison of the proposed approach performance on on sub task A



(a) Roc curve illustrating the performance of the Bert Ensemble loss model on test dataset English tweets



(b) Roc curve illustrating the performance of the baseline Bert model on test dataset Arabic tweets



(c) Roc curve illustrating the performance of the ensemble Bert model on Test dataset Arabic tweets

Figure 2: Comparison of the proposed approach performance on on sub task A

Val set	Model	Sub-task A				
		FP	Recall	F-1 sarcastic	F1-macro	Acc
English	Baseline	**37.04**	**58.54**	**71.84**	**63.76**	**74.06**
	BERT Ensemble loss A	37.15	55.03	67.51	57 .00	72.33
	BERT Ensemble loss B	42.43	54.87	58.82	41.48	70.89
Arabic	Baseline	**23.60**	64.92	83.35	**77.75**	83.28
	BERT+ Ensemble loss A	25.61	**65.75**	**83.53**	77.41	**83.92**
	BERT Ensemble loss B	26.82	61.62	64.61	42.94	75.24

Table 1: Results of validation dataset for sub-task A, where BERT+ Ensemble loss A and BERT+ Ensemble loss B represents 80% /20% and 20% /80% combination of the cross entropy loss and negative log likelihood loss respectively. Where FP represents False positives

Test set	Model	Sub-task A				
		FP	Recall	F-1 sarcastic	F1-macro	Acc
English	Baseline	46.59	**52.66**	79.37	54.88	81.00
	BERT Ensemble loss A	**38.86**	52.25	**79.75**	**58.00**	80.07
	BERT Ensemble loss B	46.65	51.51	79.12	46.15	**85.71**
Arabic	Baseline	38.86	**52.25**	79.74	**57.99**	80.07
	BERT+ Ensemble loss A	**37.22**	51.28	76.10	56.60	73.79
	BERT+Ensemble loss B	40.13	52.61	**79.12**	46.15	**85.71**

Table 2: Results of Test dataset for sub-task A, where BERT+ Ensemble loss A and BERT+ Ensemble loss B represents 80% /20% and 20% /80% combination of the cross entropy loss and negative log likelihood loss respectively. Where FP represents False positives

F-1 sarcastic metric of 79.75% Its application on the Arabic language test set yielded an F-1 sarcastic metric of 79.12%, which was just below the value obtained from the baseline model which was 79.74%

Figures 1 and 2 show the Receiver Operating Characteristic (ROC) Curve of the different models applied to the test and validation data sets, which is a plot of the true positive rates against false-positive rates. It graphically shows and compares the performance of the baseline model and our ensemble loss model at all classification thresholds. It pictorially shows the evaluation of the strength of a model; with the bigger area under the curve indicating superior model performance and the smaller area under the curve signalling poorer model classification performance respectively.

5 Conclusion

This work has proposed the use of ensemble loss on the BERT model to detect whether a given sentence in English or Arabic language is sarcastic or not. The results indicate that the use of our ensemble loss on the Bert Model exhibits superior performance over the baseline models when applied to the English and Arabic test datasets. Future work will involve the undertaking of further investigation to find the optimal proportion of each combination that yields the optimal results. It will also involve the use of ensemble loss on Arabic-based Bert models such as Ara-Bert and other Bert-Based models to find their performance. Different types of losses will also be investigated and added to the ensemble loss portfolio in order to explore the different types with superior performance. Since the two language datasets used were unbalanced in nature, future work will also explore other data balancing techniques to help improve the results of our model.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mário J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *CoRR*, abs/1607.00976.

Adithya Avvaru, Sanath Vobilisetty, and Radhika Mamidi. 2020. Detecting Sarcasm in Conversation Context Using Transformer-Based Models. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 98–103, Online. Association for Computational Linguistics.

Imran Sarwar Bajwa and Muhammad Abbas Choudhary. 2006. A rule based system for speech language context understanding. *Journal of Donghua University (English Edition) Vol*, 23(06).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding.

Debanjan Ghosh, Alexander Richard Fabbri, and Smaranda Muresan. 2017. The role of conversation context for sarcasm detection in online interactions. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 186–196, Saarbrücken, Germany. Association for Computational Linguistics.

Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection?

Akshay Khatri and Pranav P. 2020. Sarcasm detection in tweets with BERT and GloVe embeddings. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 56–60, Online. Association for Computational Linguistics.

Bing Liu. 2010. Sentiment analysis and subjectivity. In *Handbook of Natural Language Processing, Second Edition. Taylor and Francis Group, Boca*.

Paul K Mandal and Rakeshkumar Mahto. 2019. Deep cnn-lstm with word embeddings for news headline sarcasm detection. In *16th International Conference on Information Technology-New Generations (ITNG 2019)*, pages 495–498. Springer.

Diana Maynard and Mark Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

Bleau Moores and Vijay Mago. 2022. A survey on automated sarcasm detection on twitter.

Emmanuel Osei-Brefo, Thanet Markchom, and Huizhi Liang. 2021. UOR at SemEval-2021 task 12: On crowd annotations; learning with disagreements to optimise crowd truth. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1303–1309, Online. Association for Computational Linguistics.

Krishna Parmar, Nivid Limbasiya, and Maulik V. Dhamecha. 2018. Feature based composite approach for sarcasm detection using mapreduce. *2018 Second International Conference on Computing Methodologies and Communication (ICCMC)*, pages 587–591.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. Icwsm - a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *ICWSM*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2449–2460, Osaka, Japan. The COLING 2016 Organizing Committee.

I2C at SemEval-2022 Task 6: Intended Sarcasm Detection on Social Networks with Deep Learning

Pablo González Díaz
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
pablo.gonzalez682@alu.uhu.es

Pablo Cordón Hidalgo
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
pablo.gonzalez682@alu.uhu.es

Jacinto Mata Vázquez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
mata@uhu.es

Victoria Pachón Álvarez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
vpachon@uhu.es

Abstract

In this paper we present our approach and system description on iSarcasmEval: a SemEval task for intended sarcasm detection on social networks. This derives from our participation in *SubTask A: Given a text, determine whether it is sarcastic or non-sarcastic*. In our approach to complete the task, a comparison of several machine learning and deep learning algorithms using two datasets was conducted. The model which obtained the highest values of F1-score was a BERT-base-cased model. With this one, an F1-score of 0.2451 for the sarcastic class in the evaluation process was achieved. Finally, our team reached the 30th position.

1 Introduction

Sarcasm is a form of mockery intended to imply the opposite or to express displeasure. It can be classified as irony, satire, understatement, overstatement, rhetorical question and sarcasm itself. Social media platforms allow people to express themselves and speak about several different topics via text, emojis and multimedia. Many companies collect information about people's opinions to aid in marketing decision-making about their products.

Sarcasm detection is relevant in data analysis because it can be wrongly interpreted due to its nature, becoming potentially harmful to different computational systems. For example, if a computer system understands the sentence "*The only thing I got from college is a caffeine addiction*", its literal meaning would not be understood, as it refers to how stressful college can be.

Defining a phrase as sarcastic is a really challenging task to solve for the text mining community because there are many details that need to be considered such as the writer's personality and the sentence context. For instance, depending on the ideology of the person, "*Trump, that respectful man*" could be sarcastic or not.

In iSarcasmEval: Task 6, SubTask A (Abu Farha et al. 2022) participants had to decide if a particular tweet includes a sarcastic connotation or not.

2 Background

Two sample datasets were used to train the model.

The first one was the original dataset without any preprocessing. This dataset is composed of 3467 rows, with 2600 0-value rows (non-sarcastic) and 867 1-value rows (sarcastic).

The second sample dataset (Tweet – Rephrase) consists of a phrase and rephrase contrast in which the aim was to adequately learn the differences between a phrase with a sarcastic connotation and that same sentence expressing the same thing but in a literal way. This dataset is composed of 1734 rows: 867 1-value rows and 867 0-value rows, where these last ones are rephrases related to each tweet taken as a 1-value row.

So, these two datasets follow the same structure:

- tweet: *"See Brexit is going well"*
- sarcastic: "1"

In the second dataset, the column "rephrase" was added to the list of tweets. The rephrase example for the tweet below would be:

- tweet: "*Brexit really isn't going to plan.*"
- sarcastic: "0"

Table 1 shows some examples followed by both datasets.

The issue of sarcasm detection has been addressed by various authors in recent years. In the last several years, researchers have been working on a technique to analyze social media data in order to identify possible undisclosed information so it can be useful to assess new patterns and make important decisions through it (Shah and Shah 2021, 247-259). Sentiment analysis is also used for sarcasm detection (Suzuki et al. 2017), dividing a sentence into phrases and judging the situation and the sentiment separately.

In (Verma, Shukla, and Shukla 2021), the authors present a review of methodologies and techniques of sarcasm detection. They concluded that deep learning is the most common technique to identify sarcasm.

3 System overview

In this section, the models used to solve the sarcasm detection issue will be described.

Due to the nature of the problem, a decision was made not to carry out any preprocessing. The reason is that we think that sarcasm interpretation requires the original phrase to be considered in its entirety, so letter cases, emojis and punctuation were kept.

3.1 Models

A total of 30 models were used to train both datasets. In the first experiments, machine learning techniques such as LinearSVC, DecissionTrees and RandomForests using CountVectorizer and TFIDF (Pedregosa, Fabian and others 2011) were used. We also trained some deep learning techniques as LSTM Neural Networks (Hochreiter and Schmidhuber 1997) and Transformers (Vaswani et al. 2017).

For LSTM Neural Network we implemented the LSTM Simple Neural Network and we used the BERT model (Devlin et al. 2018) for Transformers.

In Table 2, the results of machine learning techniques mentioned before can be seen.

3.1.1 BERT: Bidirectional Encoder Representations from Transformers

BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning both the left and right contexts in all layers. As a result, the pretrained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question-answering and language inference, without substantial task-specific architecture modifications.

It uses Transformer, which is a simple network architecture based on attention mechanisms, dispensing with recurrence and convolutions entirely.

There are two ways to read a sentence for BERT. On one hand we have BERT-base-cased, in which BERT differences between capital letters and uppercase letters and, on the other hand, BERT-base-uncased, which will take the sentence and make all single letters uncased. As mentioned above, we did not miniscule the text of the tweets

Tweet	**sarcastic**
"Nice to be compared to a brick wall"	1
"Not happy i have been compared to a brick wall"	0
"Social Care for the young is basically a bath board and bed rest and your done"	1
"Social care for the young is non existent"	0
"I've been doing physics for 40 minutes I think I deserve a break"	1
"I'm never going to get it all done at this rate"	0

Table 1: Tweet examples

Dataset	Model		Metrics	
			Accuracy	F1-Score (class 1)
Original dataset	CountVectorizer	LinearSVC	0.68	0.28
		DecissionTree	0.74	0.10
		RandomForest	0.74	-
	TFIDF	LinearSVC	0.71	0.22
		DecissionTree	0.74	0.10
		RandomForest	0.74	-
Tweet – Rephrase	CountVectorizer	LinearSVC	0.60	0.61
		DecissionTree	0.56	0.37
		RandomForest	0.65	0.61
	TFIDF	LinearSVC	0.58	0.60
		DecissionTree	0.57	0.37
		RandomForest	0.61	0.61

Table 2: Results obtained in training phase using machine learning techniques

to preserve the sarcastic meaning of the sentence. Therefore, the BERT-base-cased was used.

The BERT model was trained with a batch size of 32 instances and 5 epochs by using a training/validation split.

3.1.2 LSTM Simple Neural Network

Long Short-Term Memory (LSTM) is a well-known recurrent neural network architecture. The most important feature of recurrent networks is their ability to persist introduction loops information in its diagram, so it can remember previous states and use this information in subsequent stages.

In order to find better results, a pretrained word vector with a size of 200d was added to our model. GloVe (Pennington, Socher, and Manning 2014) is a global log-bilinear regression model for the unsupervised learning of word representations that outperforms other models on word analogy, word similarity and named entity recognition tasks.

For this approach, a batch size of 32 instances, and 10 epochs was used. Besides, we implemented an early stopping to improve the performance.

4 Experimental setup

Many libraries were imported to develop the experiments. Some of them are: “NumPy” (Harris et al. 2020)”, a fundamental package for scientific computing; “spaCy” (Honnibal and Montani 2017), a library for advanced natural language processing; “NLTK” (Loper and Bird 2002), a tool-kit to work with human language information; “Keras” (Chollet 2015), a neural-network library; “scikit-learn”, which contains simple and efficient tools for predictive data analysis and “Pandas” (McKinney 2010), used for manipulating data.

For all the experiments, the datasets were split using 80% to train and 20% to test. We use a stratify approach.

Finally, when the test dataset was released, the whole train dataset was used to train the final model through BERT-base-cased, which was the best algorithm to complete the task.

The metrics obtained to evaluate our model are accuracy and F1-score.

5 Results

Table 3 shows the results obtained in our experiments during the training phase. The algorithm that reached the best results was BERT-base-cased with the use of the Tweet - Rephrase dataset, obtaining a score of 0.86 accuracy and F1-score.

Results shown in the leaderboard with the test dataset yielded a 0.2451 F1-score. Finally, our team reached the 30th position.

6 Conclusions

In this paper our approach to solve Task 6 (iSarcasmEval) - SubTask A: Given a text,

Dataset	**Model**	**Metrics**	
		Accuracy	**F1-Score (class 1)**
Original dataset	LSTM Simple Neural Network	0.65	0.38
	BERT-base-cased	0.72	0.43
	BERT-base-uncased	0.69	0.39
Tweet – Rephrase	LSTM Simple Neural Network	0.65	0.62
	BERT-base-cased	**0.86**	**0.86**
	BERT-base-uncased	0.84	0.84

Table 3: Results obtained in training phase using deep learning techniques

determine whether it is sarcastic or non-sarcastic, in English, has been described.

The main idea was to check models of deep learning algorithms such as BERT or LSTM Neural Network trained with the dataset Tweet – Rephrase. After training and analyzing each model and comparing deep learning with machine learning techniques, BERT-base-cased model obtained higher results F1-Score and accuracy.

In the future, a BERT model will be trained using a larger corpus and more innovative techniques such as the application of sentiment analysis in broken-down sentences.

References

Abu Farha, Ibrahim, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. "SemEval-2022 Task 6: iSarcasmEval Intended Sarcasm Detection in English and Arabic."Association for Computational Linguistics, .

Chollet, Francois and others. 2015. "Keras." . https://github.com/fchollet/keras.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." . https://arxiv.org/abs/1810.04805.

Harris, Charles R., K. Jarrod Millman, van der Walt, Stéfan J., Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, et al. 2020. "Array Programming with NumPy." Nature 585: 357–362. doi:10.1038/s41586-020-2649-2.

Hochreiter, Sepp and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." Neural Computation 9 (8): 1735-1780. doi:10.1162/neco.1997.9.8.1735. https://doi.org/10.1162/neco.1997.9.8.1735.

Honnibal, Matthew and Ines Montani. "spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing.".

Loper, Edward and Steven Bird. 2002. "NLTK: The Natural Language Toolkit." Philadelphia, Pennsylvania, Association for Computational Linguistics, . doi:10.3115/1118108.1118117. https://doi.org/10.3115/1118108.1118117.

McKinney, Wes and others. 2010. "Data Structures for Statistical Computing in Python."Austin, TX, .

Pedregosa, Fabian and Varoquaux, Ga\el and Gramfort, Alexandre and Michel, Vincent and Thirion, Bertrand and Grisel, Olivier and Blondel, Mathieu and Prettenhofer, Peter and Weiss, Ron and Dubourg, Vincent and others. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12 (Oct): 2825-2830.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. "Glove: Global Vectors for Word Representation."01. doi:10.3115/v1/D14-1162.

Shah, Bhumi and Margil Shah. 2021. "A Survey on Machine Learning and Deep Learning Based Approaches for Sarcasm Identification in Social Media."Springer Singapore, .

Suzuki, Shota, Ryohei Orihara, Yuichi Sei, Yasuyuki Tahara, and Akihiko Ohsuga. 2017. "Sarcasm Detection Method to Improve Review Analysis.".

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

Verma, Palak, Neha Shukla, and A. P. Shukla. 2021. "Techniques of Sarcasm Detection: A Review.". doi:10.1109/ICACITE51222.2021.9404585.

BFCAI at SemEval-2022 Task 6: Multi-Layer Perceptron for Sarcasm Detection in Arabic Texts

Nsrin Ashraf and **Fathy Elkazaz** and **Mohamed Taha** and **Hamada Nayel**
Department of Computer Science, Benha University, Benha, Egypt
{nisrien.ashraf19,fathy.elkazzaz}@fci.bu.edu.eg
{mohamed.taha, hamada.ali}@fci.bu.edu.eg
Tarek Elshishtawy
Department of Information System, Benha University, Benha, Egypt
t.shishtawy@fci.bu.edu.eg

Abstract

This paper describes the systems submitted to iSarcasm shared task. The aim of iSarcasm is to identify the sarcastic contents in Arabic and English text. Our team participated in iSarcasm for the Arabic language. A multi-Layer machine learning based model has been submitted for Arabic sarcasm detection. In this model, a vector space TF-IDF has been used as for feature representation. The submitted system is simple and does not need any external resources. The test results show encouraging results.

1 Introduction

Analyzing social media becomes a crucial task, due to the frequently usage of social media platforms. Sarcasm detection, the conflict of using the verbal meaning of a sentence and its intended meaning (CLIFT, 1999; Lee and Katz, 1998), is an important task. Sarcasm detection is a challenge, since sarcastic contents are used to express the opposing of what is being said. Recently sarcasm detection has been studied from a computational perspective as one of classification problems that separates sarcastic from non-sarcastic contents(Reyes et al., 2013; Nayel et al., 2021).

Arabic is an important natural language having an extensive number of speakers. The research in Natural Language Processing (NLP) for Arabic is continually increasing. However, there is still a need to handle the complexity of NLP tasks in Arabic. This complexity arises from various aspects, such as orthography, morphology, dialects, short vowels, and word order. Sarcasm detection in Arabic is a particularly challenging task (Alayba et al., 2018).

In this paper, we describe the system submitted to the iSarcasm detection shared task(Abu Farha et al., 2022). The shared task aims at detecting the sarcasm contents in Arabic tweets. In this work, a machine learning framework has been developed and various machine learning algorithms have been implemented. Term Frequency-Inverse Document Frequency (TF-IDF) has been used as vector space model for tweet representation. The rest of this paper is organized as follows: in section 2, a background about sarcasm detection is given. Section 3 and section 4 overview the dataset and the system respectively. Experimental setup and results are given in section 5 and section 6 respectively. Finally, section 7 concludes the proposed work and suggests future work to be continued.

2 Background

The research work have been done on Arabic sarcasm detection were mainly focused on creating datasets and establish a baseline for each created dataset (Ghanem et al., 2020). Karoui et al. (2017) created a corpus of sarcastic Arabic tweets that are related to politics. Distant supervision has been used for the creation of corpus. The authors used keywords that are like sarcastic contents in Arabic to label the tweets as sarcastic tweets. They implemented different machine learning algorithms such as SVM, logistic regression, Naïve Bayes, and other classifiers on the developed corpus.

An ensemble classifier of XGBoost, random forest and fully connected neural networks has been designed by Khalifa and Hussein (2019). They extracted a set of features that consists of sentiment and statistical features, in addition to word n-grams, topic modelling features and word embeddings. Nayel et al. (2019) developed an ensemble-based system for irony detection in Arabic Tweets. A set of classification algorithms namely Random forests, multinomial Naïve Bayes, linear, and SVM classifiers have been used as base-classifiers. In (Nayel et al., 2021), sarcasm detection has been formulated as a binary classification problem and SVM has been implemented.

3 Dataset

A new data collection method has been introduced, where the sarcasm labels for texts are provided by the authors themselves. The author of each sarcastic text rephrased the text to convey the same intended message without using sarcasm. Leggitt and Gibbs (2000) defined a set categories of ironic speech namely; sarcasm, irony, satire, understatement, overstatement, and rhetorical question. Linguistic experts have been asked to further label each text into one of these categories. Each text in the Arabic dataset has the following information attached to it:

- a label specifying the text dialect;
- a label specifying the nature of sarcasm (sarcastic or non-sarcastic), provided by its author;
- a rephrase provided by its author that conveys the same message non-sarcastically.

4 System Overview

In this section, we review the main structure of the proposed model. The proposed system, as shown in figure 1, consists of three phases namely; preprocessing, feature extraction and training the classification algorithms. Then, the resulted model used to predict the unseen test data.

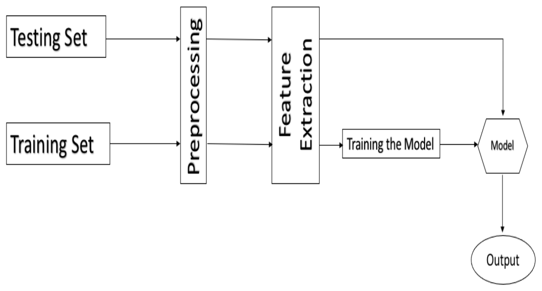


Figure 1: The structure of the proposed model

4.1 Preprocessing

The first stage of developing systems is preprocessing, where unwanted and uninformative piece of text has to be removed, it is also called text cleaning. We performed text cleaning by removing:

- special symbols, such as $\{+, -, =, \$, ...\}$;
- repeated characters such as ("*hhhhhhhh*" will be normalized to "*hh*");
- non-Arabic words, such as English characters or any other language;
- punctuations and Arabic diacritics.

4.2 Features Extraction

To prepare features to build classification model and before feeding the text into the classifier and after performing text cleaning, Term Frequency-Inverse Document Frequency (TF-IDF) technique was used to change over content to vectors and all the algorithms to investigate the best performing algorithm.

TF-IDF has been used to represent comments as vectors. If $< w_1, w_2, \dots, w_k >$ are the tokenized words of a comment $\mathcal{T}_j$, the vector associated to the comment $\mathcal{T}_j$ will be represented as $<v_{j1}, v_{j2}, \dots, v_{jk}>$ where v_{ji} is the weight of the token w_i in tweet $\mathcal{T}_j$ which is calculated as:-

$$v_{ji} = tf_{ji} * \log\left(\frac{N+1}{df_i + 1}\right)$$

where tf_{ji} is the total number of occurrences of token w_i in the comment $\mathcal{T}_j$, df_i is the number of comments in which the token w_i occurs and N is the total number of comment.

4.3 Methodology

We explored various classification algorithms as well as ensemble approach by combining the output of these classifiers (also known as base classifiers) using hard voting. The base classifiers used in this work are listed below:

- Support Vector Machines (SVMs)
- Random Forest (RF)
- K-Nearest Neighbours (KNN)
- Multinomial Naïve Bayes (M-NB)
- Multi-Layer Perceptron (MLP)
- Stochastic Gradient Descent (SGD)
- AdaBoost Classifier
- Voting Classifier

Table 1: 5-fold Cross-Validation accuracy for all classifiers in the training set

Classifier	Accuracy	STD
SVM-Linear	81.0%	0.055
SVM-RBF	76.6%	0.003
MNB	76.6%	0.005
SGD	80.0%	0.045
MLP	83.6%	0.045
RF	75.8%	0.056
KNN	79.7%	0.058
AdaBoost	75.2%	0.052
Voting	80.4%	0.043

5 Experimental Setup

For feature extraction phase we used unigram model. For the purpose of training the model, we have used 5-fold cross-validation technique to adjust the parameters.

The Scikit-Learn library implementation of classification algorithms were used in the training phase. For SVM, two kernels have been tested: linear kernel and RBF with two parameters $\gamma = 2$ and $C = 1$. While, for SGD classifier the loss function used was Hinge and the maximum iteration was set at 10000 iterations.

The number of nodes in the hidden layer of MLP was set at 20, logistic function was used as activation function and Adam solver was used. The maximum number of decision trees in random forests is set at 300.

5.1 Evaluation Metrics

F1-score has been used to evaluate the performance of all submissions. F1-score is a harmonic mean of Precision (P) and Recall (R) and calculated as follow:

$$F\text{–}score = \frac{2 * P * R}{P + R}$$

F1-score for the sarcastic class (F1-sarcastic) has been used for final evaluation.

6 Results

The cross validation accuracy of all training classifiers for the training set is given in Table 1. It is clear that MLP gives the best accuracy with moderate Standard Deviation (STD) for the five folds while development phase, so we decided to submit the output of this classifier.

Table 2: Results of MLP classifier for the test set

Measure	Value	Rank
F-1 sarcastic	0.3746	14
F-score	0.6024	11
Precision (P)	0.5968	15
Recall (R)	0.6608	17
Accuracy	0.7329	8

Results for test set is given in Table 2. The reported results show that, while training MLP gives better accuracy among implemented machine learning classifiers. Also, it gives better rank in accuracy for the unlabelled test set. While in other metrics, the performance was not satisfied. This may resulted because of using accuracy metric while comparing different classifiers in development phase.

A good suggestion is to use different evaluation metrics while developing the system. In addition, using different word representation models such as word embeddings, which encompasses the semantic meaning of words could improve the performance. Complex models such as deep learning models is promising, but the challenge in such models is the availability of suitable resources.

7 Conclusion

In this work, a classical machine learning framework has been designed for sarcasm detection in Arabic tweets. The proposed framework reported reasonable results. The future work may include applying complex framework such as deep learning structure. In addition, word representation is very important factor that can be used in different manner such as word embeddings and transformers.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Abdulaziz M. Alayba, Vasile Palade, Matthew England, and Rahat Iqbal. 2018. Improving sentiment analysis in arabic using word representation. *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*.

REBECCA CLIFT. 1999. Irony in conversation. *Language in Society*, 28(4):523–553.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Paolo Rosso, and Véronique Moriceau. 2020. Irony detection in a multilingual context. In *Advances in Information Retrieval*, pages 141–149, Cham. Springer International Publishing.

Jihen Karoui, Farah Banamara Zitoune, and Veronique Moriceau. 2017. Soukhria: Towards an irony detection system for arabic in social media. *Procedia Computer Science*, 117:161–168.

Muhammad Khalifa and Noura Hussein. 2019. Ensemble learning for irony detection in arabic tweets. In *Working Notes of FIRE 2019 - Forum for Information Retrieval Evaluation, Kolkata, India, December 12-15, 2019*, volume 2517 of *CEUR Workshop Proceedings*, pages 433–438. CEUR-WS.org.

Christopher J. Lee and Albert N. Katz. 1998. The differential role of ridicule in sarcasm and irony. *Metaphor and Symbol*, 13(1):1–15.

John S Leggitt and Raymond W Gibbs. 2000. Emotional reactions to verbal irony. *Discourse processes*, 29(1):1–24.

Hamada Nayel, Eslam Amer, Aya Allam, and Hanya Abdallah. 2021. Machine learning-based model for sentiment and sarcasm detection. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 386–389, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Hamada A. Nayel, Walaa Medhat, and Metwally Rashad. 2019. Benha@idat: Improving irony detection in arabic tweets using ensemble approach. In *Working Notes of FIRE 2019 - Forum for Information Retrieval Evaluation, Kolkata, India, December 12-15, 2019*, volume 2517 of *CEUR Workshop Proceedings*, pages 401–408. CEUR-WS.org.

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Lang. Resour. Eval.*, 47(1):239–268.

akaBERT at SemEval-2022 Task 6: An Ensemble Transformer-based Model for Arabic Sarcasm Detection

Abdulrahman Mohamed Kamr and **Ensaf Hussein Mohamed**
Research Support Center in Computing and Informatics
Department of Computer Science,
Faculty of Computers and Artificial Intelligence,
Helwan University, Cairo, Egypt.
abdokamr94@fci.helwan.edu.eg, ensaf_hussein@fci.helwan.edu.eg

Abstract

Due to the widespread usage of social media sites and the enormous number of users who utilize irony implicit words in most of their tweets and posts, it has become necessary to detect sarcasm, which strongly influences understanding and analyzing the crowd's opinions. Detecting sarcasm is difficult due to the nature of sarcastic tweets, which vary based on the topic, region, the user's attitude, culture, terminologies, and other criteria. In addition to these difficulties, detecting sarcasm in Arabic has its challenges due to its complexities, such as being morphologically rich, having many different dialects, and having low resources. In this research, we present our submission of (iSarcasmEval) sub-task A of the shared task on SemEval 2022. In Sub-task A; we determine whether the tweets are sarcastic or non-sarcastic. We implemented different approaches based on Transformers. First, we fine-tuned the AraBERT, MARABERT, and AraELECTRA. One of the challenges that faced us was that the data was not balanced. Non-sarcastic data is much more than sarcastic. We used data augmentation techniques to balance the two classes, significantly affecting the performance. The performance F1 score of the three models was 87%, 90%, and 91%, respectively. Then we boosted the three models by developing an ensemble model based on hard voting. The final performance F1 Score was 93%.

1 Introduction

According to the Cambridge dictionary definition of Sarcasm, it is the use of remarks that mean the opposite of what they say, made to hurt someone's feelings or humorously criticize something, Like when you say Love this weather. (When the weather is horrible).

Sarcasm detection is determining whether or not a piece of text is sarcastic. Sarcasm is a significant challenge for sentiment analysis systems. This is because a sarcastic sentence usually contains an implicit negative sentiment that is expressed with positive expressions. This discrepancy between the surface and intended sentiments creates a difficult challenge for sentiment analysis systems.

Sarcasm detection is a difficult task for a variety of reasons. First of all, there aren't many labeled data resources for sarcasm detection. Moreover, any available texts that can be collected (for example, Tweets) contain many issues, such as an evolving dictionary of slang words and abbreviations, so it usually takes many hours for human annotators to prepare the data for any potential use. Furthermore, the nature of sarcasm detection adds to the task's difficulty, as Sarcasm can be considered relative and varies significantly between people, and it depends on many factors such as the topic, region, time, the events surrounding the sentence, and the readers/writers mentality and the; in other words, a sentence that one person finds sarcastic may sound normal to another. (Farha and Magdy, 2021)

In addition to these previous challenges, discovering irony in the Arabic language has its own set of challenges due to the complexities of the language, such as being formally rich, different dialects, lack of resources, and rapid development due to the inclination of the Arabic language. The Arab citizen makes fun of all his affairs, especially politics, which uses many words and terms that are implicit in it.

Transformers greatly assisted in significant advancements in NLP tasks. They are a new neural network that does not employ convolution or recursion. They instead use their attention to find correlations between words in the text. Transformers can process text in parallel, allowing them to learn much faster than sequential methods. They also outperform previous methods in terms of results.

Transformer-based language models have recently proven to be highly efficient at language understanding, giving promising results across various NLP tasks and benchmark datasets. The language modeling capability of these models aids in capturing the literal meaning of context-heavy texts. For Arabic NLP in particular, the best results for sentiment analysis are currently achieved by AraBERT, a language model proposed by (Antoun et al., 2020).

Despite recent advances, detecting sarcasm remains a difficult task because of implicit, indirect phrasing and the symbolic nature of language. When working with Twitter data, the task becomes even more difficult because the social media posts are often short and contain noise sources, code-switching, and the use of nontraditional dialectal variations. Furthermore, BERT-based models have struggled with rare words, which are more common in social media texts due to their informal nature and the prevalence of slang words. It is difficult for language models like AraBERT, which have been trained on structured corpora from Wikipedia.(Hengle et al., 2021)

In this study, we tackle (iSarcasmEval) sub-task A of the shared task on SemEval 2022.In SubTask A; we determine whether the tweets are sarcastic or non-sarcastic. We have proposed models based on transformers to discover the sarcasm that transformers have proven to excel in other NLP tasks such as sentiment analysis. We summarize what has been accomplished in this research in the following:

- Fine-tuning the state-of-the-art transformers-based models such as AraBERT, AraELECTRA, and MARBERT.

- One of the challenging problems in this task is that the dataset is unbalanced, with 2357 non-sarcastic tweets and 745 sarcastic. We solved this problem by using augmentation and balancing the two classes.

- We proposed an ensemble model based on hard voting between the three fine-tuned transformers, and it outperforms each of them.

The following sections are organized as follows; Section 2 presents the background, section 3 describes the task and dataset description, section 4 gives an overview of the proposed system, section 5 explores the results and discussion, and section 6 concludes and gives possible directions for the future.

2 Background

There are few attempts to work on Arabic sarcasm. The workshops in the field of Natural language processing for the Arabic language revived the interest in detecting sarcasm, as this workshop provided annotated datasets, which was considered a challenging obstacle in front of researchers. Such as the previous shared tasks on irony detection(Ghanem et al., 2019) along with the participants' submissions and dialectal sarcasm datasets by (Abbes et al., 2020); (Farha and Magdy, 2021);(Abu Farha et al., 2021).

This survey mainly focuses on the WANLP 2021 workshop and its shared task on sarcasm in Arabic. This shared task seeks to promote and draw attention to Arabic sarcasm detection, which is critical for improving performance in other tasks such as sentiment analysis. The dataset used in this collaborative task, ArSarcasm-v2, comprises 15,548 tweets that have been labeled for sarcasm, sentiment, and dialect. Subtask 1 on sarcasm detection received 27 submissions. The majority of approaches relied on using and fine-tuning pre-trained language models like AraBERT and MARBERT (Abdel-Salam, 2021); (Abuzayed and Al-Khalifa, 2021); (Alharbi and Lee, 2021);(Bashmal and AlZeer, 2021);(Faraj et al., 2021);(Gaanoun and Benelallam, 2021);(Hengle et al., 2021);(Husain and Uzuner, 2021);(Israeli et al., 2021);(Naski et al., 2021);(Wadhawan, 2021). Deep learning and traditional machine learning approaches were used by a few of the participants(Nayel et al., 2021).

The BhamNLP(Alharbi and Lee, 2021) team was ranked best in the sarcasm detection task, with an F1-Score 0.6225. They deployed a multi-task learning architecture trained for sarcasm and sentiment classification in their approach. The model is based on both a MARBERT and a CNN-LSTM model, with each model's output being concatenated and supplied to the final output layer. The CNN-LSTM utilized both word and character embeddings.

3 Task And Dataset Description

In SemEval 2022 Task 6: iSarcasmEval (Intended Sarcasm Detection In English and Arabic)(Abu Farha et al., 2022). The organizer offered two datasets, Arabic and English. There are three subtasks. SubTask A its aim is to determine whether the given text is sarcastic or non-sarcastic; SubTask B is applied in (English only): it aims to determine which ironic speech category the given

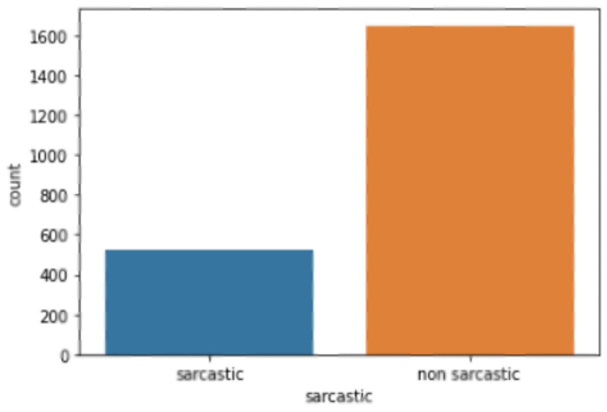


Figure 1: The distribution of the original dataset.

text belongs to, if an. It's a binary multi-label classification task. SubTask C: it aims to determine which text is the sarcastic one; if it is given two texts that convey the same meaning,
we have to clarify that our proposed system is a solution to Subtask A on the offered Arabic dataset. The Arabic dataset consists of 3,102 tweets with 2357 non-sarcastic tweets and 745 sarcastic ones. The test set contains 1400 tweets. It was released without labels for evaluation purposes. The dataset is provided with dialects and sarcasm labels. Table 1 shows some statistics of the released training set. Figure 1 shows the distribution of sarcastic and non-sarcastic tweets. Table 2 shows a sample of sarcastic and non-sarcastic tweets.

4 System Overview

The proposed model comprises two main phases: Data augmentation and the Ensemble module, consisting of three pre-trained transformers(AraBERT, MARABERT, AraELCTRA) and the Voting module, as shown in this Figure 2.

4.1 Data Augmentation

One of the challenging problems in this dataset is its small size, and it's unbalanced. The number of non-sarcastic tweets is 2357, while sarcastic is 745. We used data augmentation to balance the minority classes, Sarcastic. We used the NLPAug tool for augmentation. It is a python library based on non-contextual embeddings like Glove, Word2Vec, etc. And also for contextual embeddings like BERT and RoBERTa. By inserting or replacing words using word embedding, we used AraBERT for that purpose. Tables 3 shows a sample of tweets before and after augmentation.

4.2 Ensemble Module

This module consists of two phases. Firstly, apply three of the state of art pre-trained transformers, then use an ensemble hard voting technique.

(I) **Transformer Based Models:**

Deep learning methods have shown promising results in many machine learning domains, including natural language processing, computer vision and speech recognition. Due to architectures inspired by the human brain, deep learning techniques have recently outperformed traditional machine learning methods in terms of performance. Most deep learning techniques in the context of NLP use word vector representations to represent textual inputs (Mikolov et al., 2013a),(Mikolov et al., 2013b). These traditional techniques are being replaced by transformer-based techniques and significantly improve most NLP tasks, such as classification. As a result of the pre-training process, transformer-based techniques can generate efficient word embedding, making them powerful language models.

- **AraBERT** (Antoun et al., 2020) Pre-trained to handle Arabic text, AraBERT is a language model that is inspired by Google's BERT architecture. Six variants of the same model are available for experimentation: AraBERTv0.2-base, AraBERTv1- base, AraBERTv0.1-base, AraBERTv2-large, AraBERTv0.2-large, and AraBERTv2-base.
- **AraELECTRA**
(Antoun et al., 2020) With reduced computations for pre-training the transformers, ELECTRA is a method aimed toward the task of self-supervised language representation learning. ELECTRA models are inspired by the two primary components of Generative Adversarial Networks: generator and discriminator. They aim at distinguishing between real input tokens and fake ones. These models have shown convincing state-of-the-art results on Arabic QA data.

- **MARBERT**
provided by (Abdul-mageed et al., 2020). These models are based on the BERT-

Table 1: Distribution of tweets among different dialects

SARCASTIC/DIALECT	EGYPT	GULF	LEVANT	MAGHREB	MSA	NILE	TOTAL
NON-SARCASTIC	472	67	81	12	1470	255	2357
SARCASTIC	0	17	35	77	49	567	567

Table 2: Sample of tweets with Dialect and classified to sarcastic and non-sarcastic.

DIALECT	SARCASTIC	NON-SARCASTIC
Egypt	- -	احمد الشيخ احسن لاعب في الدورى و المصرى احسن كوره لحد دلوقتى Ahmed Al-Sheikh is the best player in the league and Al-Masry team is the best football so far.
Gulf	الف مبروك الانتصار لفريقنا، مدرب الفريق الثاني ماخذ وضعية سوي نفسك ميت ! Congratulations to our team for wining , the coach of the another team didn't take anyting expect the week breath	اغنية رامي عياش اللي بحكي فيها على عيني و راسي والله عنك لحظة ما أتخلى كتير حلوة Ramy Ayash song's which is about on my eyes and my head I swear I will not let you it's awesome
Levant	مافي حدا مثله يضحك عليك There is no one like him laughs of you	اللهم انتقم منه انتقم من كل من فرح بدماء المسلمين حسرك على ابناءك Oh God, take revenge on him, take revenge on everyone who rejoiced in the blood of Muslims
Magreb	واحد قال لمو بغيت نتزوج بوحدة بيضاء وطويلة وخدامة قالت له خد الثلاجة Someone said to her mom: I wanted to marry a white woman and tall and serve me, she told him to take the refrigerator instade	طاح القدر يا ليبيا عارفين OH MY GOD, Libya, you know
MSA	رياض محرز ينقذ سمكة من الغرق، الله على اخلاقك يا فخر العرب Riyad Mahrez saves a fish from drowning, may God bless your morals, the best arabian player	بعد غيبة و طول انتظار ها قد عاد الاستقرار من جديد :)) النور قطع الهرم After an absence and a long wait, stability has returned again :)) The light cut agine
Nile	ضبط شخص بدبلوم انتحل صفة طبيب The arrest of a person with a diploma who pretended to be a doctor without getting in faculty of medicine	دراكولا لو كان عايش معانا لغاية دلوقت كان زمانه بيمص قصب If Dracula had been living with us until now, his time would have been sucking a cane

Table 3: Sample of the original and augmented tweets.

Original Text	Augmented Text
الليله دي دموع تماسيح	دي دموع تماسيح
ياعم مبارك دا جلده و مش بيطلع منه جنيه	ياعم دا جلده و مش بيطلع جنيه
الشحات مين له نص دستور البلد يا بلد	الشحات له نص البلد يا بلد

base and trained on a set of books and news articles. AraBERT was trained on 66GB of text-only news articles. MARBERT was trained on a larger dataset (128 GB), 50% of which is tweets. The variation in MARBERT's training data gives it the ability to handle better the variations in colloquial Arabic, which is very useful for sarcasm detection

(II) **Voting Module:** The three classifier outputs are fed into the voting module based on hard voting. It selects the majority prediction, whether sarcastic or non-sarcastic. For example, the ensemble module predicts sarcastic if two models predicted sarcastic, as shown in Figure 3.

5 Results and Discussion

5.1 Performance Evaluation Metrics

We used a variety of metrics to evaluate the proposed model quality. To evaluate the model's performance, we have to compute its Accuracy, Recall, Precision, and F1-Score. The equations for these evolution metrics are as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

5.2 Experimental Results

We applied two experiments, one without augmentation and the other with augmentation. We split the data into 60% for training, 20% for validation, and 20% for testing.

- **First experiment,** After splitting, we fine-tuned the three pre-trained transformers AraBRT, MARBERT, AraELECTRA on the original data and the ensemble model. The results show that F1-Score for AraBERT, AraELECTRA, MARABERT, and the Ensemble model were 74%, 69%, 77%, 77% respectively. As shown in Table 4.

- **Second experiment,** after data augmentation, and the number of tweets in sarcastic and non-sarcastic are equal. Experiment 2 shows that F1-Score for AraBERT, AraELECTRA, MARABERT, and the Ensemble model were 87%, 90%, 91%, 93% respectively. As we see in Table 5, there is an improvement in the performance of ARAELECTRA, MARABERT, and the Ensemble model.

  The results show that our proposed model ranked sixth on the official competition. F1-score for sarcastic tweets is 0.4438, while F1-score for non-sarcastic tweets is 0.6222.

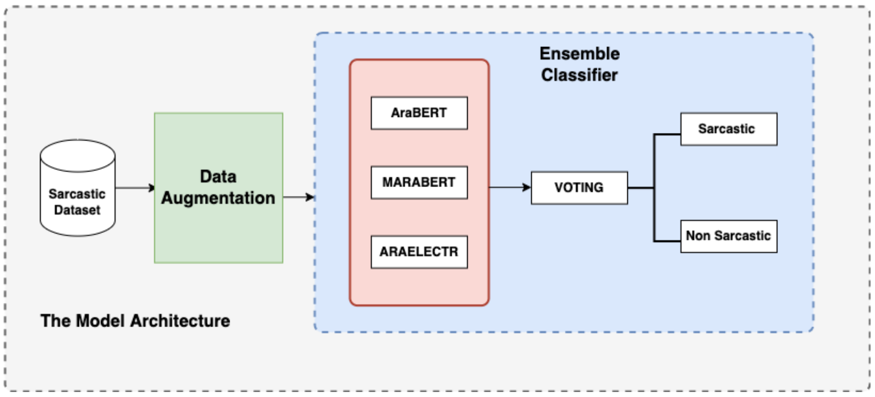


Figure 2: The proposed model architecture

Table 4: AraBert, MARBERT, ARAELECTRA, and Ensemble models on the original data.

Models	Precision	Recall	Accuracy	Macro-F1 score	F1 sarcastic
ARABERT	83%	84%	88%	83%	74%
ARAELECTRA	84%	79%	87%	81%	69%
MARABERT	89%	83%	90%	86%	77%
ENSEMBLE	88%	84%	90%	85%	77%

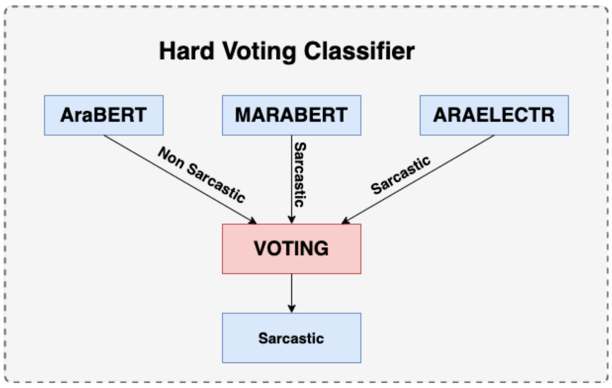


Figure 3: Hard voting Module.

6 Conclusion

We presented our submission on the shared Subtask A on Arabic sarcasm detection iSarcasmEval 2022 to tackle the problem of detecting sarcasm in Arabic. We explored different pre-trained models based on BERT. We noticed that the performance of AraELECTRA, MARABERT, and the Ensemble model is greatly improved after data augmentation and balancing the sarcastic and non-sarcastic tweets. The best submission model was the ensemble model; it applies hard voting on the AraBERT, MARABERT, and AraELCTRA. In the future, we plan to improve the performance of the proposed model by understanding and identifying which features are essential that contribute to enhancing the model prediction and troubleshooting unexpected model outputs.

References

I. Abbes, W. Zaghouani, O. El-Hardlo, and F. Ashour. 2020. Daict: A dialectal arabic irony corpus extracted from twitter. In *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings, May*, page 6265–6271.

R. Abdel-Salam. 2021. Wanlp 2021 shared-task: Towards irony and sentiment detection in arabic tweets using multi-headed-lstm-cnn-gru and marbert. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 306–311.

M. Abdul-mageed, A. Elmadany, E. Moatez, and B. Nagoudi. 2020. Arbert marbert : Deep bidirectional transformers for arabic.

I. Abu Farha, W. Zaghouani, and W. Magdy. 2021. Overview of the wanlp 2021 shared task on sarcasm and sentiment detection in arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 296–305.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Table 5: AraBert, MARBERT, ARAELECTRA, and Ensemble models on the augmented data.

Models	Precision	Recall	Accuracy	Macro-F1 score	F1 sarcastic
ARABERT	87%	87%	86%	86%	87%
ARAELECTRA	90%	90%	90%	90%	90%
MARABERT	91%	91%	91%	91%	91%
ENSEMBLE	93%	93%	93%	93%	93%

A. Abuzayed and H. Al-Khalifa. 2021. Sarcasm and sentiment detection in arabic tweets using bert-based models and data augmentation. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 312–317.

A.I. Alharbi and M. Lee. 2021. Multi-task learning using a combination of contextualised and static word embeddings for Arabic sarcasm detection and sentiment analysis. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 318–322.

W. Antoun, F. Baly, and H. Hajj. 2020. Arabert: Transformer-based model for arabic language understanding.

L. Bashmal and D. AlZeer. 2021. Arsarcasm shared task: An ensemble bert model for sarcasmdetection in arabic tweets. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 323–328.

D. Faraj, D. Faraj, and M. Abdullah. 2021. Sarcasmdet at sarcasm detection task 2021 in arabic using arabert pretrained model. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 345–350.

I.A. Farha and W. Magdy. 2021. Benchmarking transformer-based language models for arabic sentiment and sarcasm detection. In *Arabic Natural Language Processing Workshop*, page 21–31.

K. Gaanoun and I. Benelallam. 2021. Sarcasm and sentiment detection in arabic language a hybrid approach combining embeddings and rule-based features. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, volume Figure 1, page 351–356.

B. Ghanem, J. Karoui, F. Benamara, V. Moriceau, and P. Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in arabic tweets. In *ACM International Conference Proceeding Series*, page 10–13.

A. Hengle, A. Kshirsagar, S. Desai, and M. Marathe. 2021. Combining context-free and contextualized representations for arabic sarcasm detection and sentiment identification.

F. Husain and O. Uzuner. 2021. Leveraging offensive language for sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 364–369.

A. Israeli, Y. Nahum, S. Fine, and K. Bar. 2021. The idc system for sentiment classification and sarcasm detection in arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 370–375.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013b. Distributed representations ofwords and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, page 1–9.

M. Naski, A. Messaoudi, H. Haddad, M. BenHajhmida, C. Fourati, and A. Ben Elhaj Mabrouk. 2021. In *iCompass at Shared Task on Sarcasm and Sentiment Detection in Arabic. Proceedings of the Sixth Arabic Natural Language Processing Workshop*, page 381–385. [link].

H. Nayel, E. Amer, A. Allam, and H. Abdallah. 2021. Machine learning-based model for sentiment and sarcasm detection. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop, Ml*, page 386–389.

A. Wadhawan. 2021. Arabert and farasa segmentation based approach for sarcasm and sentiment detection in arabic tweets.

AlexU-AL at SemEval-2022 Task 6: Detecting Sarcasm in Arabic Text Using Deep Learning Techniques

Aya Lotfy, Marwan Torki and Nagwa El-Makky
Computer and Systems Engineering Department
Alexandria University
Alexandria, Egypt
eng-aya.lotfy1419, mtorki, nagwa.elmakky@alexu.edu.eg

Abstract

Sarcasm detection is an important task in Natural Language Understanding. Sarcasm is a form of verbal irony that occurs when there is a discrepancy between the literal and intended meanings of an expression. In this paper, we use the tweets of the Arabic dataset provided by SemEval-2022 task 6 to train deep learning classifiers to solve the sub-tasks A and C associated with the dataset. Sub-task A is to determine if the tweet is sarcastic or not. For sub-task C, given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which is the sarcastic one. In our solution, we utilize fine-tuned MARBERT (Abdul-Mageed et al., 2021) model with an added single linear layer on top for classification. The proposed solution achieved 0.5076 F1-sarcastic in Arabic sub-task A, accuracy of 0.7450 and F-score of 0.7442 in Arabic sub-task C. We achieved the 2^{nd} and the 9^{th} places for Arabic sub-tasks A and C respectively.

1 Introduction

Sarcasm is ubiquitous phenomenon on the social web, and is difficult to be analysed automatically and manually by humans because of its nature. Sarcasm data can be very confusing to computer systems which use it to perform tasks such as sentiment analysis, opinion mining, author profiling, and harassment detection (Liu, 2012; Rosenthal et al., 2014; Maynard and Greenwood, 2014; Van Hee et al., 2018).

(Rosenthal et al., 2014) show that the sentiment polarity classification performance on non-sarcastic tweets is much better than on sarcastic ones, in the context of SemEval. Sentiment polarity classification is used widely in industry, driving marketing, administration, and investment decisions (Hassan Yousef et al., 2014). So it is important to create models for sarcasm detection.

A comparatively small dataset is a challenge we faced when working on the Arabic dataset that makes it difficult to train complex models. We used transfer learning to treat this issue. By using a transfer learning, a pre-trained model for some task on a large dataset can be used as a starting point in another task which improves the performance. It is used in a wide range of natural language processing (NLP) tasks.

Word embeddings such as word2vec (Mikolov et al., 2013), FastText (Joulin et al., 2016) and Glove (Pennington et al., 2014) can be used to initialize vectors learnt form large dataset. Recently, Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) became the most popular NLP approach to transfer learning. Google AI Language team pretrained BERT model and fine-tuned it for a large range of tasks, such as question answering and language inference where it achieved state-of-the-art performance. MARBERT is built using the same network architecture as $BERT_{Base}$ (Devlin et al., 2019), without the next sentence prediction (NSP). MARBERT is trained on a large Twitter dataset. Therefore, we utilize fine-tuned MARBERT to solve this challenge.

This paper is organized as follows. In Section 2, we discuss relevant related works in sarcasm detection. We describe our proposed system in Section 3. The models implementation details are explained in Section 4. Section 5 describes the dataset for the shared task. Then we report and analyze the evaluation results in Section 6. Finally, we provide our conclusions in Section 7.

2 Related Work

A weak supervision and manual labelling methods can be used for the annotation process. A weak supervision method is to consider the texts sarcastic, if they meet predefined criteria, like including specific tags (e.g. #sarcasm, #irony) (Ptáček et al., 2014; Khodak et al., 2018). As pointed out by (Oprea and Magdy, 2020), noisy labels can be induced by this labelling method. Manual labelling is collecting texts and presenting them to human annotators for labelling (Filatova, 2012; Riloff et al., 2013; Abercrombie and Hovy, 2016). This can lead to a problem when annotator perception differs from author intention, as further outlined by (Oprea and Magdy, 2020).

Compared to English, only a few studies have been made on Arabic sarcasm detection. Among the studies completed in this area are research by (Riloff et al., 2013; Oprea and Magdy, 2019; Joshi et al., 2016; Bamman and Smith 2015; Campbell and Katz, 2012; Amir et al., 2016; Hazarika et al., 2018). (Oprea and Magdy, 2020) show the effect of sociocultural variables on sarcasm communication online, which makes the performance of models trained on English unpredictable, if they are trained on other languages. (Benamara et al., 2017; Abbes et al., 2020; Abu Farha and Magdy, 2020) relay on the two labelling methods mentioned above. Recently, efforts have been made by (Abu Farha and Magdy, 2020; Abu Farha et al., 2021 and Abbes et al., 2020) to create standard datasets to support sarcasm detection. In the SemEval-2022 Workshop, a shared task ('iSarcasmEval: Intended Sarcasm Detection In English and Arabic') (Abu Farha et al., 2022) was organised to contribute to the development of this area using a new labelling method that avoids the limitations of previous labelling methods.

3 Proposed System

SemEval 2022 task 6 focuses on detecting sarcastic tweets. The task supports Arabic and English languages and we tackle the problem on Arabic language. This task has three sub-tasks.

1. Sub-task A: Given a text, determine whether it is sarcastic or non-sarcastic.
2. Sub-task B (English only): A binary multi-label classification task. Given a text, determine which ironic speech category it belongs to, if any.
3. Sub-task C: Given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which is the sarcastic one.

We convert the input tweet to a fixed length sequence of words by padding shorter tweets and truncating longer ones. Then each word is replaced by its representation vector obtained from the pre-trained word embeddings model.

A MARBERT model is fine-tuned for sub-task A and then used for sub-tasks A and C. For sub-tasks C, the input tweets are passed to the model simultaneously and we consider the class of the tweet with the higher predicted score. We perform hyper parameters tuning to find the best parameters configuration.

Dialect	Non-Sarcastic	Sarcastic	Total
MSA	1470	49	1519
Egypt/Nile	727	567	1294
Gulf	67	17	84
Levant	81	35	116
Magreb	12	77	89
Total	2357	745	3102

Table 1: Arabic dataset statistics for sarcasm detection over the dialects.

4 Data Description

As mentioned above, annotator perception may differ from author intention. To overcome this problem, the authors annotated the data themselves, which is a new method to collect data introduced by the task's organisers.

Arabic and English datasets are collected using this method. For each sarcastic text, they provide a non-sarcastic rephrase to convey the same intended message, for English and Arabic datasets. Eventually, for English dataset, linguistic experts label each tweet to one of the ironic speech categories outlined by (Leggitt and Gibbs, 2000): sarcasm, irony, satire, understatement, overstatement, and rhetorical question. The dialect label of the text is included for the Arabic dataset.

Table 1 shows statistics of the Arabic training set, where we can find that 24% of the data is sarcastic (745 tweets). Most of the data is either in Modern Standard Arabic (MSA) or the Egyptian/Nile dialects, while there are few examples of the Magreb and Gulf dialects.

5 Implementation

We trained the proposed solution model using the given Arabic dataset. We divided its training data into 80% for training, 10% for validation and 10% for testing. In this section, we discuss the details of the different deep learning models we built or fine-tuned[1].

For our solution, we fix the tweets length to 64 by truncating longer tweets and padding shorter ones. This length is selected as the max value of the Arabic training tweets lengths after tokenization. After that each token in the input tweet is replaced with its vector representation obtained from a pretrained word embeddings model. We used pretrained MARBERT to initialize the words embeddings but these representations are then updated during the training of the deep learning models. The huggingface[2] pytorch implementation includes a set of interfaces designed for a variety of NLP tasks. Though these interfaces are all built on top of a trained BERT models, each has different top layers and output types designed to accomodate their specific NLP task. We used BertForSequenceClassification which is the normal MARBERT model with an added single linear layer on top for classification that we used as a sentence classifier.

5.1 MARBERT Model

Language models (LMs) exploiting self-supervised learning such as BERT (Devlin et al., 2019) which became a popular NLP approach to transfer learning. Transfer learning is used to reduce the time of the training and provide a better performance. This uses a pre-trained model as a starting point for training. Monolingual LMs pre-trained with larger vocabulary and bigger language-specific datasets usually perform better than multilingual models such as mBERT (Devlin et al., 2019; Virtanen et al., 2019).

Arabic has a large number of diverse dialects. Multilingual and Monolingual models such as mBERT and AraBERT (Antoun et al., 2020), respectively, are trained on mostly MSA datasets. The Arabic dataset used in this task has multiple dialects. This motivated us to use MARBERT which is trained on a large Twitter dataset (1B Arabic tweets), which involves both MSA and diverse dialects. The authors used the same network architecture as $BERT_{Base}$ (Devlin et al., 2019) to build the model, without the NSP objective which was found not crucial for model performance (Liu et al., 2019).

[1]The source code for the developed models can be found through: https://github.com/AyaLotfy/iSarcasmEval.

[2]https://huggingface.co/UBC-NLP/MARBERT.

Metric	Non-Sarcastic	Sarcastic
Precision	$\frac{TN}{TN+FP}$	$\frac{TP}{TP+FP}$
Recall	$\frac{TN}{TN+FN}$	$\frac{TP}{TP+FN}$

Table 2: Precision and recall with respect to the sarcastic and non-sarcastic classes.

6 Experiment Results

In this section we report and discuss the results of the proposed solution when evaluated on our testing data. Moreover, we show the results of our submission to SemEval 2022 task 6 on Arabic data.

Our model ranked the second out of 32 participants for sub-task A and the 9^{th} out of 13 participants for sub-task C, SemEval 2022 task 6 (iSarcasmEval: Intended Sarcasm Detection In English and Arabic).

6.1 Results and Evaluation

We divided the training data into 80% for training, 10% for validation and 10% for testing. The official evaluation metric for sub-task A was the F-score of the sarcastic class (F1-sarcastic), the macro average of the F-score for sub-task B and the accuracy for sub-task C. F1-sarcastic is calculated using the following equation:

$$F1^{sarcastic} = 2 \times \frac{P^{sarcastic} \times R^{sarcastic}}{P^{sarcastic} + R^{sarcastic}}$$

Where $P^{sarcastic}$, $R^{sarcastic}$ are the precision and recall with respect to the sarcastic class. Table 2 presents the equations to calculate the precision and recall with respect to the sarcastic and non-sarcastic classes.

Table 3 presents the models' performance for sub-task A, the best result (0.9) was obtained by BertForSequenceClassification, which is the normal MARBERT model with an added single linear layer on top for classification that we used as a sentence classifier. The proposed system has also been submitted for the sub-task C, and was ranked the 9^{th} out of 13 participants. For sub-task C, the input tweets are passed to the model simultaneously and we consider the class of the tweet with the higher predicted score. We believe this result should be studied as a future work by investigating different

Model	**F1-sarcastic**
MARBERT	**0.90**

Table 3: Performance of the model using our testing set for sub-task A on Arabic dataset.

Task	**Main Metric**	**Result**	**Rank**
A	F1-sarcastic	0.5076	2^{nd}
C	Accuracy	0.7450	9^{th}

Table 4: Main metric results obtained by the proposed model on the official test set for both sub-tasks A and C on Arabic dataset.

setup settings and, more importantly, to analyse errors on the test dataset. The model was fine-tuned for 4 epochs using the initial learning rate $2e^{-06}$, a batch size of 32, Adam weight decay optimizer and cross-entropy loss function. As well, we built other models but MARBERT outperforms them. The submitted model achieved an F1-sarcastic of 0.5076 on the official testing set for sub-task A.

6.2 Submission Results

For both aforementioned sub-tasks, we (AlexU-AL team) submitted the predicted classes based on the MARBERT model. For sub-task A, the proposed model achieved the second rank compared with the other systems proposed by other 31 participants. The submitted model achieved an F1-sarcastic of 0.5076 on the official testing set for sub-task A. For sub-task C, the model achieved an F-Score of 0.7442 and an accuracy of 0.7450 on the official testing set. Table 4 presents the official results achieved by our proposed model on the official testing set for sub-tasks A and C.

7 Conclusion

We used the fine-tuned MARBERT model in our submissions to SemEval 2022 task 6. We participated in the A and C sub-tasks for sarcasm detection in Arabic tweets. Our proposed approach is ranked the 2^{nd} and the 9^{th} in sub-tasks A and C, respectively. For future work, we explore the impact of building deeper neural networks with multiple convolutions or recurrent layers applied sequentially on the input text.

References

Ines Abbes, Wajdi Zaghouani, Omaima El-Hardlo, and Faten Ashour. 2020. DAICT: A dialectal Arabic irony corpus extracted from Twitter. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6265–6271, Marseille, France. European Language Resources Association.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Gavin Abercrombie and Dirk Hovy. 2016. Putting sarcasm detection into context: The effects of class imbalance and manual labelling on supervised machine classification of Twitter conversations. In *Proceedings of the ACL 2016 Student Research Workshop*, pages 107–113, Berlin, Germany. Association for Computational Linguistics.

Ibrahim Abu Farha and Walid Magdy. 2020. From Arabic sentiment analysis to sarcasm detection: The ArSarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 32–39, Marseille, France. European Language Resource Association.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ibrahim Abu Farha, Wajdi Zaghouani, and Walid Magdy. 2021. Overview of the WANLP 2021 shared task on sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 296–305, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

David Bamman and Noah Smith. 2015. Contextualized sarcasm detection on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 9, pages 574–577.

Farah Benamara, Cyril Grouin, Jihen Karoui, Véronique Moriceau, and Isabelle Robba. 2017. Analyse d'opinion et langage figuratif dans des tweets: présentation et résultats du défi fouille de textes deft2017.

John D Campbell and Albert N Katz. 2012. Are there necessary conditions for inducing a sense of sarcastic irony? *Discourse Processes*, 49(6):459–480.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 392–398, Istanbul, Turkey. European Language Resources Association (ELRA).

Ahmed Hassan Yousef, Walaa Medhat, and Hoda Mohamed. 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. *arXiv preprint arXiv:1805.06413*.

Aditya Joshi, Pushpak Bhattacharyya, Mark Carman, Jaya Saraswati, and Rajita Shukla. 2016. How do cultural differences impact the quality of sarcasm annotation?: A case study of indian annotators and american text. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 95–99.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. A la carte embedding: Cheap but effective induction of semantic feature vectors. *arXiv preprint arXiv:1805.05388*.

John Leggitt and Raymond Gibbs. 2000. Emotional reactions to verbal irony. *Discourse Processes - DISCOURSE PROCESS*, 29:1–24.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

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

Diana G Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Lrec 2014 proceedings*. ELRA.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Silviu Oprea and Walid Magdy. 2019. Exploring author context for detecting intended vs perceived sarcasm. *arXiv preprint arXiv:1910.11932*.

Silviu Vlad Oprea and Walid Magdy. 2020. The effect of sociocultural variables on sarcasm communication online. *Proc. ACM Hum.-Comput. Interact.*, 4(CSCW1).

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

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers*, pages 213–223.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 704–714, Seattle, Washington, USA. Association for Computational Linguistics.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. Semeval-2014 task 9: Sentiment analysis in twitter. pages 73–80.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: Bert for finnish. *arXiv preprint arXiv:1912.07076*.

reamtchka at SemEval-2022 Task 6: Investigating the effect of different loss functions for Sarcasm detection for unbalanced datasets

Reem Abdel-Salam
Computer Engineering Department
Cairo University
reem.abdelsalam13@gmail.com

Abstract

This paper describes the system used in SemEval-2022 Task 6: Intended Sarcasm Detection in English and Arabic. Achieving 20th, 3rd places with 34& 47 F1-Sarcastic score for task A, 16th place for task B with 0.0560 F1-macro score, and 10, 6th places for task C with 72% and 80% accuracy on the leaderboard. A voting classifier between either multiple different BERT-based models or machine learning models is proposed, as our final model. Multiple key points have been extensively examined to overcome the problem of the unbalance of the dataset as: type of models, suitable architecture, augmentation, loss function, etc. In addition to that, we present an analysis of our results in this work, highlighting its strengths and shortcomings.

1 Introduction

Sarcasm detection in any language text is hard for many reasons. First Sarcasm is much more than just written words. It is the tone, emphasis, experience, personal knowledge, and even facial expressions and body language that convey the meaning. When written into text such information is lost completely. Another reason is cultural references which make it hard for non-natives to get. In addition, all that is needed to show sarcasm is to be clear and articulate and not be as sketchy and without formatting, punctuation, proper use of language as texting might be for some people in their usual use of such media. Sarcasm is widespread on the social web and, by definition, may be extremely disruptive to machine learning/ deep learning models that use this data to perform tasks such as sentiment analysis, opinion mining, author profiling, and harassment identification. Thus, sarcasm detection might be the first crucial step in these systems. Several methods have been proposed to detect sarcasm in text as the recent shared-task for sarcasm detection in Arabic language (Abu Farha et al., 2021), where several teams used pretrained language models such as AraBERT and MarBert (Wadhawan, 2021; Song et al., 2021; Naski et al., 2021; Faraj et al., 2021; Abdel-Salam, 2021; Abuzayed and Al-Khalifa, 2021). In some of the work proposed in English language sarcasm detection involved rule-based and statistical approaches using: (a) Unigrams and pragmatic features (for example emoticons, etc.) (b) Sentiment and polarity estimation (c) Extraction of common patterns (Nagwanshi and Madhavan, 2014; Kumar et al., 2020; Bouazizi and Ohtsuki, 2016). A Recent shared-task for English language (Ghosh et al., 2020), where several methods has been proposed were based on pretrained model (i.e. BERT and RoBERTa) (Baruah et al., 2020; A. and D., 2020; Amir et al., 2016; Shangipour ataei et al., 2020).

In SemEval-2022 Task 6 (Abu Farha et al., 2022), the goal is to determine if sarcasm is presented in text or not, based on the text that is manually labeled by the task organizers. The shared task consists of three subtasks:

- Subtask A: this subtask is a binary classification task, where the goal is to determine whether is tweet is sarcastic or not, for the English language and Arabic languages. The official metric for this subtask is F1-Sarcastic.
- Subtask B: this subtask is a multi-label classification task where the goal is to determine which ironic speech category the tweet belongs to (English only). There are 6 ironic speech categories which are: irony, sarcasm, satire, understatement, overstatement, and rhetorical question. The official metric for this subtask F1- Macro score
- Subtask C: the goal of this task is to determine which text is more sarcastic given a pair of texts. The official metric for this subtask is accuracy.

This paper describes the system developed by the rematchka team for SemEval-2022 Task 6. Given that a key challenge in this task is the limited size of annotated data and the unbalanced distribution, we follow best practices from recent work on enhancing model generalization and robustness and propose a voting classifier model that leverages pre-trained representations(i.e. BERT and RoBERTa).

The main contributions of our work are as follows:

1. Identifying appropriate loss functions to help train Bert-Base models and Deep learning models in presence of extremely unbalanced datasets.
2. Investigating the importance of different layers in Bert-Base models. In addition, we present an analysis of our results in this work, highlighting its strengths and shortcomings.

Our code is made public and can be found here[1]. The rest of the papers goes as follow: section 2 discusses the proposed methods, section 4 shows experimental results, and section 5 concludes the paper.

2 System Overview

In this section, we discuss our prepossessing techniques, different hand-crafted features. We further explore different training techniques using Bert (Devlin et al., 2018), RoBERTa (Liu et al., 2019), BertTweet (Nguyen et al., 2020), Deberta (He et al., 2020), XLNet (Yang et al., 2019), and CANINE (Clark et al., 2022) models for English Language, and AraBert (Antoun et al., 2020), MarBert (Abdul-Mageed et al., 2021) and QARiB (Abdelali et al., 2021) models for Arabic Language. Moreover we investigate machine learning based models.

2.1 Dataset

In subtasks A, B, and C of the English language the dataset provided consisted of a total of 3468 manually annotated tweets. As shown in table 2, for subtask A English language, 867 of the total tweets are labeled sarcastic while 2601 are labeled not sarcastic. While in subtask A, C Arabic language the dataset provided consisted of 3102 manually annotated tweets. 745 of the total tweets are labeled sarcastic while 2357 are labeled not sarcastic. Table 1 shows distribution of labels for subtask B. For subtask B irony types: 713 out of the 867 sarcastic tweets are labeled sarcasm, 155 out of 867 are labeled irony, 25 out of 867 are labeled satire, 10 out of 867 are labeled understatement, 40 are labeled overstatement and 101 are labeled rhetorical-question.

2.2 Machine learning Based Approaches

The machine learning pipeline for subtask A and subtask B goes as follows: given a tweet a set of features are computed: lexical, syntactic, semantic, pragmatic, and polarity feature representations. These features are then fed to multiple models such as SVM, Logistic Regression (LR), random forest, boosting classifiers, and Xgboost for classification.

The Sarcastic tweets in the provided dataset are categorized as sarcasm, irony, satire, overstatement, rhetorical question, and understatement. It is crucial to extract features that cover those classes. For example, overstatement contains exaggerated terms, one way to extract it is to calculate the number of elongated punctuation or the number of characters in the word. Therefore a large set of hand-crafted features including lexical, syntactic, semantic, pragmatic, and polarity features are used.

Lexical features our lexical features contains word and character level n-grams. For word-level we use 1-gram, and for character level we use 4-gram, top 5000 n-grams are utilized based only on the term frequency-inverse document frequency (TF-IDF) values.

Syntactic features for syntactic features we use Spacy to calculate the number of adjectives, adverbs, nouns, pronouns, and verbs. In addition to that, we calculated the count of NER words in the tweet.

Sentiment & Polarity features sarcasm is used to express annoyance or outrage about a bad circumstance. As a result, people employ exaggerated and extremely positive terms to describe their negative condition (Yadollahi et al., 2017). Therefore it is important to extract them. For polarity & sentiment estimation, each tweet is divided into two parts. Then, for each part, its polarity and sentiment are calculated using NLTK Senti-WordNet (Baccianella et al., 2010). In addition to that, the overall polarity and sentiment for the whole tweet are calculated. Furthermore, the number of positive and negative sentiment words in a document, the number, and the count of the longest run of positives/negatives are computed as described in (Joshi

[1] https://github.com/rematchka/Intended-Sarcasm-Detection-In-English-and-Arabic-for-extremly-unbalanced-datasets

Task	Tweets Distribution					
	sarcastic	irony	satire	understatement	overstatement	rhetorical-question
B	713	155	25	10	40	101
Total	867					

Table 1: Dataset distribution for subtask B for English language

Language	Number of Tweets	Task	
		A	C
EN	Total Number	3468	1734
	Number of Sarcastic Tweets	867	867
	Number of Non-Sarcastic Tweets	2601	867
AR	Total Number	3102	1490
	Number of Sarcastic Tweets	745	745
	Number of Non-Sarcastic Tweets	3102	745

Table 2: Dataset distribution for subtask A and C for English and Arabic language

et al., 2015).

Pragmatic features it is hard to detect sarcasm in speech, as the word may have several meanings, and also the text contains behavioral aspects such as low tones, facial gestures, or exaggeration. These forms can be translated into elongation, repetition, and punctuation. To recognize such characteristics, we extract a set of features known as punctuation-related features. For each tweet, the following is computed:

- Presence of emoji's (González-Ibánez et al., 2011)
- Count of number of question marks
- Count of number of colons
- Count of number of a dollar sign
- Count of the number of quotes.
- Count of the number of exclamation marks.
- Count of number of special characters
- Count of number of hashtags
- Count of number of mentions
- Rate of capitalization
- Mixed cases count
- Rate of punctuation

Although punctuation is not relevant in itself and may not indicate whether the user is expressing sarcasm or any other emotion, when paired with other features, these attributes are anticipated to provide value to the classification.

Semantic& other features semantic features usually capture the conceptual relationship between words. For this we extracted multiple semantic features such as the presence of contradiction, interjections, the number of laughing expressions (Bouazizi and Ohtsuki, 2016) and the number of specific hashtags such as: "irony", "sarcasm", "hypocrisy" and "seriously". In addition to these features, other features are extracted as profanity count, topic modeling, and the presence of a numeric mismatch.

2.3 BERT-based Models

Figure 1 and 2 show overall architecture for BERT-based models used for most of the subtask A, subtask B and subtask C. The pipeline for training for most of the models in subtasks A and B as shown in figure 1 goes as follows, the input text is fed to BERT-base models, and the output of arbitrary 4 layers is taken and fed to KimCNN. The output is fed finally to the Fully connected layer (FC) layer. 4 losses can be used to assess the model performance F1-Cross-Entropy, Recall-Cross-Entropy, Balanced loss, and Asymmetric loss.

2.4 Subtask A& B English language

One of the biggest challenges in this task was the presence of an extremely unbalanced dataset. Using the provided dataset without any external dataset or modification of the widely used loss function as cross-entropy, hinge loss, etc leads to bad model performance, where the model focuses only on the majority class which was the Non-Sarcastic class. Even data augmentation couldn't boost performance that much. In one of our early experiments using RoBERTa model on the provided dataset, the model could achieve 77% accuracy and an F1-sarcastic score of zero. Therefore, increasing the length of the dataset seemed crucial at that point to allow better fine-tuning. NLPAug (Ma, 2019) was used for augmentation. Spelling augmentation and ContextualWordEmbsAug for insertion and substitution using Bert and RoBERTa were used. Furthermore, SynonymAug and ContextualWordEmbsForSentenceAug were used to make the dataset balanced, resulting in a balanced dataset where the number of non-sarcastic examples was 1560 and the number of sarcastic examples was 1040. Using this setting with RoBERTa model the model performance improved having F1-sarcastic 27 and accuracy 73% on the dev set. However augmentation didn't work all the time, some augmentation may lead to performance degrading or no improvements as RandomWordAug and WordEmbsAug using google news. Another interesting key point is that loss function does matter. Early experiments using cross-entropy with any BERT-based models achieved high accuracy however, achieved a 0 F1-sarcastic score on the dev set. When using loss functions that incorporate class imbalance the model performance improved. In one of the experiments conducted using Bert-Base uncased with sigmoid focal cross-entropy without any augmentation, the model performance improved reaching a 32 F1-Sarcastic score on the dev set. Therefore, we believe that there are 5 key points needed to be adjusted in order to improve model performance and recognition for the Sarcastic class:

- BERT-based models: which BERT-based model to use and achieve high performance.
- Model architecture: it is crucial to adjust the model architecture whether to use only the last layers and feed it to the Fully connected layer (FC), or whether to use the last n-layers from the model and apply average pooling which is then fed to FC layer, or feed output to Convolution or LSTM layer then FC layer. Also if important to choose which layer/s output will be used.
- Augmentation: it is important to decide whether to use augmentation to balance the dataset (however it is debatable whether it's okay to use them or not since the original dataset is manually labeled. Therefore augmentation might produce new examples we are not sure if it's truly sarcastic or not), or just down-sample the provided dataset by removing some examples.
- Loss function: loss function plays a key role in improving model performance. Therefore choosing the appropriate one will hugely impact the model performance to focus on both classes. A good choices may be sigmoid focal cross-entropy, recall-cross entropy loss (Tian et al., 2020), Dice Loss (Li et al., 2019), Asymmetric loss for multi-label classification and multi-class classification (Ben-Baruch et al., 2020), Distribution Balanced Loss (Wu et al., 2020) and F1-cross-entropy loss (which we believe is effective in our problem and propose).
- Data-Sampler: whether to use data-sampler instead of using augmentation. However, in this case, there are no examples removed from the dataset. The sampler (Yang et al., 2021) just makes sure that each batch fed to the model is balanced.

4 out of the 5 key points were heavily investigated: 1) BERT-based models, 2) Loss functions 3) Different architectures 4) Data-Sampler. For BERT-based models. It turns out that BertTweet, Bert-Base Uncased, RoBERTa, were the best performing models. Furthermore using multiple layers from the BERT-based model improves model performance drastically by 5-15%. Moreover, sigmoid focal cross-entropy, F1-cross-entropy, and also data-sampler improve model performance by 4-10%. F1-cross-entropy loss is denoted by this equation 1

$$\sum_{c=1}^{c=C} -(1 - F1_c) * N_c * log(P_c) \tag{1}$$

, where c is the class number, $F1_c$ is the F1-Score corresponding to specific class, P_c output of sigmoid/softmax for specific class c. Similarly, the recall-cross-entropy loss is denoted by this equation 2.

$$\sum_{c=1}^{c=C} -(1 - Recall_c) * N_c * log(P_c) \tag{2}$$

2.5 Subtask A Arabic Language

For the Arabic language, multiple Bert-Base models were fine-tuned on the provided dataset as AraBert, MarBert, and QARiB. Although the Arabic dataset was also imbalanced the model fine-tuning was easier. Without any modification to the dataset or the loss function, these models could achieve high performance on the dev set, unlike the English language. This might indicate that it is easier to detect sarcasm or find the sarcastic pattern in the Arabic language than the English.

2.6 Subtask C English and Arabic Languages

Based on Analysis and Experiments conducted in subtasks A & B, the following were investigated:

A) several architectures were investigated including the usage of the output of the last layer from Bert-Base models only of the usage of multiple layers from Bert-Base models. In addition to that, all architecture was similar to the siamese network where, the input to the model is both sarcastic and non-sarcastic, and the loss function is calculated based on the model prediction for both texts.

2.7 Final Recipe

In this subsection, we discuss the final models that were used during the evaluation phase for all subtasks.

2.7.1 subtask A English

A voter classifier is used based on different approaches since the voter classifier is model-agnostic, and can be tested on a variety of models and tasks to ensure that our findings are generalizable. 7 different approaches were used for the final classification. The first Model was Berttweet, where we used the output of the last four layers and fed it into KimCNN (Chen, 2015) while using recall-cross-entropy loss and data-sampler. The second model was Berttweet, where the input to the model was the original dataset. The input is fed to the model then the output of the last four layers is extracted and fed to four LSTM layers. The output of the four LSTM layers is fed into the FC layer. F1-cross-entropy loss is used with a data sampler. The third model is Bert-Base Uncased, where the input to the model was the original dataset and all hand-crafted features described in the previous subsection. The input is fed to the model then the output of the last four layers is extracted and fed to four LSTM layers. The output of the four LSTM layers is fed into the FC layer, the hand-crafted features are fed into another FC layer then the output of both is concatenated and fed to the last FC layer with sigmoid activation. F1-cross-entropy loss is used with a data sampler. For the fourth model, the architecture is similar to the first model except that Deberta model is used, and for the fifth model RoBERTa model is used with the same architecture and data-sampler as the first model but F1-cross-entropy loss is used instead of recall-cross-entropy loss. For the 6th and 7th models, linear SVM and linear SVM bagging classifiers were used with all of the set of hand-crafted features.

2.7.2 Subtask A Arabic

For this subtask, MarBert was used only. The input was fed to the model then the output was fed into FC layers. Cross-entropy loss was used.

2.7.3 Subtask B

The final model was a voting classifier between 5 models. The first and the second models were based on BertTweet model with KimCNN. However, the first model was trained using Asymmetric loss and the other model was trained using Distribution Balanced Loss with data sampler. RoBERTa model was used as the third model where the output of the 6th, up t 9th layers, were fed to KimCNN, and the model was trained using Asymmetric loss. For the 4th and 5th models multi-output linear SVM and multi-output linear SVM bagging classifiers were used with all of the set of the hand-crafted features.

2.7.4 subtask C English and Arabic

In the English subtask, the final model was voting classifiers between three models. The first model was based on XLM-RoBERTa (Conneau et al., 2019), where the output of the 6th, up t 9th layers, were fed to KimCNN and the model was trained using Margin ranking loss. For the second and third models, RoBERTa model was used with Margin ranking loss. However, for the second model architecture, the output of the last layer of RoBERTa was fed into FC layer, while for the third model the output of the 6th, up to 9th layer was fed to KimCNN. Similarly for the Arabic language subtask, a voting classifier between three models where used. The first and the second models were based on Arabert and MarBert where the output of the last layer was fed into FC layer, and the models were trained using Margin ranking loss. For the third layer, MarBert with KimCNN was utilized with Margin ranking loss for training. The whole training pipeline is demonstrated in figure 2. Since, the target of the subtask is to determine, which tweet is more sarcastic. During inference, both tweets are fed into the final model and whoever has a larger value is determined as a sarcastic tweet.

3 Experimental setup

Experiments were conducted via Python and PyTorch framework, running on Google Colab resources, which are Nvidia Tesla P100-PCIE-16GB GPU, Intel ® Xeon ® CPU @ 2.20 GHz, and 12GB RAM. We used 70%-10%-20% strategy for train-validation-test splits respectively for the training phase and 5-fold cross-validation during training. All of the presented models (submitted to the leaderboard) were trained on the provided dataset for the shared task only with no augmentation and the same splitting criteria. If augmentation is used in the dev-phase/test-phase, the following types of augmentation were used: 1) Spelling augmentation and contextual Word embedding augmentation for insertion and substitution using BERT and RoBERTa. 2) Synonym augmentation and contextual word embedding for sentence augmentation. Precision, recall, f-score (f1-sarcastic for subtask A, and macro average f1-score for subtask B), and accuracy (for subtask C) were used as evaluation metrics. All BERT-based models were trained using an AdamW optimizer with an initial learning rate of 0.001, weight decay rate of 0.0000001, and cosine annealing learning rate scheduler with minimum learning rate values of 0.0000001 and maximum temperature of 500. All the models were fine-tuned for 3-5 epochs. Pre-processing wasn't conducted on the dataset. During inferences, an ensemble of the 5 models is used (due to 5-fold validation)and is referred to as 1 model during our discussion.

4 Results

In this section, we discuss our main results during the development and evaluation phases. In addition to that failed cases analysis is discussed.

4.1 Results on Trial and Simulated Data

The evaluation was based on the train-test split criteria on the provided dataset. Table 3 shows different models' performance on the dev-set, for subtask A in the English language based on F1-sarcastic and accuracy. Early experiments were based on augmentation and Dice/cross-entropy loss. The performance of the models started to increase when using suitable architecture, multiple outputs from BERT-based models, and a suitable loss function that is sensitive for low classes. The difference between RoBERTa with dice loss and augmentation and RoBERTa with KimCNN and F1-cross-entropy is around 20 in F1-score. Surprisingly machine learning models could achieve a good F1-Sarcastic score. Table 7 shows different models performance on the dev-set, for subtask B based on F1-macro. The model performance is not high due to the small size of the dataset. It can be seen that Distribution Balanced Loss and Asymmetric loss is a good choice for unbalanced multi-label classification. For subtask, A Arabic language MarBert could achieve 89 F1-sarcastic while AraBert and QARiB achieve 54 and 79 F1-sarcastic on dev-set. For subtask C English language XLM-RoBERTa KimCNN could achieve 89% accuracy, while RoBERTa KimCNn and RoBERTa could achieve 90% and 85% on dev-set. For subtask C Arabic language, AraBert could achieve 82%, while MarBert and MarBert KimCNN could achieve 83% and 76% on dev-set.

4.2 Test Results

Tables 5, 4, 6, and 8 show the results of the proposed models and the official model (Voting classifier) that was used in the leaderboard. The voting classifier model is our official submitted model. Table 4 shows the results of our submitted model and the performance of each individual model that is used in the voting classifier. Table 8 shows the results of our submitted model (the voting classifier) in both languages. In addition, the performance of each model contributed to the voting classifier. For subtask C, In the English language, the voting classifier is composed of XLM-RoBERTa and RoBERTa with KimCNN and RoBERTa. For the Arabic language, the voting classifier is composed of MarBert, MarBert, and MarBert with KimCNN. Table 5 shows the results of our submitted model (the voting classifier). In addition to other models that were described in the dev-phase. The final voting classifier for subtask A English is composed of Bagging SVM with feature engineering, SVM with feature engineering, RoBERTa with KimCNN and F1-Cross-Entropy, Deberta with KimCNN and Recall-Cross-Entropy, BertTweet with KimCNN and Recall-Cross-Entropy, BertTweet with LSTM and F1-Cross-Entropy, Bert-Base-Uncased with LSTM and feature engineering and Recall-Cross-Entropy.

Our voting based classifier achieved 20th, while MarBert achieved 3rd place for subtask A, voting based classifier achieved 16th place for subtask B and 10, 6th places for subtask C. For subtask C the performance of the voting classifier and the other models are similar to the dev-set. For subtask B the voting classifier model performance was bad compared to a single model. All the models failed to detect over-statement in the test set. For subtask A English languages the voting classifier model performance was bad compared to a single model, this happened as the performance of machine learning models was bad on the test set. For subtask A, Arabic language the model performance falls below the performance observed on the dev-set. Based on tables 3 and 5 it can be seen that suitable architecture and loss aware function are the key to better performance. In addition to that, it seems that Augmentation has promising performance, and might boost results significantly if integrated with suitable architecture and loss function.

4.3 Error Analysis

In subtask A, the test-set contains 1400 examples, 200 of which are sarcastic, and 1200 of which are non-sarcastic. For subtask1 English language, the submitted model (Voting classifier) miss-classified 224 tweets as sarcastic and 113 as not-sarcastic. In the misclassifications, there were numerous prominent trends. Among the false negatives and false positives in the samples (Most of the time), the sarcasm and non-sarcasm communicated in many cases could only be extrapolated via world knowledge (for example:"Max Verstappen is such a clean driver, he never makes dirty moves when racing." and "This does not surprise me! Kat is a PR queen "). Some of the false positives (detected as sarcasm which is not) depend on the personality and the situation, which is in reality hard to detect is it a touch of sarcasm or not (for example: "Brrrr it's cold outside...I love it!" and "What people?"). For the Arabic language subtask A, Marbert miss-classified 324 tweet as sarcastic and 36 as not-sarcastic. Similarly most the wrongly tweets miss-classified as no-sarcastic was due to world knowledge (for example: trasnalted as "fifi abdo the ideal mother"). الام المثالية فيفي عبده

Some of the miss-classified tweets as sarcastic was related to personal life and personality (for example: trasnslated as "The class who shouldn't be called/named"). الدفعة التي لا يجب ذكر اسمها
In subtask B, the test-set contains 1400 examples, 180 are labeled sarcastic, 20 labeled irony, 49 labeled satire, 1 labeled under-statement, 10 labelled over-statement, and 11 labelled rhetorical-question. One the biggest problems that the dataset contained alot of non-sarcastic tweets, which affected the model prediction performance. A modification that should have done was to enter the tweet to subtask A models and then for the predicted sarcastic tweet subtask B model should be used to determine the type. In subtask C, the test-set contains 200 examples, 107 of them text 0 is more sarcastic than text 1. The voting model miss-classified 52 as text 0 being sarcastic and 4 examples as text 1 being more sarcastic. Some of the errors were a result of unclear sarcasm as an example these two text"Brr! It's really cold outside today" and "I'm loving how warm it is outside today". The more sarcastic text is the second one, however, it's hard to determine whether the weather is hot or cold, to come to this conclusion. Another common type of error is world knowledge as an example: "Benioff and Weiss will definitely go down in history for how terrible the script they wrote for GOT S8 was." and "Benioff and Weiss will definitely go down in history for their amazing work on the script of GOT S8.". If the model doesn't have a knowledge about the TV series and rating it's hard to tell that the second on is the sarcastic. In conclusion common error can be classified to:

- World knowledge
- Personality
- Personal experiences.

Model	F1-Saracstic
MarBert	0.4767

Table 6: Results for subtask A on the official test-set for Arabic language.

5 Conclusion

In this paper, we have discussed our system submitted to the SemEval-2022 Task 6. our model ranked 20th out of 43 participating teams in subtask A English language, 3rd out of 32 participating teams in subtask A Arabic language, 16th out of 22 participating teams in subtask B, 10th out of 16 teams in subtask C English language, and 6th out of 12 participating teams in subtask C Arabic language. We proposed a voting classifier that leverages fine-tuned, per-trained models. We proposed the usage of the different loss functions in each task to accommodate dataset imbalance and improve model training. Overall we showed the power of the loss function to improve model performance without the need to access any external dataset or to use any kind of augmentation. We also investigated the main common error, which disturbs model performance. In future efforts, we plan to further improve our model to better handle data-imbalance constraints and world knowledge needed to improve model performance.

Model	Loss Function	Augmentation	Accuracy	F1-Sarcastic
Bert-Base-Uncased with KimCNN	Recall-Cross-Entropy	No	74%	38
Bert-Base-Uncased with LSTM and Feature Engineering	Recall-CrossEntropy	No	75%	41
Bert-Base-multilingual-uncased	Dice Loss	Yes	73%	32
Bert-Base-Uncased	Dice Loss	Yes	76%	35
Bert-Base-Uncased	Cross-Entropy	Yes	73%	38
RoBERTa	CrossEntropy	Yes	73%	27
RoBERTa with KimCNN	F1-Cross-Entropy	No	75%	43
BertTweet with KimCNN	Recall-Cross-Entropy	No	79%	57
BertTweet with LSTM	F1-Cross-Entropy	No	78%	54
CANNIE	F1-Cross-Entropy	No	77%	33
Deberta with KimCNN	Recall-Cross-Entropy	No	77%	43
SVM + Feature Engineering	_	Yes	58%	38
SVM + Feature Engineering	_	No	63%	40
Logistic regression + Feature Engineering	_	Yes	57%	38
Logistic regression + Feature Engineering	_	No	55%	37
Bagging SVM + Feature Engineering	_	Yes	54%	34
Bagging SVM + Feature Engineering	_	No	66%	40

Table 3: Results for subtask A on the dev-set for various models and techniques for English language.

Model	Loss Function	F1-Macro	F1-Sarcasm	F1-irony	F1-satire	F1-understatement	F1-overstatement	F1-rhetorical question
RoBERTa + KimCNN	Distribution Balanced Loss	0.09	0.23	0.04	0.18	0.00	0.02	0.09
RoBERTa + KimCNN	Asymmetric loss	0.09	0.23	0.04	0.17	0.00	0.02	0.09
BertTweet + KimCNN	Asymmetric loss	0.10	0.23	0.05	0.16	0.00	0.03	0.11
BertTweet + KimCNN	Recall-Cross-Entropy	0.05	0.23	0.03	0.00	0.00	0.02	0.00
BertTweet + KimCNN	Distribution Balanced Loss	0.12	0.23	0.11	0.17	0.00	0.06	0.11
SVM+ FE	_	0.07	0.20	0.03	0.09	0.01	0.02	0.09
Bagging SVM + FE	_	0.08	0.24	0.03	0.14	0.00	0.02	0.09
Voting	_	0.0560	0.2251	0.0285	0.0664	0.0000	0.0161	0.0000

Table 4: Results for subtask B on the official test-set for various models and techniques for English language. The voting Classifier model is our official submitted model which is composed of the above models.

Model	Loss Function	Augmentation	Accuracy	F1-Sarcastic
Bert-Base-Uncased with KimCNN	Recall-Cross-Entropy	No	74%	28
Bert-Base-Uncased with LSTM and Feature Engineering	Recall-Cross-Entropy	No	76%	37
Bert-Base-Uncased	Dice Loss	Yes	79%	32
Bert-Base-Uncased	Cross-Entropy	Yes	74%	31
RoBERTa with KimCNN	F1-Cross-Entropy	No	73%	31
BertTweet with KimCNN	Recall-Cross-Entropy	No	76%	40
BertTweet with LSTM	F1-Cross-Entropy	No	75%	42
Deberta with KimCNN	Recall-Cross-Entropy	No	78%	32
SVM + Feature Engineering	_	No	55%	21
Bagging SVM + Feature Engineering	_	No	60%	18
Voting Classifier	_	_	_	34.05

Table 5: Results for subtask A on the official test-set for various models and techniques on English language. The voting Classifier model is our official submitted model.

Model	Loss Function	F1-Macro
RoBERTa + KimCNN	Distribution Balanced Loss	38
RoBERTa + KimCNN	Asymmetric loss	43
BertTweet + KimCNN	Asymmetric loss	44
BertTweet + KimCNN	Recall-CrossEntropy	21
BertTweet + KimCNN	Distribution Balanced Loss	44
SVM+ FE	_	32
Bagging SVM + FE	_	37

Table 7: Results for subtask B on the dev-set for various models and techniques for English language.

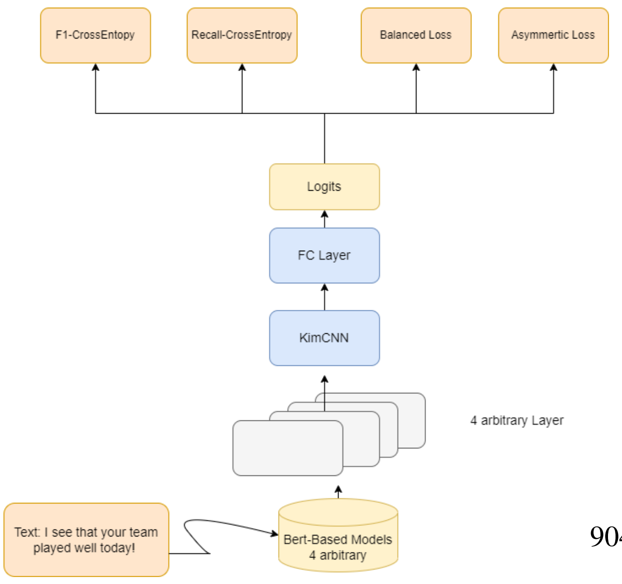


Figure 1: The overall architecture of our proposed BERT-based system used for subtask A and B.

Model	Language	Accuracy
AraBert	Ar	73
MarBert	Ar	81
MarBert + KimCNN	Ar	70
Voting	Ar	80
XLM-RoBERTa+ KimCNN	En	73
RoBERTa+ KimCNN	En	72
RoBERTa	En	69
Voting	En	72

Table 8: Results for subtask C on the official test-set for various models and techniques For English and Arabic language. The voting Classifier model is our official submitted model.

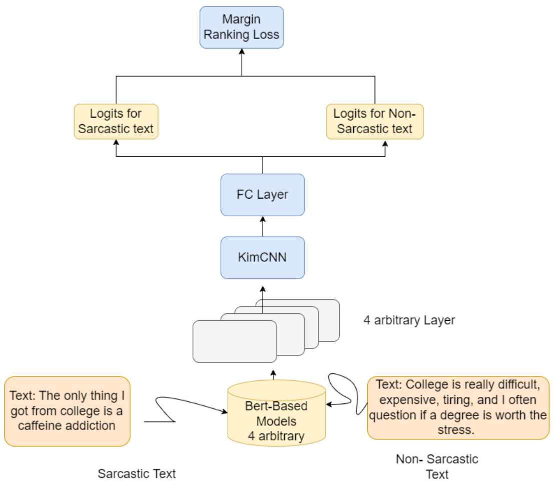

Margin Ranking Loss
Logits for Sarcastic text
Logits for Non-Sarcastic text
FC Layer
KimCNN
4 arbitrary Layer
Text: The only thing I got from college is a caffeine addiction
Bert-Based Models 4 arbitrary
Text: College is really difficult, expensive, tiring, and I often question if a degree is worth the stress.
Sarcastic Text
Non- Sarcastic Text

Figure 2: The overall architecture of our proposed BERT-based system used for subtask C.

References

Kalaivani A. and Thenmozhi D. 2020. Sarcasm identification and detection in conversion context using BERT. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 72–76, Online. Association for Computational Linguistics.

Reem Abdel-Salam. 2021. WANLP 2021 shared-task: Towards irony and sentiment detection in Arabic tweets using multi-headed-LSTM-CNN-GRU and MaRBERT. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 306–311, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Ahmed Abdelali, Sabit Hassan, Hamdy Mubarak, Kareem Darwish, and Younes Samih. 2021. Pre-training bert on arabic tweets: Practical considerations.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ibrahim Abu Farha, Wajdi Zaghouani, and Walid Magdy. 2021. Overview of the WANLP 2021 shared task on sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 296–305, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Abeer Abuzayed and Hend Al-Khalifa. 2021. Sarcasm and sentiment detection in arabic tweets using bert-based models and data augmentation. In *Proceedings of the sixth Arabic natural language processing workshop*, pages 312–317.

Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Arup Baruah, Kaushik Das, Ferdous Barbhuiya, and Kuntal Dey. 2020. Context-aware sarcasm detection using bert. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 83–87.

Emanuel Ben-Baruch, Tal Ridnik, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. 2020. Asymmetric loss for multi-label classification. *arXiv preprint arXiv:2009.14119*.

Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488.

Yahui Chen. 2015. Convolutional neural network for sentence classification. Master's thesis, University of Waterloo.

Jonathan H Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dalya Faraj, Dalya Faraj, and Malak Abdullah. 2021. SarcasmDet at sarcasm detection task 2021 in Arabic using AraBERT pretrained model. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 345–350, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. *arXiv preprint arXiv:2005.05814*.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.

Avinash Kumar, Vishnu Teja Narapareddy, Veerubhotla Aditya Srikanth, Aruna Malapati, and Lalita Bhanu Murthy Neti. 2020. Sarcasm detection using multi-head attention based bidirectional lstm. *Ieee Access*, 8:6388–6397.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2019. Dice loss for data-imbalanced nlp tasks. *arXiv preprint arXiv:1911.02855*.

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

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Prateek Nagwanshi and CE Veni Madhavan. 2014. Sarcasm detection using sentiment and semantic features. In *KDIR*, pages 418–424.

Malek Naski, Abir Messaoudi, Hatem Haddad, Moez BenHajhmida, Chayma Fourati, and Aymen Ben Elhaj Mabrouk. 2021. iCompass at shared task on sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 381–385, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *arXiv preprint arXiv:2005.10200*.

Taha Shangipour ataei, Soroush Javdan, and Behrouz Minaei-Bidgoli. 2020. Applying transformers and aspect-based sentiment analysis approaches on sarcasm detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 67–71, Online. Association for Computational Linguistics.

Bingyan Song, Chunguang Pan, Shengguang Wang, and Zhipeng Luo. 2021. DeepBlueAI at WANLP-EACL2021 task 2: A deep ensemble-based method for sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 390–394, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Junjiao Tian, Niluthpol Chowdhury Mithun, Zachary Seymour, Han-pang Chiu, and Zsolt Kira. 2020. Recall loss for imbalanced image classification and semantic segmentation.

Anshul Wadhawan. 2021. Arabert and farasa segmentation based approach for sarcasm and sentiment detection in arabic tweets. *arXiv preprint arXiv:2103.01679*.

Tong Wu, Qingqiu Huang, Ziwei Liu, Yu Wang, and Dahua Lin. 2020. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *European Conference on Computer Vision*, pages 162–178. Springer.

Ali Yadollahi, Ameneh Gholipour Shahraki, and Osmar R Zaiane. 2017. Current state of text sentiment analysis from opinion to emotion mining. *ACM Computing Surveys (CSUR)*, 50(2):1–33.

Ming Yang, Jihwan Bang, Jirka Borovec, Tobias, and Davi Innovation. 2021. Imbalanced dataset sampler using pytorch.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

niksss at SemEval-2022 Task 6: Are Traditionally Pre-Trained Contextual Embeddings Enough for Detecting Intended Sarcasm ?

Nikhil Singh
Manipal University Jaipur
nikhil3198@gmail.com

Abstract

This paper presents the 10th and 11th place system for Subtask A - English and Subtask A - Arabic respectively of the SemEval 2022 - Task 6. The purpose of the Subtask A was to classify a given text sequence into sarcastic and non-sarcastic. We also breifly cover our method for Subtask B which performed subpar when compared with most of the submissions on the official leaderboard . All of the developed solutions used a transformers based language model for encoding the text sequences with necessary changes of the pretrained weights and classifier according to the language and subtask at hand .

1 Introduction

According to (Yaghoobian et al., 2021), "Sarcasm detection is the task of identifying irony containing utterances in sentiment-bearing texts". Even though sarcastic humor is present throughout social media, it is hard for even humans to comprehend it certainly as the reader doesn't always perceive it the same way the speaker intended with intricate socio-psychological and cultural references. With the way, current models for text representation are trained like Word2Vec (Mikolov et al., 2013) and Glove (Pennington et al., 2014), which have the same representation for a particular word irrespective of the context they fail to incorporate the meaning of sarcastic texts. Even though the recent state-of-the-art language models provide contextual embeddings for words, it still fails to tell the sarcastic humor from normal as we explain later in the paper.

There have been a lot of attempts at computationally automating sarcasm detection in the literature. Discernibly, the task of sarcasm detection can be classified into content and context-based methods. With features like Structural, morphosyntactic and semantic ambiguity features (Reyes et al., 2012),User mentions (replies), emoticons, N-grams, dictionary- and, sentiment-lexicon-based features(González-Ibáñez et al., 2011)) and features based on word embedding similarity (Joshi et al., 2016) coming inside content based method. With the surge of seq2seq based model such as BERT (Bidirectional Encoder Representations from Transformers (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and XLNet (Yang et al., 2019), etc. have been heavily employed for sarcasm detection in the literature. (Potamias et al., 2020) proposed an R-CNN-RoBERTa which is a hybrid model leveraging RoBERTa's contextual embedding into a recurrent convolutional neural network. (Dadu and Pant, 2020) created an ensemble of RoBERTa and ALBERT (Lan et al., 2019). Seeing the success of pre-trained language model, we decided to leverage the pre-trained contextual embeddings and transformers toward sarcasm detection.

While some of the previous textual sarcasm detection datasets involved annotation via finding some predefined criteria, such as including specific tags (e.g. sarcasm, irony) (Ptácek et al., 2014) associated with the text. Other datasets involved manual labeling (Filatova, 2012) (Yang et al., 2016). However, the mentioned labeling techniques produced noisy or uncertain labels which would further compromise the effectiveness of the models trained on them. Further, most of the sarcasm detection work has been done in the English language and it is highly unlikely that the models trained on one language would generalize well on other languages.

The purpose of SemEval-2022 Task 6 - Intended Sarcasm Detection in English and Arabic (Abu Farha et al., 2022) is to advance the development of automatic textual sarcasm detection by providing a dataset which overcomes the problem of noisy and uncertain labels by having the authors provide the labels themselves. The shared task is divided into three subtasks:

- SubTask A(English Arabic): Given a text, determine whether it is sarcastic or non-sarcastic;

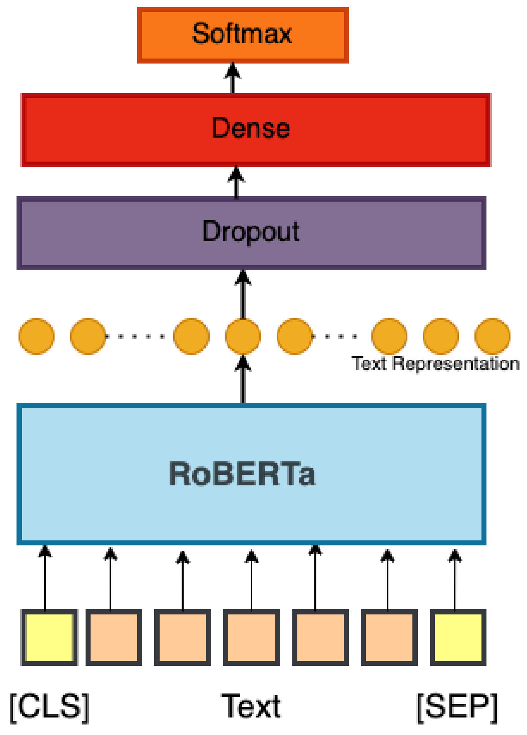


Figure 1: Model Architecture for Subtask A

- SubTask B (English only): A binary multi-label classification task. Given a text, determine which ironic speech category it belongs to, if any;

- SubTask C(English Arabic): Given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which is the sarcastic one.

2 System Overview

Here, we describe the data pre-processing steps along with the models for each subtask, and the experimental setup for the system. We provide an overview of the system in Figure 1.

2.1 Data

Manually examining the provided dataset, it was found that the datapoints were mostly from social media. Hence, basic text denoising steps such as user-handle removal, number removal, de-emojifying and repeating punctuation were handled for all three subtasks.

2.2 SubTask A

For this task the participating teams had to develop a system which, given a text, determines whether it is sarcastic or non-sarcastic. We develop a binary sequence classifier as shown in Figure 1. It comprises of a roberta base model as the text encoder with weights taken from cardiffnlp/twitter-roberta-base-sentiment(Barbieri et al., 2020) for English and AraBERT (Antoun et al., 2020) for Arabic. Both of these models are hosted on the HuggingFace library [1] .

The detailed steps involved in this experiment is present below.

- The pre-processed data comprising of cleaned text sequences is tokenized using a BertTokenizer from Huggingface and is passed through the model mentioned above to embed it into a 768 dimensional feature vector containing the syntactical information of the input string.

- The feature vector is then passed through a dropout layer to increase the regularization which in-turn increases the generalizability of the model.

- The model was trained in a supervised manner in a binary classification regime for 5 Epochs with a batch size of 32. Rest of the Hyper-parameters are shown in Table 1. A seed value of 42 to keep the model deterministic.

- The model took approximately 45 minutes to train on Nvidia's P100 GPU with a memory of 16Gb.

- The complete experiment was done on Google Colab Pro.

2.3 SubTask B

For this subtask the participants were required to determine which ironic speech category the given input text belongs to. We treated this problem as a multilabel classification with the same encoder as subtask A but with a multilabel classifier instead of a binary classifier to capture the dependency of one sarcasm type on other. We trained this model using a Label Ranking average precision loss for 3 Epochs, with a batch size of 4 and rest of the parameters same as Subtask A for English. Simple Transformers[3] was used to do the development. The model was trained on Google colab and it took

[1] https://huggingface.co/
[3] https://github.com/ThilinaRajapakse/simpletransformers

Parameter	Value
Max sequence Length	96
Batch Size	16
Learning rate	2e-5
Weight decay	Linear
Momentum	0.9
Optimizer	AdamW [2]
Epochs	5
Loss	Cross Entropy
Parameter	**Value**
Max sequence Length	64
Batch Size	16
Learning rate	2e-5
Weight decay	Linear
Momentum	0.9
Optimizer	AdamW
Epochs	5
Loss	Cross Entropy

Table 1: Experimental Setup for Subtask A

Parameter	Value
Max sequence Length	96
Batch Size	4
Learning rate	1e-5
Epochs	3
Loss	LRAP [4]

Table 2: Experimental Setup for Subtask B

around 35 minutes to finish the training. The Experiment setup has been shown in Table 2 in a concise manner.

3 Results

3.1 Subtask A

All the submitted systems for Subtask A were evaluated using the five metrics that are, Accuracy, Precision, Recall, F1 score, and F1 score of the sarcastic class. However, the ranking of the systems was determined using the F1 score of the sarcastic class. We were officially ranked 10th in Subtask A - English and 11th in Subtask A - Arabic. With an F1 score of 40.16% for English and 40% for Arabic. The rest of the metrics are shown in Table 3.

3.2 Subtask B

The submitted systems for Subtask B were evaluated using the F1 scores of the respective classes in the dataset. The ranking was determined by the Macro F1 score. We ranked at position 22 with a Macro F1 score of 0.0380. The rest of the scores are present in Table 3.

4 Error Analysis

After examining the predictions from the submitted model, we saw that the model struggled significantly in classifying the text sequences to sarcastic type. We inferred that, even though we put less weight on the non-sarcastic class during the loss computation, the model overfitted to the abundant class of non-sarcastic text sequences with a ratio of roughly around 3:1 between the two classes for both Engligh and Arabic tasks. We also noted that individual hyper-parameters had significant roles in the performance of the model. Training different models with different hyper-parameters and ensembling them together showed a significant increase in performance in the post evaluation period. For Subtask B, the main reason for the poor performance of our model and most submitted models on the leaderboard, is the inter-class difference between individual sarcasm types is very low.Which in-turn confuses the model and the output probability is roughly close to each other.

5 Conclusion

We developed a system to classify sarcastic text from non-sarcastic text using contextualized embeddings from a language model which didn't have any prior information about what the fundamental concepts of sarcasm. It inculcates language understanding through self-supervised training techniques namely, masked words prediction and next sentence prediction. However, sarcasm doesn't work in the same way as declarative, exclamatory, imperative, and interrogatory sentences. These were the major type of sentences used for pre-training the language models.

In future work, we plan to use a seq2seq based encoder-decoder model for instilling knowledge of sarcasm in the already available seq2seq models like T5 (Raffel et al., 2019). Wherein we'll use data similar to what was provided in SubTask C and train a seq2seq model with input as the sarcastic text and the model will learn to paraphrase that input sentence into a non-sarcastic sentence as output.

Language	F-1 score	Precision	Recall	Accuracy
English	0.6353	0.6215	0.6683	0.7850
Arabic	0.5800	0.6083	0.7167	0.6571

Table 3: Other Metrics for Subtask A

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

Tanvi Dadu and Kartikey Pant. 2020. Sarcasm detection using context separators in online discourse. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 51–55.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 392–398, Istanbul, Turkey. European Language Resources Association (ELRA).

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in Twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586, Portland, Oregon, USA. Association for Computational Linguistics.

Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1006–1011, Austin, Texas. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas-Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Tomás Ptácek, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *COLING*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data Knowledge Engineering*, 74:1–12. Applications of Natural Language to Information Systems.

Hamed Yaghoobian, Hamid R Arabnia, and Khaled Rasheed. 2021. Sarcasm detection: A comparative study. *arXiv preprint arXiv:2107.02276*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Dartmouth at SemEval-2022 Task 6: Detection of Sarcasm

Rishik Lad, Weicheng Ma, Soroush Vosoughi
Department of Computer Science, Dartmouth College
{rishik.lad.23,weicheng.ma.gr,soroush.vosoughi}@dartmouth.edu

Abstract

This paper introduces the result of Team Dartmouth's experiments on each of the five subtasks for the detection of sarcasm in English and Arabic tweets. This detection was framed as a classification problem, and our contributions are threefold: we developed an English binary classifier system with RoBERTa$_{\text{BASE}}$, an Arabic binary classifier with XLM-RoBERTa$_{\text{BASE}}$, and an English multilabel classifier with BERT$_{\text{BASE}}$. Preprocessing steps are taken with labeled input data prior to tokenization, such as extracting and appending verbs/adjectives or representative/significant keywords to the end of an input tweet to help the models better understand and generalize sarcasm detection. We also discuss the results of simple data augmentation techniques to improve the quality of the given training dataset as well as an alternative approach to the question of multilabel sequence classification. Ultimately, our systems place us in the top 14 participants for each of the five subtasks.

1 Introduction

Sarcasm is a form of irony that occurs when there is a discrepancy between the literal and intended meanings of a text or utterance. This discrepancy typically manifests itself in the form of dislike, contempt, or derogation. Furthermore, sarcasm can be divided into multiple types: general sarcasm, irony, satire, understatement, overstatement, and rhetorical question.

This task concerns itself with the detection of sarcasm in online tweets (Abu Farha et al., 2022). This is an important issue to solve because the nature of sarcasm can interfere with the effectiveness of natural language processing models, particularly when conducting sentiment analysis, opinion mining, or other emotion-based tasks. Machine learning models deployed for such business use cases can be negatively impacted when processing sarcastic texts and provide inaccurate results, thereby harming an organization's bottom line. Therefore, it is critical that machine learning models be developed that can understand how to detect, ingest, and truly understand sarcasm.

This paper discusses how to identify sarcastic tweets in binary and multi-label classification contexts for both English and Arabic, as directed by SemEval-2022 Task 6 (Abu Farha et al., 2022). There are five subtasks: general sarcasm detection in standalone English and Arabic tweets (Task A), identification of sarcastic category in an English tweet (Task B), and identification of the sarcastic tweet in an English/Arabic pair of tweets (Task C).

Our experiments show that pre-trained transformer models demonstrate a strong ability of solving most of the subtasks of this challenge. Specifically, we fine-tune a RoBERTa$_{\text{BASE}}$ model to detect the presence of sarcasm in tweets for the English components of Tasks A and C. For the Arabic version of Tasks A and C, we apply an XLM-RoBERTa$_{\text{BASE}}$ model. For Task B, which is framed as a multilabel sequence classification problem, after experimenting with several RoBERTa and BERT models, we report the performance of a BERT$_{\text{BASE}}$ model which is fine-tuned to detect the categories of sarcasm in English tweets, if any.

While these models perform reasonably well in the evaluations, the imbalanced distributions of labels and poor annotation quality for some instances introduce unexpected noise to the fine-tuning process of these models and harm their performance in the evaluations. Despite our effort to augment unrepresented classes in the training data, simple data augmentation approaches do not show clear positive effects on the models' performance. More advanced data augmentation methods could be tried to trigger notable performance improvements on the challenge test dataset.

2 Approaches

As mentioned above, there are three systems designed for the five subtasks. We discuss the model architecture, input processing, and other key elements for each of the systems below.

2.1 RoBERTa for Binary Classification

The English component for Tasks A and C require us to distinguish whether a tweet is sarcastic, either as a standalone text (Task A) or in comparison to another tweet (Task C). Both of these tasks were framed as binary classification problems and our solution leveraged the RoBERTa model (Figure 1) from Facebook AI to achieve this by producing rich feature representations from the inputs.

In particular, RoBERTa builds upon the Bidirectional Encoder Representations from Transformers (BERT) by modifying some key hyperparameters, training with much larger learning rates and batches, and removing BERT's next-sentence pretraining objective (Liu et al., 2019). Critically, this allows RoBERTa to improve on the masked language modeling objective and allows for better task performance down the road. The RoBERTa base is composed of 12-layers, 768-hidden, 12 self-attention heads, and 125M parameters.

Processing each input tweet first began with changing sarcasm labels from 1 (sarcastic) to 0.8 and 0 (non-sarcastic) to 0.2, although the 0.2 was eventually changed back to 0. This was to account for random noise in the dataset and for training examples that were of lesser quality than others.

The next step was tweet normalization: among other things, this included replacing hyperlinks, user tags, and emojis with a standardized token (`"@URL"` for hyperlinks and `"@USER"` for user tags, for example). Furthermore, contracted words were separated out to extract the key token. This was to ensure the model did not learn from random, irrelevant noise found in hyperlinks or user tags.

Before passing each normalized tweet into the tokenizer, however, we first extracted all verbs and adjectives from the original tweet (Sequence A) and strung them together with whitespace to create a new string (Sequence B). In turn, those verbs and adjectives were replaced with the `<mask>` token in the original tweet. This was executed in an attempt to help the model better learn the relationship between a tweet's sarcastic presence and any available verbs or adjectives in it. These two sequences were then joined together with separator tag `</s>` and fed into the tokenizer as a single sequence.

Padding tokens were added to make each input the same length for the RoBERTa model. The maximum length used for this system was 256. Although the longest string of tokens from the available training dataset was 111, we set it to 256 for the sake of safety. This ensured that all tensor inputs were set to equal the maximum sequence length used for batched parallelized training. This meant the ultimate input passed into the tokenizer looked like Table 1, where `<s>` is the classifier token, `</s>` is the separator token, `<pad>` is the padding token, `Seq-A` contains `<mask>` tokens where its adjectives and verbs originally were, and `Seq-B` is a string of all verbs and adjectives from the original tweet.

Input	`<s>Seq-A</s>Seq-B<pad><pad>`

Table 1: A sample input for encoding.

A sequence classification head containing a linear layer was applied on top of the final hidden-states output, with a label prediction of 1 denoting a sarcastic tweet and 0 denoting a non-sarcastic tweet. For standalone tweets (Task A), the threshold to pass a tweet as sarcastic was set to 0.40, where tweets with a score higher than that were marked as sarcastic while those underneath this threshold were not. For determining which of two tweets is sarcastic (Task C), the tweet with the highest absolute score, regardless of its relation to a threshold, was marked as sarcastic.

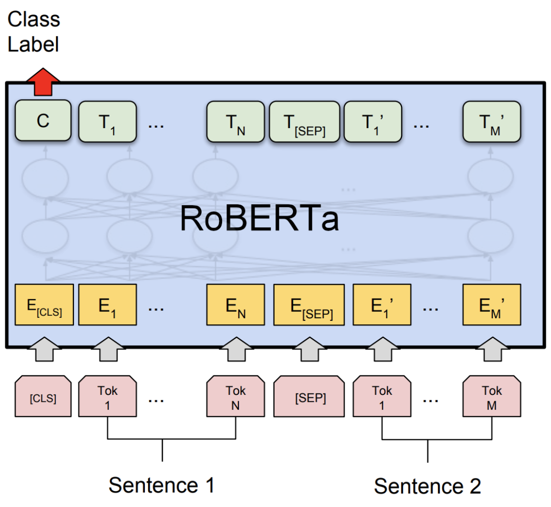


Figure 1: The RoBERTa model architecture.

2.2 XLM-RoBERTa for Binary Classification

The Arabic subtask for Tasks A and C require us to distinguish whether a tweet is sarcastic, either as a standalone text (Task A) or in comparison to another tweet (Task C). We framed these as binary classification problems and leveraged the XLM-RoBERTa$_{\text{BASE}}$ model, which is a multi-lingual version of the RoBERTa model and is pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages (Conneau et al., 2019). It is composed of 12-layers, 768 hidden, 8 self-attention heads, and 125M parameters.

In preprocessing each input Arabic tweet, we began by changing the sarcasm confidence labels from 1 to 0.8. This was again executed to account for random noise as well as subpar training data examples that did not encapsulate sarcastic qualities nearly as well as others.

Regular tweet normalization does not apply to Arabic. Certain qualities in the written form of Arabic, such as diacritization, further complicate this matter. The same word in two different diacritizations can have meanings that are seemingly completely unrelated (Alkhatib, 2017). This introduces difficulty in extracting the true semantic meaning of a text in Arabic.

We therefore relied on CAMeL Tools, a Python library designed for the Arabic language to execute dediacritization and remove any non-essential components from the input texts (Obeid et al., 2020). Further normalization was also conducted with functions from this specialized library, such as removing orthographic ambiguity.

After processing input tweets, the technique of extracting verbs and adjectives to form a secondary sequence was utilized. As in the first approach, the extracted verbs and adjectives were replaced with `<mask>` tokens in the original input tweet, and this was prepended to the secondary sequence of verbs and adjectives using a `</s>` separator token. This combined sequence was then fed into the XLM-RoBERTa's tokenizer to be encoded. In this particular case, a specialized part-of-speech tagger was used from CAMeL Tools to identify and extract each input tweet's set of verbs and adjectives.

During tokenization, padding tokens were added to the right to make all the tensor inputs of uniform length. Although the longest examined length of any tweet was 143, we utilized 256 to account for any unexpected inputs.

The final element of this system was a sequence classification head containing a linear layer that was applied on top of the final hidden-states output with a label of 1 predicting sarcasm and a label of 0 predicting otherwise. For detection of sarcasm in standalone Arabic tweets (Task A), those with a score that exceeded our threshold of 0.40 were marked as sarcastic while others were not. For determining the sarcastic tweet in a pair of tweets (Task C), the tweet with the highest absolute score, regardless of whether it exceeded Task A's threshold of 0.40, was marked as sarcastic, while the other tweet was not.

2.3 BERT Base for Multilabel Classification

Task B required us to distinguish which category of sarcasm an English tweet belonged to, of which there are six: general sarcasm, irony, satire, understatement, overstatement, and rhetorical question. A tweet can belong to none, one, or multiple categories. We framed this task as a multilabel sequence classification problem and we leveraged the BERT$_{\text{BASE}}$ model (see Figure 2 for a high-level architecture visual). It is composed of 12-layers, 768-hidden, 12 self-attention heads, and 110M parameters (Devlin et al., 2018).

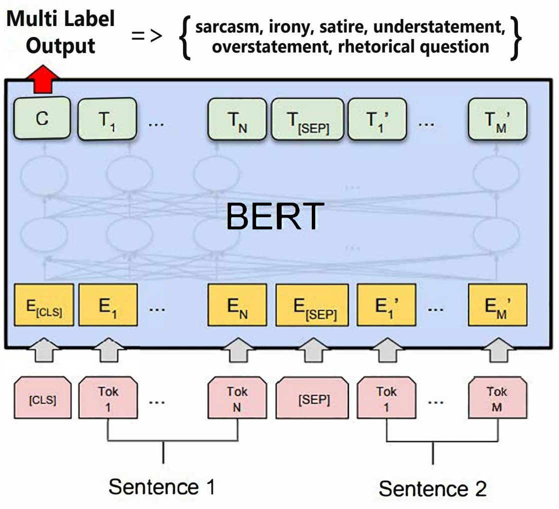


Figure 2: The BERT model architecture modified to reflect multilabel output.

Each tweet was first normalized following the steps outlined in the first approach. This included ”demojifying” any present emojis and standardizing any present hyperlinks and user tags with `"@URL"` and `"@USER"`. Before feeding the tweet into the tokenizer, however, we utilized an approach to improve the model's understanding of the keywords that might point towards a particular

Data Augmented Results	**Scores**		
	Weighted Recall	Weighted Precision	Macro-F1 Score
Baseline (No Augmentation)	0.549	0.091	0.156
Category Duplication	**0.642**	0.089	0.156
Manual Sentence Generation	0.561	**0.093**	**0.159**
GPT-3 Sentence Generation	0.552	0.082	0.151

Table 2: A summary of the performance of data-augmented systems for multilabel classification. The highest performing technique is bolded for weighted recall, weighted precision, and macro F1 score.

category of sarcasm. A TF-IDF vectorizer was used to extract the 15 most significant and representative keywords across sentences of each category. Then, for each input training tweet, all the keywords associated with the categories of sarcasm the tweet belonged to were strung together to create a secondary sequence (Sequence B) that was appended to the original input tweet (Sequence A) and separated with a separator token (`</s>`). This combined sequence was then fed into the BERT$_{\text{BASE}}$ tokenizer as a whole. This was done in an attempt to help the model be able to better seek out key phrases and words that might indicate a tweet's categorization as sarcastic, ironic, satirical, etc.

After training the multilabel classifier and generating six predictions for a tweet's likelihood of categorization in each of the six sarcastic categories, the tweet would be marked as valid for a category if its score was greater than 0.30. This was a relatively low threshold that we felt was necessary to account for the similarities across tweets belonging to different categories of satire. Furthermore, we needed to compensate for the lack of training data in categories like satire, understatement, and overstatement, which had 25, 10, and 40 training examples, respectively. By setting a lower threshold, we are able to ensure that we are not prohibitively preventing classifying any tweets as satire, understatement, overstatement, or any other category.

2.3.1 Data Augmentation

To support our efforts for multilabel classification, we explored three data augmentation techniques.

Our first technique was simply duplicating the satire, understatement, and overstatement categories to double the quantity of sentences in each of those categories. This involved copying and pasting each sentence back into the category to hopefully strengthen the model's understanding of sarcastic keywords, phrases, and qualities. We observed no meaningful improvement in results.

The second technique involved the manual generation of sentences to expand the dataset. As mentioned before, a TF-IDF vectorizer was used to extract the 15 most relevant and representative keywords for sentences across each category. See Figure 2 for some example keywords extracted through this technique. Using basic sentence templates, new training data examples were created with keywords for each satirical category being substituted into various parts of each new training datapoint. 30-40 new training examples were created for each of the satire, understatement, and overstatement categories. However, this effort did not yield meaningful or significant improvement in results.

Sarcasm	”just”, ”like”, ”really”
Understatement	”good”, ”like”, ”sorta”
Overstatement	”hate”, ”love”, ”worst”

Table 3: A subset of keywords observed as the most representative for sarcasm and under/over-statements through a TF-IDF vectorizer.

A final technique involved utilizing GPT-3 for generating new sentences. A prompt like ”Generate 10 sarcastic sentences” or ”Create 15 rhetorical questions” was provided to the model, however it was observed that the produced sentences were particularly repetitive with little variance in structure or style. The difference between most sentences produced by the model was a simple substitution in topic, object, or subject, especially as we asked the model to produce an increasing number of new sentences. A lack of training data to properly finetune the GPT-3 model was likely an issue here, and this technique was eventually dropped.

Results for each of these techniques are provided in Table 2, with the highest performing technique bolded for each metric. As observed, while most techniques seemed to perform better than the baseline, the improvements are marginal and unreliable.

2.3.2 Additional Techniques

It is worth noting that another system was explored to conduct multilabel classification. Specifically, we attempted to create 6 binary classifiers (one for each category of sarcasm) with the intent of aggregating results across all binary classifiers to mimic a multilabel classifier's output. This, however, was complicated by the severe lack of training examples for some categories as well as issues with computational capacity and consumption on Google Colab Pro. This system was eventually dropped in favor of a single multilabel classifier built with $\text{BERT}_{\text{BASE}}$, as described earlier.

3 Experimental Setup

3.1 Dataset

The task provided two training data files, one for English and another for Arabic. In each case, the task organizers provided the sarcasm labels for each tweet themselves. This avoided the need to rely on existing proxies like predefined tags or third-party labelers (Abu Farha et al., 2022).

Within the English training file, there are nine pieces of information: the tweet, a `0/1` value for the presence of sarcasm, a non-sarcastic rephrase of that tweet, and a `0/1` value for each of the various sarcasm subtypes (general sarcasm, irony, satire, understatement, overstatement, and rhetorical question). This dataset contained 3466 training examples, of which 866 were sarcastic and the remaining 2600 were not. These 866 examples are further split into multiple labels as follows: 713 for sarcasm, 155 for irony, 25 for satire, 10 for understatement, 40 for overstatement, and 101 for rhetorical question. The underresourced nature of categories like satire, understatement, and overstatement introduced challenges for our multilabel classifier system in extracting and understanding the key characteristics belonging to those categories. It is worth noting that data quality is, at times, questionable, with training examples such as *"whoop diddy scoop poop"* adding random noise into an already scarce dataset.

Within the Arabic training file, there are four pieces of information: the tweet, a `0/1` value for the presence of sarcasm, a non-sarcastic rephrase of the tweet, and a dialect label for that particular tweet (e.g. Nile, Maghreb). This training file contained 3102 training examples, of which 745 were sarcastic and the remaining 2357 were not.

3.2 Evaluation Metric

For both the English and Arabic binary classification approaches, a confusion matrix was produced to determine accuracy, precision, recall, and F-1 scores. For the multilabel classification task, weighted precision/recall for each category as well as macro F-1 scores were utilized.

3.3 Implementation Details

All three systems were developed with the PyTorch framework, HuggingFace's transformers library for the BERT, RoBERTa, and XLM-RoBERTa models, and Google Colab Pro using a single Tesla P100-PCIE-16GB GPU. For Tasks A and C, the English and Arabic binary classifiers trained for those problems shared the same hyperparameters: training and validation batch sizes of 16, a maximum sequence length of 256, 6 training epochs, and an AdamW optimizer with a learning rate of 3e-5 and epsilon value of 1e-8. For Task B, the English multilabel classifier's training policy utilized the following hyperparameter values: training and validation batch sizes of 16, a maximum sequence length of 256, 4 training epochs, and an AdamW optimizer with a learning rate of 1e-05 and epsilon value of 1e-12. All other hyperparameter values were set to their defaults according to the HuggingFace implementation. It should also be noted that for all systems, a random seed was set for the sake of reproducibility.

4 Experimental Results

Team Dartmouth received the following ranks:

- Task A (English): 13th place
- Task A (Arabic): 9th place
- Task B: 14th place
- Task C (English): 12th place
- Task C (Arabic): 10th place

Table 4 on the following page displays a tabular summary of all official scores received on each of the five subtasks, which vary from accuracy and precision to recall and macro F1 scores. As observed, our best performing system for binary classification was the English classifier developed for Task A, whereas our worst performing for binary

Binary Classification Tasks	Scores				
	F-1 Sarcastic	F1-Score	Precision	Recall	Accuracy
Task A (English)	0.386	0.635	0.625	0.648	**0.804**
Task A (Arabic)	0.350	0.529	0.581	0.665	0.597
Task C (English)	-	0.659	-	-	0.660
Task C (Arabic)	-	0.679	-	-	0.680

Multilabel	Scores						
	Macro F1	F1-Sarcasm	F1-Irony	F1-Satire	F1-Under	F1-Over	F1-Rhet-Q
Task B	0.0590	0.2293	0.0202	0.0824	0.0000	0.0077	0.0143

Table 4: A tabular summarization of the performance of all three systems across all five subtasks, reporting various metrics including accuracy, precision, recall, and regular/macro F-1 scores. Experiments revealed that further data augmentation did not improve the scores of any system.

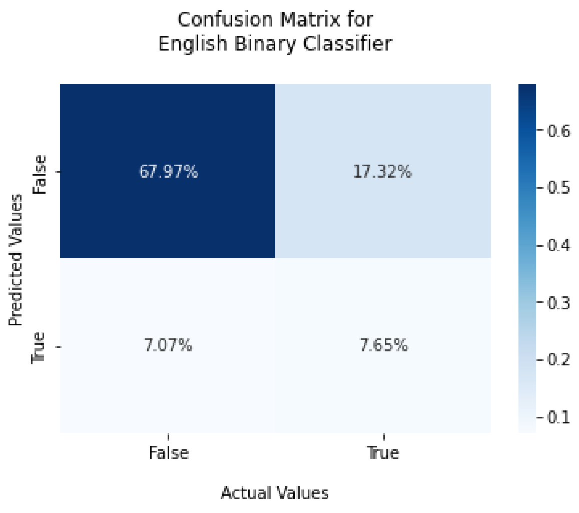


Figure 3: A confusion matrix of the English binary classifier developed for Tasks A and C.

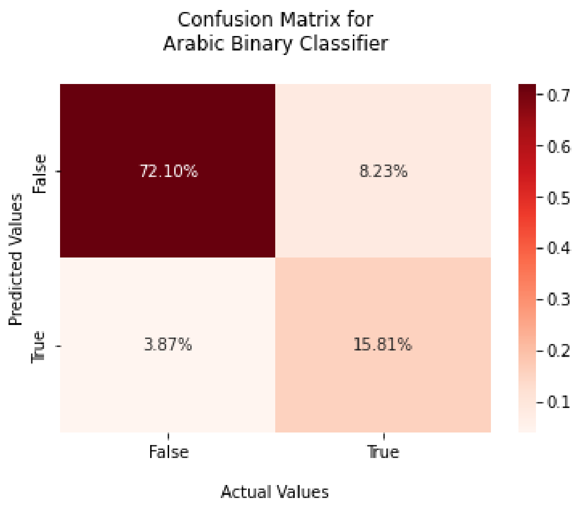


Figure 4: A confusion matrix of the Arabic binary classifier developed for Tasks A and C.

classification was the Arabic classifier also developed for Task A. For the multilabel classification task, macro F-1 as well as weighted categorical scores are provided.

Confusion matrices displaying our best development metrics are provided in Figures 3 and 4. This reveals insights into the relatively imbalanced split of the training dataset, creating issues which were further compounded by the overall small number of training examples.

4.1 Case Study: Task A (English)

It's worth exploring Task A (English) in greater detail to understand the elements that factored into our model's scores. To begin, it should be noted that certain inputs in the test dataset for this subtask were sometimes single tweets like ”Followed” or ”Pinball!”, while other tweets were random noise, such as the following:

> ”20:00 GMT:
> Temp: 13.7°C,
> Wind: SSW, 3 mph (ave), 8 mph (gust),
> Humidity: 92
> Rain (hourly) 0.0 mm,
> Pressure: 1017 hPa, rising slowly.”

The prevalence of random noise such as the above in the test set can make it somewhat challenging for the model at hand to be able to relate the test input to what it has learned. There's very little context to learn from in one-word tweets like the ones mentioned, and this may bias the model towards marking such tweets as non-sarcastic, when in reality they may be sarcastic (e.g. perhaps the one-word tweet was a sarcastic remark towards another tweet). Granted, this is the nature of text in short-form online social forums like Twitter, but it does contribute to a concrete decrease in model performance.

Furthermore, our technique of masking verbs and adjectives in the original tweet while simultaneously stringing those verbs and adjectives together into a second sequence to be fed into the tokenizer alongside the tweet input may have overfit the model towards certain words and phrases from the training dataset. While this may have been helpful in identifying sarcastic tweets which did include those words, it may also have caused the model to overlook other sarcastic sentences that did not include them in the test dataset.

As such, random noise, poor quality, and certain learning techniques may have been factors in contributing to the scores received by our binary and multilabel classification systems. The same observations apply to Tasks A and C in Arabic as well as Task B. In particular, our technique of extracting and appending the top 15 words for each category a tweet belongs to may have inadvertently overfit the model to overlook other textual signals that indicate a tweet's sarcastic categorization in preference for certain words and phrases.

5 Conclusion and Future Work

In this paper, we have described three binary and multilabel sequence classification systems using the BERT, RoBERTa, and XLM-RoBERTa architectures from HuggingFace for the detection of sarcasm in English and Arabic tweets. We found that additional work to augment the training data with duplication of sentences and manually/automatically synthesizing new sarcastic sentences did not improve the results of the model. Furthermore, challenges were observed with the multilabel classifier in learning to extract the key characteristics that categorize a tweet as a distinct example of satire, understatement, or overstatement – categories which were generally underresourced in the training dataset.

Further investigation could include implementing six binary classifiers instead of a single multilabel sequence classifier for Task B. Given enough training time, data, and resources, it could certainly be the case that aggregating results across specialized binary classifiers provide more concrete results than what has been produced. In particular, this may allow each of the binary classifiers to more deeply learn the unique characteristics, keywords, and structure of the sarcastic sentences it ingests.

It would also be interesting to see how the results across all three systems change with sufficient training data, with perhaps tens of thousands of more valid examples that can allow the models to truly capture the essence of sarcasm across a wide and varied set of training examples.

6 Acknowledgements

Rishik Lad is grateful to be supported by the James O. Freedman Presidential Scholars program at Dartmouth College.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Manar Alkhatib. 2017. *Challenges in Arabic Natural Language Processing*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

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

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

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7022–7032, Marseille, France. European Language Resources Association.

(Liu et al., 2019) (Conneau et al., 2019) (Abu Farha et al., 2022) (Obeid et al., 2020) (Alkhatib, 2017) (Devlin et al., 2018)

ISD at SemEval-2022 Task 6: Sarcasm Detection Using Lightweight Models

Samantha Huang
Lynbrook High School
Saratoga, CA
sam.y.huang2005@gmail.com

Ethan A. Chi
Stanford University
Stanford, CA
ethanchi@cs.stanford.edu

Nathan A. Chi
De Anza College
Cupertino, CA
chinathan
@student.deanza.edu

Abstract

Robust sarcasm detection is critical for creating artificial systems that can effectively perform sentiment analysis in written text. In this work, we investigate AI approaches to identifying whether a text is sarcastic or not as part of SemEval-2022 Task 6. We focus on creating systems for Task A, where we experiment with lightweight statistical classification approaches trained on both GloVe features and manually-selected features. Additionally, we investigate fine-tuning the transformer model BERT. Our final system for Task A is an Extreme Gradient Boosting Classifier (XGB Classifier) trained on manually-engineered features. Our final system achieved an F1-score of 0.2403 on Subtask A and was ranked 32 of 43.

1 Introduction

Sarcasm is the use of irony–which communicates the opposite of what is said–to humorous and derisive effect (Bouazizi and Ohtsuki, 2016). On the web, sarcasm is ubiquitous–not least because social media users often apply sarcasm to incorporate a sardonic sense into their statements (Hancock, 2004). This poses a substantial challenge to artificial systems evaluating tasks including sentiment analysis (Liu and Zhang, 2012). It is already a challenge enough for human annotators to determine what is intended to be taken at face-value or not in context-lacking text; it is even more difficult for NLP systems to distinguish between what should be taken literally and what is sarcastic.

Task 6 of SemEval-2022 (Abu Farha et al., 2022) provides an environment to build systems to approach these challenges. In particular, Subtask A in Task 6 (Table 1) of SemEval-2022 tests the ability of automated systems to determine whether a text is sarcastic or non-sarcastic. We investigate whether lightweight models (which use few computational resources) are able to effectively identify sarcastic speech; we also experiment with fine-tuned Transformer-based models to identify whether larger models perform better.

2 Dataset

In Subtask A, we train our models on the official SemEval-2022 Task 6 English training set, which was curated from a set of tweets. Each sentence (examples in Table 6) has been annotated for sarcasm status by the text authors themselves, with 1 denoting a sarcastic text and 0, a non-sarcastic text.

Task	Description	Metric
A	Determine whether a given text is sarcastic or non-sarcastic.	F1, sarcastic-class

Table 1: Subtask A overview.

2.1 Train-test Split

The dataset has a total of 4868 examples, with 3468 being part of the training set, and 1400 being part of the test set. In total, there are 867 sarcastic and 2601 non-sarcastic texts in the training set. As the testing set labels were not provided until after the competition, we created our own validation set with a 75 : 25 train : test split. Thus, our train set has 2774 examples and our test set has 693 examples.

Feature	Description	LR Coefficient
POL	Words referring to political leaders (e.g. "Boris," "Trump").	0.127
GAY	The word "gay."	0.008
BANG	The character "!".	-0.067
AT	The character "@".	0.005
DEFINITELY	The word "definitely."	0.035
PLEADING FACE	The pleading face emoji.	0.136
EMOJI	The grinning face emoji.	0.095
HASH	The character "#".	-0.095
THANK	The word "thank."	0.008
HAHA	The word "haha."	0.095

Table 2: Manual features for sarcasm detection.

Model	F1 Sarcastic	F-score	Precision	Recall	Accuracy
XGBClassifier	0.2403	0.1332	0.3651	0.4792	0.1464

Table 3: Official test set performance of our best-performing lightweight model (XGBClassifier trained with manual features) on Subtask A (binary classification). LR coefficient represents the linear regression coefficient value for the given feature.

3 Methods

3.1 Subtask A: Sarcasm Detection

This subtask examines whether a given text is sarcastic, and we investigate using the following models. Our lightweight machine learning models were implemented using the Scikit-learn library (Pedregosa et al., 2011):

- **Logistic Regression** is a supervised learning algorithm that predicts a binary outcome using a logistic function.
- **GaussianNB** is a type of Naive Bayes algorithm used for continuous data that follows a normal distribution (Qiu et al., 2020).
- **SVM** is a non-probabilistic binary linear supervised learning algorithm that can be used for classification and regression (Yu and Kim, 2012).
- **AdaBoostClassifier** is a meta-algorithm that assigns higher weights to incorrectly classified samples to improve the following classifiers (Solomatine and Shrestha, 2004).
- **XGBClassifier** stands for e**X**treme **G**radient **B**oosting **Classifier** and is a decision tree based algorithm that uses gradient boosting methods to avoid overfitting (Kumar et al., 2021).
- **BERT** is a transformer based algorithm that uses masked language modeling. We fine-tune BERT–an approach commonly used in tasks such as sentiment prediction–which was pretrained on language modelling and next-sentence prediction tasks. In particular, we use BERT base cased, BERT large cased, BERT base uncased, and BERT large uncased (Devlin et al., 2018).

3.2 Results

On the unofficial evaluation set, XGBClassifier performed the best compared to other models. On the official evaluation set, we achieve a F1-score of 0.2403. We were ranked 32 out of 43. Our official and unofficial results are listed in Table 3 and Table 4 respectively. The hyperparameters that we used for all models trained on manual features is included in Table 5.

4 Conclusion

Our models were trained to determine whether texts were sarcastic or not. For the most part, our models struggled to detect sarcasm in text—-as was expected, given that the task was quite challenging even for humans. We find that the models that achieve the highest degree of success in detecting sarcasm were GaussianNB and XGBClassifier models.

Model	Features	positive-class F1	Accuracy	Normalize
LogisticRegression	Manual	0.44	0.34	True
LogisticRegression	GloVe	0.09	0.71	False
GaussianNB	Manual	0.43	0.34	True
GaussianNB	GloVe	0.42	0.47	False
SVM	Manual	0.30	0.63	True
SVM	GloVe	0.08	0.72	False
AdaBoostClassifier	Manual	0.06	0.72	False
AdaBoostClassifier	GloVe	0.23	0.68	False
XGBClassifer	**Manual**	**0.45**	**0.32**	**False**
XGBClassifier	Glove	0.38	0.59	False
BERT base cased	–	0.32	0.63	False
BERT large cased	–	0.14	0.50	False
BERT base uncased	–	**0.40**	**0.75**	**False**
BERT large uncased	–	0.26	0.63	False

Table 4: Unofficial validation set performances of candidate models. For this task, the highest-performing lightweight model is XGBClassifier and the highest-performing transformer model is BERT base uncased.

We also find that using manual features, as listed in Table 2, is a fruitful approach to determining the sarcasm status of a sentence. In particular, we preprocess the data by identifying the number of instances of characters or words described in each feature category, then train our models on these summed feature values. Our top-scoring classifiers yielded substantially greater postive-class F1 scores with manual features than with automatic GloVe features. That being said, it should be noted that using these manual features also lowered the accuracy greatly, which indicates a tradeoff between F1 score and accuracy due to the extreme class imbalance of the dataset.

Finally, fine-tuning BERT achieves reasonable results while detecting sarcasm. However, this method is still inferior to a lightweight approach.

Overall, our best model, the XGBClassifier with manually engineered features, did not perform significantly better than the Logistic Regression model. Our results demonstrate that boosting algorithms can predict sarcasm in text to a moderate degree of success.

Acknowledgments

The authors would like to acknowledge Google Colaboratory for their free compute services.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jeffrey T Hancock. 2004. Verbal irony use in face-to-face and computer-mediated conversations. *Journal of Language and Social Psychology*, 23(4):447–463.

Munish Kumar, Manish Kumar, et al. 2021. Xgboost: 2d-object recognition using shape descriptors and extreme gradient boosting classifier. In *Computational Methods and Data Engineering*, pages 207–222. Springer.

Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In *Mining text data*, pages 415–463. Springer.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Ming Qiu, Yiru Zhang, Tianqi Ma, Qingfeng Wu, and Fanzhu Jin. 2020. Convolutional-neural-network-based multilabel text classification for automatic discrimination of legal documents. *Sens. Mater*, 32(8):2659–2672.

Dimitri P Solomatine and Durga L Shrestha. 2004. Adaboost. rt: a boosting algorithm for regression problems. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, volume 2, pages 1163–1168. IEEE.

Hwanjo Yu and Sungchul Kim. 2012. Svm tutorial-classification, regression and ranking. *Handbook of Natural computing*, 1:479–506.

Model	**Hyperparameter**	**Task 1a**
GaussianNB	priors	0.025, 0.975
	var_smoothing	1e-09
SVM	class_weight	balanced
	C	1.0
	kernel	rbf
	degree	50
AdaBoostClassifier	base_estimator	max_depth = 1
		class_weight = {0: 0.1, 1: 0.9}
	$n_{estimators}$	50
	learning_rate	1.0
	loss	linear
XGBClassifier	$n_{estimators}$	100
	max_depth	5
	eta	0.3
	min_child_weight	1
	booster	gbtree

Table 5: Hyperparameters for best-performing Manual models.

Sentence	**Sarcastic**
yeah your girl is fine but does she pass out while giving blood	1
just impulse bought a mandolin and in 3-5 buisness days i will impulse learn some jigs	0

Table 6: Examples that are sarcastic and not sarcastic, respectively.

Plumeria at SemEval-2022 Task 6: Sarcasm Detection for English and Arabic Using Transformers and Data Augmentation

Mosab Shaheen[*]
Indian Institute of Technology Kanpur
mosab@iitk.ac.in

Shubham Kumar Nigam[*]
Indian Institute of Technology Kanpur
sknigam@iitk.ac.in

Abstract

The paper describes our submission to SemEval-2022 Task 6 on sarcasm detection and its five subtasks for English and Arabic. Sarcasm conveys a meaning which contradicts the literal meaning, and it is mainly found on social networks. It has a significant role in understanding the intention of the user. For detecting sarcasm, we used deep learning techniques based on transformers due to its success in the field of Natural Language Processing (NLP) without the need for feature engineering. The datasets were taken from tweets. We created new datasets by augmenting with external data or by using word embeddings and repetition of instances. Experiments were done on the datasets with different types of preprocessing because it is crucial in this task. The rank of our team was consistent across four subtasks (fourth rank in three subtasks and sixth rank in one subtask); whereas other teams might be in the top ranks for some subtasks but rank drastically less in other subtasks. This implies the robustness and stability of the models and the techniques we used.

1 Introduction

Sarcasm is a figurative language where speakers or writers usually mean the contrary of what they say. Recognizing whether a speaker or writer is sarcastic is essential to downstream applications to understand the sentiments, opinions, and beliefs correctly (Ghosh et al., 2020). Sarcasm is ubiquitous on the social media text and, due to its nature, can be highly divisive of computational systems that perform tasks on that kind of data such as sentiment analysis, opinion mining, and harassment detection (Van Hee et al., 2018; Bing, 2012; Rosenthal et al., 2014; Maynard and Greenwood, 2014).

Our team Plumeria participated in SemEval 2022 task 6 (Abu Farha et al., 2022) in all its subtasks on English and Arabic. Previous shared tasks on sarcasm detection (Hee et al., 2018; Ghanem et al., 2019; Ghosh et al., 2020; Abu Farha et al., 2021) have only two subtasks; one is sarcasm detection and another is predicting the type of sarcasm. However, in SemEval 2022 task 6 (Abu Farha et al., 2022) organizers formulate three subtasks for both languages following the methods described in (Oprea and Magdy, 2020).

Using the two datasets for English and Arabic, organizers formulate three subtasks as follows:

- **Subtask A (English and Arabic):** It is a binary classification subtask where submitted systems have to predict whether a tweet is sarcastic or not.
- **Subtask B (for English only):** It is a multi-label classification subtask where submitted systems have to predict one or more labels out of six ironic-speech labels: sarcasm, irony, satire, understatement, overstatement, and rhetorical question.
- **Subtask C (English and Arabic):** It is a binary classification subtask. Given two texts, a sarcastic tweet and its non-sarcastic rephrase which conveys the same meaning, submitted systems have to predict which text is the sarcastic one.

In this shared task, our submitted systems primarily focused on the transformer based approaches because of their success in the field of NLP. The multi-head attention mechanism in transformers captures the relations between the words in a sentence which helps in identifying sarcasm. Moreover, to capture long-term dependencies between words, especially the contradicting ones, we used a hierarchical network by stacking a BiLSTM layer on top of a transformer. In order to emphasize the important tokens afterwards, we tried adding a dot-product attention layer to give different weights to the tokens. For prediction, the final information

[*]These authors contributed equally to this work

are passed to a fully connected layer followed by a linear layer with a softmax activation function for classification (i.e. subtask A and C) or a sigmoid activation function for multi-label classification (i.e. subtask B).

The major constraint of a neural network is that they need lots of data for training to give satisfactory results. In addition, if the dataset is imbalanced with few instances of a class, this can result in poor results for detecting instances that belong to the class. This motivates us to create biased datasets, towards the concerned class/label, from existing datasets by increasing the number of instances of such a class/label. A detailed explanation of dataset creation is in section 3. We also illustrate the composition of created datasets in each subtask and the performance of the models on them.

The rank of our team was consistent across most subtasks (fourth rank in three subtasks, sixth rank in one subtask, and tenth rank in one subtask); unlike many teams which scored high in one or two subtasks but scored considerably less in other subtasks. This shows that the robustness and the consistency of our methods. The biased datasets were crucial for subtask A and subtask B, while augmenting a dataset plays a key role in subtask C. We released the codes and datasets for all subtasks via GitHub[1].

The paper is organized as follows. Section 2 lists some abbreviations used across the paper. Section 3 shows the datasets we used for fine-tuning the models. In Section 4 we list the preprocessing types applied on the datasets. The experiments and results are presented in section 5, and the analysis of the results is presented in section 6. This is followed by a conclusion in section 7.

2 Abbreviations

The following abbreviations were used frequently, and are shown in Table 1.

Full Form	Abbreviation
Language	Lang
Sarcastic	S
Non-Sarcastic	NS
English	En
Arabic	Ar
External	Ext
True Positive	TP
False Positive	FP
True Negative	TN
False Negative	FN

Table 1: Abbreviations used in the paper.

3 Datasets

The organizers provided datasets for English and Arabic. Regardless of the dataset, these are the fields in each row of a dataset:

- **Tweet**: a text specifying a tweet. This field is for all subtasks.
- **Sarcastic**: a binary field specifying whether a tweet is sarcastic or not. This field is for subtask A.
- **Rephrase**: a text specifying a non-sarcastic rephrase of a sarcastic tweet. This field is for subtask C.
- **Sarcasm, Irony, Satire, Understatement, Overstatement, and Rhetorical question** (English only): These binary fields are the labels of a sarcastic tweet. These fields are for subtask B.
- **Dialect** (Arabic only): a text specifying the dialect of a tweet from one of five dialects: Modern Standard Arabic (MSA), Egyptian, Levantine, Maghrebi, and Gulf. This field is for subtask A and C.

In the following sections we will describe the datasets given by the organizers, other available datasets, and augmented datasets.

3.1 Datasets Given by Organisers (Original)

The organizers released two training datasets for Arabic and English to train the systems on them for all subtasks. Later on, they released a test dataset for each subtask. We called these datasets "original" datasets as they are the official datasets for the subtasks. Information about the distribution of sarcastic and non-sarcastic tweets is presented in Appendix A.7 in Table 27 and Figure 6. Furthermore, information about sarcastic labels for subtask B is presented in Appendix A.7 in Table 28 and Figure 7.

3.2 Other Available Datasets (External)

The datasets in this section are not the official datasets and thus we called them "external" datasets. However, we used these datasets for subtask A and subtask B as they are created for similar subtasks.

[1]https://github.com/mosab-shaheen/iSarcasm-SemEval-2022-Task-6

Datasets Downloaded Using Twitter API: Initially the organizers provided the participants with train and test datasets which covered subtask A and subtask B for English only (later on the original datasets explained in subsection 3.1 were released instead). However, they provided the tweet ID instead of the tweet text and they asked the participants to download the tweet text using the Twitter API[2] and the tweet ID. Therefore, we downloaded the tweet texts we found for these two datasets. We were able to download 2841 tweets for training and 713 tweets for testing as shown in Table 27. The distribution of the tweets over the sarcastic labels is presented in Table 28.

Datasets of SemEval-2018 Task 3: These datasets are on same subtasks of subtask A and subtask B but for SemEval 2018 (Hee et al., 2018). We used the dataset for subtask A which has emojis and sarcasm hashtags. The datasets are available for download in this link[3]. More information about the distribution of sarcastic and non-sarcastic tweets is presented in Table 27.

ArSarcasm-v2 Dataset: It contains train and test datasets for sarcasm detection in Arabic (Abu Farha et al., 2021). Each row contains a tweet, sarcastic class, sentiment, and dialect. The datasets are available for download in this link[4]. More information about the distribution of sarcastic and non-sarcastic tweets is presented in Table 27.

3.3 Augmented Datasets

In addition to the original and external datasets, we created more datasets with more number of instances using the following methods:

1. **Augmenting an original dataset with external datasets:** We added instances to an original dataset from the matching external datasets either to balance it or just to augment it, and we filtered out the NAN entries .

2. **Augmenting a dataset using word embeddings:** For word embeddings we used Gensim library[5] together with GloVe word vectors[6] trained on two billion tweets with 100 dimension word vectors (glove-twitter-100). To create new instances in a dataset, we took a copy of one instance in the dataset and replaced up to four keywords in a tweet (or its rephrase) by replacing each keyword with one of the top three similar words according to the similarity between the corresponding word vectors, then we added the copied modified instance to the dataset and we repeated the process for other instances till we reached the required number of instances.

3. **Augmenting a dataset by repeating instances:** We repeated instances from a dataset mostly to balance the classes/labels in a dataset.

The final datasets used for each subtask is explained in the dedicated section for it.

4 Preprocessing

- **Type I:** no preprocessing

- **Type II:** Emotion icons were converted to their string text using the "emoji" Python library. Then, URLs were converted to "HTTPURL" token, also every mention in a tweet was converted to "@USER" token using regular expressions. These conversions was done because the BERTweet model (we will talk about it later) was pre-trained on tweets after these conversions.

- **Type III:** same as in Type II besides converting the smiley face codes e.g. ":-)" and ":)" to one of three values (smiley, sad, and playful). More than two successive occurrences of any punctuation like in "why?!!!!" were removed, then we removed more than two successive occurrences of same character like in "Superrrr" which can be found frequently in tweets. Moreover, a contraction (e.g. "isn't and "'cause") was replaced with its full form (e.g. "is not" and "because").

- **Type IV:** same as in Type III besides stemming and stop-word removal. For English we used WordNet lemmatizer and for Arabic we used ISRI stemmer. The NLTK Python library[7] was used for this purpose.

5 Approaches and Results for Subtasks

5.1 Conventions

In the following tables, if a table cell is highlighted with a light brown color, it means the score is

[2]developer.twitter.com/en/docs/api-reference-index
[3]SemEval2018-Task3 Dataset
[4]ArSarcasm-v2
[5]Gensim Library
[6]GloVe Word Vectors
[7]NLTK Python Library

among the best results, in the corresponding section of the table, on the validation dataset; whereas the one with brown color is the highest score. Furthermore, if a cell is highlighted with a green color, it means that the score is our final submission score (released by the organizers) in the subtask on the test dataset; whereas the one with blue color is the score of a submitted model but not the final one (a team can have multiple submissions).

5.2 Subtask A (English)

5.2.1 Datasets

- **Original**: The train split of the original dataset for English in Table 27 is splitted into train and validation datasets as shown in Table 2.
- **External**: As the measure for this task is F1-score for the sarcastic class, thus we created datasets which are biased towards the sarcastic class as shown in Table 29 in Appendix A.7.

Dataset	Total	S%	NS%
Original Train	2080	25	75
Original Val	1388	25	75

Table 2: Original datasets for subtask A (English) with the total number of tweets and the percentage of sarcastic (S%) and non-sarcastic (NS%) tweets.

5.2.2 Approaches

We primarily focused on the transformer based models. Since the task is a binary classification on tweets, the excellent choice to start with is BERTweet-base[8] and BERTweet-large[9] (Nguyen et al., 2020), a pre-trained language model on 845M English Tweets. Likewise, we tried the ELECTRA[10] (Clark et al., 2020) replaced token detection model (a pre-training task in which the model learns to distinguish real input tokens). In ELECTRA model, some tokens in the input are replaced with sample tokens instead of masking the tokens as in BERT. Moreover, we used a hierarchical network by passing the input tokens to the BERT model, then each token embedding is passed to a Bi-LSTM layer either with or without attention. The architecture of the BERT model, ELECTRA model, and hierarchical network is shown in Appendix A.3, A.4, and A.1 respectively. The final layer of each model was a linear layer with softmax activation function and we used the cross entropy loss function.

[8]HuggingFace Bertweet-Base
[9]HuggingFace Bertweet-Large
[10]HuggingFace Electra Large Discriminator

Note: We ran several experiments on all the approaches of this subtask with different datasets and preprocessing types. We also experimented with different learning rates, epochs, and loss functions to verify which one is performing best.

5.2.3 Results

Metric: The main metric is F1-score for the sarcastic class.

BERT: We used BERTweet-large in the following experiments, as it gave better performance than BERTweet-base, on the original train dataset shown in Table 2. We experimented with different learning rates and preprocessing types, and ran for 5 epochs. The results are shown in Table 3.

Learning Rate	Type	Val	Test
2 e - 6	I	0.0057	0.0293
	II	0.5017	0.457
	III	0	0
	IV	0	0
3 e - 6	I	0.3786	0.5068
	II	0.5552	0.4874
	III	0.5405	0.4972
	IV	0	0
4 e - 6	I	0.5585	0.4981
	II	0.4926	0.4717
	III	0.5407	0.4772
	IV	0	0
5 e - 6	I	0.5275	0.4724
	II	0.5655	0.4841
	III	0	0
	IV	0	0

Table 3: F1-score of the BERT model for subtask A (English) on original datasets.

From these experiments we found that Type II preprocessing is performing better than other types and same applies for the learning rate 4e-6. We conducted similar experiments on the external datasets in Table 29 in Appendix A.7 and we found similar results. We tried using the cross entropy loss function with and without weights on the external datasets using the same learning rate, preprocessing type, and number of epochs. We got our best result on the B4 dataset with weighted loss function which was our final submission score for this subtask. The results are shown in Table 4.

Biased	Loss1: Without Weights		Loss2: W1=1/#NS, W2=1/#S	
	Val	Test	Val	Test
B0	0.5784	0.4487	0.5944	0.4519
B1	0.5714	0.4548	0.5738	0.479
B2	0.5767	0.465	0.5951	0.4917
B3	0.601	0.4626	0.5931	0.4817
B4	0.5954	0.4727	0.6025	0.4769
B5	0.5874	0.5142	0.5803	0.5008
B6	0.5858	0.4791	0.5624	0.4884
B7	0.5957	0.5016	0.5637	0.5034
B8	0.584	0.492	0.5814	0.5
B9	0.5723	0.487	0.5554	0.49

Table 4: F1-score of the BERT model for subtask A (English) on external datasets.

We used 5 epochs and 4e-6 learning rate because they gave the best results as shown in Appendix A.5.

The official scores and leader-board ranks of the teams for subtask A (English) are shown in Table 5.

Rank	User	F-1 sarcastic
1	stce	0.6052
2	emma	0.5691
3	saroyehun	0.5295
4	**ShubhamKumarNigam**	**0.4769**

Table 5: Scores and leader-board ranks for subtask A (English).

ELECTRA: We used the ELECTRA model on the external datasets with Type II preprocessing, 6e-6 learning rate, and 5 epochs as they were performing the best as shown in Table 6.

Biased	Val	Test
B0	0.5525	0.4684
B1	0.4002	0.25
B2	0.5738	0.4762
B3	0.4002	0.25
B4	0.5756	0.4879
B5	0.4002	0.25
B6	0.5468	0.4642
B7	0.5702	0.4789
B8	0.479	0.5073
B9	0.4002	0.25

Table 6: F1-score of the ELECTRA model for subtask A (English) on external datasets.

BERT+BiLSTM with and without attention: The results we got using this architecture were not deterministic (i.e. they change when re-running the experiment and they may become better or worse than the results of BERT alone) and thus we did not use this model for the official submission. More details about the results of the model can be found in Appendix A.6.

5.3 Subtask A (Arabic)

5.3.1 Datasets

Original: The train split of the original dataset for Arabic in Table 27 is splitted into train and validation datasets as shown in Table 7.

Dataset	Total	S%	NS%
Original Train	1861	24	76
Original Val	1241	24	76

Table 7: Original datasets for subtask A (Arabic).

External: Same as in subtask A (English), we created datasets which are biased towards the sarcastic class as shown in Table 30 in Appendix A.7.

5.3.2 Approaches

The approaches used here are similar to subtask A (English) except for the used transformers. Since the data is in the Arabic language, we tried some models from The **C**omputational **A**pproaches to **M**odeling **L**anguage (CAMeL) research lab [11]. They majorly focused on Arabic and Arabic dialect processing, machine translation, text analysis, and dialogue systems.

The models are available on the Hugging Face library. CAMeLBERT is a collection of BERT models pre-trained on Arabic texts with different sizes and variants (Inoue et al., 2021). They released pre-trained language models for Modern Standard Arabic (MSA), dialectal Arabic (DA), and classical Arabic (CA). We tried CAMeLBERT-DA and CAMeLBERT-Mix for sarcasm detection. Likewise, we tried the AraBERT v2 which is a pre-trained BERT based on Google's BERT architecture for Arabic Language Understanding[12] (Antoun et al.).

5.3.3 Results

Metric: The main metric is F1-score for the sarcastic class.

BERT: We used CAMeLBERT-Mix in the following experiments as it performed the best among other BERT models. We applied it on the external datasets with non-weighted cross entropy loss function, 5 epochs, and 2e-5 learning rate because they

[11] HuggingFace CAMeL-Lab
[12] HuggingFace Bert-Base-Arabert-v02

gave the best results, which included our final submission score for this subtask, as shown in Table 8.

Biased	Val	Test
B0	0.7168	0.3438
B1	0.7025	0.4163
B2	0.7131	0.4071
B3	0.6804	0.4335
B4	0.7015	0.4186
B5	0.6927	0.4332
B6	0.7012	0.4048
B7	0.7124	0.4365
B8	0.6731	0.4589
B9	0.7094	0.4589

Table 8: F1-score of the BERT model for subtask A (Arabic) on external datasets.

BERT+BiLSTM+Attention: We used attention with BiLSTM on top of BERT model. The results also were not deterministic. However, the best results for this architecture occurred when using 5 epochs and 9e-6 learning rate on B3 and B9 datasets as shown in Table 9.

Biased	Hidden State Size	Val	Test
B3	50	0.6849	0.4234
B9	1000	0.7123	0.4693

Table 9: F1-score of the BERT+BiLSTM+Attention model for subtask A (Arabic).

The official scores and leader-board ranks of the teams for subtask A (Arabic) are shown in Table 10.

Rank	User	F-1 sarcastic
1	Abdelkader	0.5632
2	Aya	0.5076
3	rematchka	0.4767
10	**ShubhamKumarNigam**	**0.4072**

Table 10: Scores and leader-board ranks for subtask A (Arabic).

5.4 Subtask B

5.4.1 Datasets

Original: The train split of the original dataset for English in Table 28 in Appendix A.7 is splitted into train and validation datasets as shown in Table 11.

External: The original and external datasets presented in Table 28 in Appendix A.7 (without

Dataset	Total	Sarcasm Under-statement	Irony Over-statement	Satire Rhetorical question
Original Train	606	67.60% 1%	15.10% 3.50%	2.60% 10.20%
Original Val	261	70% 1%	14.20% 4.50%	1.90% 8.40%

Table 11: Original datasets for subtask B (English).

the validation dataset) were added together to form a new dataset (Ext-NB). Then the resulting dataset was balanced either by using word embeddings (Ext-UW) or by repeating instances (Ext-UR). We created a dataset (Ext-EB) to give more importance to the labels of low number of instances by repeating the instances of these labels up to the limits specified by these heuristic formulas:

#irony=#sarcasm*(1+1/sqrt(#irony)) (1)

#satire=#sarcasm*(1+2/sqrt(#satire)) (2)

#understatement=#sarcasm*(1+3/sqrt(understatement)) (3)

#overstatement=#sarcasm*(1+1.5/sqrt(overstatement)) (4)

#rhetorical=#sarcasm*(1+1.2/sqrt(#rhetorical)) (5)

The datasets are shown in Table 31 in Appendix A.7.

5.4.2 Approaches

This subtask primarily focused on BERTweet-large. As it is a multi labeling subtask, we used sigmoid as the activation function in the last layer and binary cross entropy as the loss function.

5.4.3 Results

Metric: The main metric is Macro-F1 score.

BERT: We used BERTweet-large in the following experiments on the external datasets using 5 epochs, 6e-6 learning rate, and Type II preprocessing. The results are shown in Table 12 which included our final submission score for this subtask.

Dataset	Val	Test
Ext-NB	0.1513	0.038
Ext-UW	0.318	0.0716
Ext-UR	0.3412	0.076
Ext-EB	0.4152	0.0778

Table 12: Macro-F1 score of the BERT model for subtask B (English) on external datasets.

The official scores and leader-board ranks of the teams for subtask B (English) are shown in Table 13.

Rank	User	macro F1-score
1	Duxy	0.163
2	Abdelkader	0.0875
3	robvanderg	0.0851
6	**ShubhamKumarNigam**	**0.0778**

Table 13: Scores and leader-board ranks for subtask B (English).

5.5 Subtask C (English)

5.5.1 Datasets

Original: The train split of the original dataset for English in Table 27 in Appendix A.7 is splitted into train and validation datasets. As we do not have external datasets here, so we augmented the train split once with word embeddings (Original-Embedding) and once with repetition (Original-Repetition). We swapped between the tweet and its rephrase for half of the instances together with flipping the value of the sarcastic field, so that the model will be able to learn. Otherwise, it may always predict the first text as the sarcastic tweet and the second one as its non-sarcastic rephrase. The datasets are shown in Table 14.

Dataset	Total
Original-Train	606
Original-Validation	261
Original-Embedding	1606
Original-Repetition	1606

Table 14: Original datasets for subtask C (English).

5.5.2 Approaches

Organizers provide a sarcastic text, and its non-sarcastic rephrase, i.e., two texts convey the same meaning. Since the input format changed in this subtask, we input both texts together to the BERT model as one text separating them by the separating token. We focused on transformers which are trained on question-answering tasks. We tried BERT models trained on the **S**tanford **Q**uestion **A**nswering **D**ataset (SQuAD) dataset (Rajpurkar et al., 2018). SQuAD is a reading comprehension dataset consisting of questions posed by crowdworkers on a set of Wikipedia articles.

We took models from the Hugging Face library; one is BERT large model (cased)[13], trained on whole word masking, and fine-tuned on the SQuAD dataset. Another is BERT base model (uncased)[14], trained on Masked language modeling (MLM), and fine-tuned on the SQuAD dataset.

[13]HuggingFace Bert-Large-Cased-Whole-Word-Masking-Finetuned-Squad

[14]HuggingFace Bert-Base-Uncased-Squad2

5.5.3 Results

Metric: The main metric is the accuracy.

BERT: We used BERT large model (cased) on the original datasets, because it gave better results compared to the other models, using 15 epochs, 8e-6 learning rate, and the cross entropy as the loss function. The results are shown in Table 15. We did our submission using Type I preprocessing as it gave the best result on the validation dataset.

Dataset	Type	Val	Test
Original-Training	I	0.951	0.79
	II	0.9395	0.8
	III	0.9193	0.83
	IV	0.9135	0.79
Original-Embedding	I	0.9454	0.83
	II	0.9385	0.8
	III	0.9366	0.82
	IV	0.8588	0.72
Original-Repetition	I	0.9395	0.8
	II	0.9222	0.815
	III	0.9078	0.8
	IV	0.9078	0.68

Table 15: Accuracy of the BERT model for subtask C (English) on original datasets.

The official scores and leader-board ranks of the teams for subtask C (English) are shown in Table 16.

Rank	User	Accuracy
1	emma	0.87
2	lizefeng	0.855
3	leon14138	0.805
4	**ShubhamKumarNigam**	**0.79**

Table 16: Scores and leader-board ranks for subtask C (English).

5.6 Subtask C (Arabic)

5.6.1 Datasets

Original: The datasets were generated in the same way as in subtask C (Engligh) and are shown in Table 17

5.6.2 Approaches

The approached used here are similar to subtask C (English) except for the used transformers. Since the data is in the Arabic language, we took a couple of models from the Hugging Face library like the

Dataset	Total
Original-Train	521
Original-Validation	224
Original-Embedding	1521
Original-Repetition	1521

Table 17: Original datasets for subtask C (Arabic).

multilingual model mBERT base (cased), trained on the Question Answering (QA) dataset in seven languages and fine-tuned on the combination of XQuAD (Artetxe et al., 2020) and MLQA (Lewis et al., 2020) datasets. We compared their performance to the CAMeLBERT-Mix model.

5.6.3 Results

Metric: The main metric is the accuracy.

BERT: We used CAMeLBERT-Mix model, because it gave better results among the other models, on the original datasets using 5 epochs, 3e-5 learning rate, and the cross entropy as the loss function. The results are shown in Table 18. We used Type II preprocessing for submitting the results as it is the best performing.

Dataset	Type	Val	Test
Original-Train	I	0.6242	0.5
	II	0.6242	0.5
	III	0.6711	0.72
	IV	0.3758	0.5
Original-Embedding	I	0.8792	0.825
	II	0.8792	0.845
	III	0.8691	0.845
	IV	0.7517	0.71
Original-Repetition	I	0.8993	0.855
	II	0.9262	0.87
	III	0.8658	0.86
	IV	0.6409	0.705

Table 18: Accuracy of BERT model for subtask C (Arabic) on original datasets.

The official scores and leader-board ranks of the teams for subtask C (Arabic) is shown in Table 19.

Rank	User	Accuracy
1	lizefeng	0.93
2	AlamiHamza	0.885
3	maryam.najafi	0.875
4	**ShubhamKumarNigam**	**0.87**

Table 19: Scores and leader-board ranks for subtask C (Arabic).

6 Analysis

6.1 Subtask A

English: There are 1400 instances in the test set out of which our model correctly classified 1060 instances (TP=155 and TN=905). There are bigger number of misclassified negative (NS) instances (FP=295) than the number of misclassified positive (S) instances (FN=45) and these are reflected in the precision (34.44) and the recall (77.5). This can be due to the fact that our model was trained on B4 dataset (59% S and 41% NS) taking F1-score for the sarcastic class as the metric for evaluation. This made the model focus more on identifying the positive instances sacrificing the considerable number of misclassified negative instances.

We dived into the details to see when the model predicted well and when it could not predict properly. We found that the majority of tweets which have specific punctuation marks like the exclamation mark were classified correctly. Short tweets were not classified properly which can be due to the insufficient information present in the tweets for classification. Interestingly, the existence of emojis highly increased the recall but not the F1 score and this is because sarcastic tweets can be easily recognized from their emojis but this does not apply on non-sarcastic ones. Also, tweets which include opposite emotions tend to be more sarcastic and to give better F1-score. Moreover, we found that tweets which contain misspellings or that need human knowledge to interpret can cause misclassification. Examples of all previous cases are presented in Table 20.

Arabic: There are 1400 instances in the test set out of which our model correctly classified 903 instances (TP=170 and TN=733). There are bigger number of misclassified negative (NS) instances (FP=467) than the number of misclassified positive (S) instances (FN=30) and these are reflected in the precision (26.69) and the recall (85). The number of instances which contain exclamation marks are less compared to English, and short tweets also were not classified properly for the same reason. Tweets that contain emojis were classified poorly and this is because we used Type II preprocessing which converted the emojis to text similar to the preprocessing of the model we used for English "BERTweet-large" and unlike the preprocessing of the model we used for Arabic "CAMeLBERT-Mix".

Example	Prediction
Exclamation Mark	
So you think the vaccine is a bad idea then. Glad you have that PHD in immunology!	TP
So many error codes in R today!	TN
Short Tweets	
Probably Jude mate	FP
Having the worst time on holiday	FN
Opposite Emotions	
Don't you just love Monday mornings, they are even better when its freezing cold and you have an uncooperative child too!	TP
Emojis	
Wow, can't wait to go into ANOTHER lockdown 🙄	TP
it doesn't need a consultation. Just ban it 🙃	FP
Human Knowledge	
Max Verstappen is such a clean driver, he never makes dirty moves when racing.	FN
Misspelling	
Boris has to bring in these restrictions he is dammed if he does and dammed if he doesn't. I live Boris.❤️❤️❤️❤️	FP

Table 20: Examples of the cases where tweets were classified correctly and incorrectly in subtask A (English).

6.2 Subtask B

Since the train and test datasets contain a very small number of instances of the understatement and overstatement tweets; therefore, the model could not identify any of them and the scores of them were zeros as shown in Table 21. For other labels, the model has far higher recall scores than precision scores which means the model performed well at identifying instances that belong to particular labels at the cost of mislabeling many instances.

	Precision	Recall	F1-score	Support
sarcasm	0.1335	0.8333	0.2301	180
irony	0.0708	0.75	0.1293	20
satire	0.0345	0.0204	0.0256	49
under-statement	0	0	0	1
over-statement	0	0	0	10
rhetorical_question	0.061	0.4545	0.1075	11

Table 21: Performance analysis for subtask B.

6.3 Subtask C

The model performed well in this subtask for both English and Arabic as shown in Table 22. This can be attributed to the nature of the task where the sarcastic tweet and its non-sarcastic rephrase of same meaning are given, besides the ability of the model to extract the relevant information for classification.

	Precision	Recall	F1-score	Support
English				
non_sarcastic	0.7576	0.8065	0.7812	93
sarcastic	0.8218	0.7757	0.7981	107
Arabic				
non_sarcastic	0.8936	0.84	0.866	100
sarcastic	0.8491	0.9	0.8738	100

Table 22: Performance analysis for subtask C.

7 Conclusion

The paper describes our participation in SemEval-2022 Task 6. The models used for sarcasm detection were mainly stand-alone transformers. In addition to this, we ran other experiments by stacking a BiLSTM layer with or without attention mechanism on top of the transformers. We created new datasets for each subtask by augmenting with external datasets, word embedding, or repetition. Our results shows that the augmented datasets enhanced the results for most subtasks. Moreover, we found that the fine-tuned stand-alone transformers gave the best results especially with Type II preprocessing. We also showed the enhancement when using a weighted loss function and the effect of using different learning-rates, epochs, and preprocessing types. We gave analysis of the performance of the models for each subtask, and revealed the possible cases that might have enhanced or deteriorated the performance. Finally, the rank of our team is consistent across most of the subtasks (the fourth rank) which shows the robustness of the used techniques.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ibrahim Abu Farha, Wajdi Zaghouani, and Walid Magdy. 2021. Overview of the WANLP 2021 shared task on sarcasm and sentiment detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 296–305, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. Arabert: Transformer-based model for Arabic language understanding. In *LREC 2020 Workshop Language Resources and Evaluation Conference 11–16 May 2020*, page 9.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

L Bing. 2012. Sentiment analysis and opinion mining (synthesis lectures on human language technologies). *University of Illinois: Chicago, IL, USA*.

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

Bilal Ghanem, Jihen Karoui, Farah Benamara, Véronique Moriceau, and Paolo Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in Arabic tweets. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 10–13.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 1–11, Online. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Veronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in Arabic pre-trained language models. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, Kyiv, Ukraine (Online). Association for Computational Linguistics.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics.

Diana Maynard and Mark Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

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

Silviu Oprea and Walid Magdy. 2020. iSarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1279–1289, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. SemEval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendix

A.1 Hierarchical Architecture

Figure 1 shows a hierarchical network based on a transformer. The input tokens are passed to the transformer, then the output token embeddings are passed to a Bi-LSTM layer which can be with or without attention mechanism.

A.2 Sarcasm Types Description

1. **Sarcasm:** tweets that contradict the state of affairs and are critical towards an addressee.

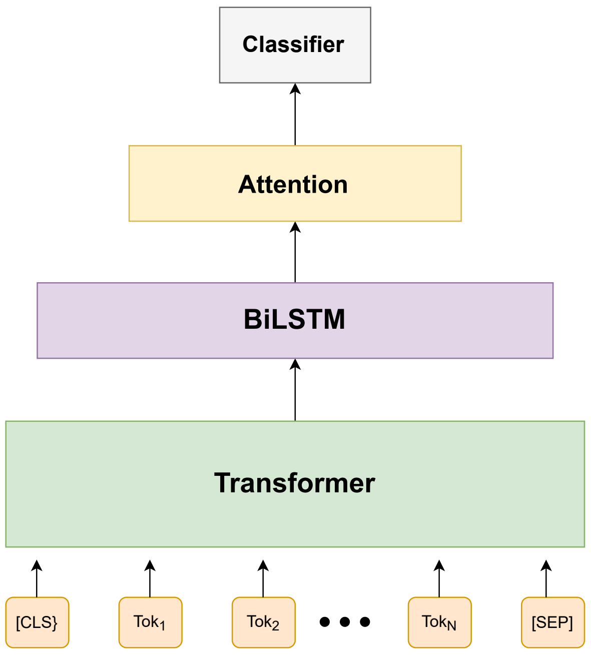


Figure 1: Hierarchical architecture.

2. **Irony:** tweets that contradict the state of affairs but are not obviously critical towards an addressee.

3. **Satire:** tweets that appear to support an addressee, but contain underlying disagreement and mocking.

4. **Understatement:** tweets that undermine the importance of the state of affairs they refer to.

5. **Overstatement:** tweets that describe the state of affairs in obviously exaggerated terms.

6. **Rhetorical question:** tweets that include a question whose invited inference (implicature) is obviously contradicting the state of affairs.

A.3 BERT Classification Architecture

Figure 2 shows the BERT-base classification architecture[15]. From the output of the final (12th) transformer, only the first embedding (corresponding to the [CLS] token) is used by a classifier.

A.4 ELECTRA:- Replaced Token Detection

ELECTRA (**E**fficiently **Le**arning an **E**ncoder that **C**lassifies **T**oken **R**eplacement **A**ccurately) (Clark et al., 2020) replaces the MLM of BERT with Replaced Token Detection (RTD), which looks to be more efficient and produces better results. In BERT, the input is replaced by some tokens with [MASK] and then a model is trained to reconstruct the original tokens.

[15]BERT Fine-Tuning Tutorial with PyTorch by Chris McCormick and Nick Ryan

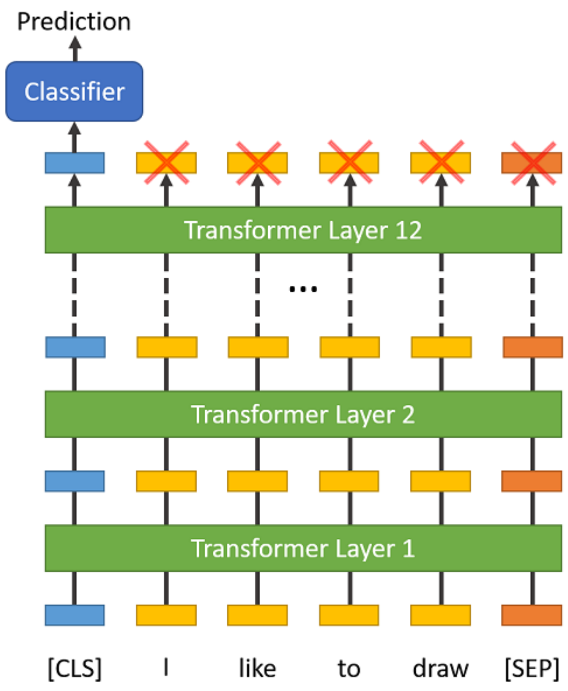


Figure 2: BERT classification architecture.

In ELECTRA, instead of masking the input, the adopted approach corrupts it by replacing some input tokens with plausible alternatives sampled from a small generator network. Then, instead of training a model that predicts the original tokens, a discriminative model is trained that predicts whether each token in the corrupted input was replaced by a generator sample or not.

This approach trains two neural networks, a generator and a discriminator. Each one primarily consists of an encoder (e.g., a transformer network) that maps a sequence of input tokens into a sequence of contextualized vector representations. The discriminator then predicts whether it's fake by analyzing its data distribution.

A.5 Effect of Learning Rates and Epochs in Subtask A (English)

In Subtask A (English), we fine-tuned the BERT model on the B4 dataset with different learning rates and epochs as shown in Table 23 and Table 24 respectively. As shown in tables the best learning rate was 4e-6 and the best number of epochs was 5.

A.6 Performance of BERT+BiLSTM with and without Attention in Subtask A (English)

BERT+BiLSTM: The best results we got using this architecture were on the B4 dataset with 4e-

Val	Test	Learning Rate
0.4002	0.25	1 e - 5
0.5871	0.4558	9 e - 6
0.5913	0.4639	8 e - 6
0.4002	0.25	7 e - 6
0.4002	0.25	6 e - 6
0.5952	0.4883	5 e - 6
0.6025	0.4769	4 e - 6
0.5798	0.475	3 e - 6
0.5644	0.4724	2 e - 6
0.6012	0.5017	1 e - 6
0.4002	0.25	9 e - 7
0.4002	0.25	8 e - 7

Table 23: The effect of learning rates on the performance of the BERT model on the B4 dataset with weighted loss function using F1-score.

Val	Test	Epochs
0.5282	0.5304	1
0.5877	0.5092	3
0.6025	0.4769	5
0.6019	0.4647	7
0.5856	0.457	10
0.6017	0.5099	13
0.594	0.475	15
0.575	0.4943	17
0.5882	0.4675	20
0.5938	0.4633	23
0.575	0.4926	25

Table 24: The effect of epochs on performance of the BERT model on the B4 dataset with weighted loss function using F1-score.

6 learning rate, 10 epochs, and 50 LSTM hidden state size. Table 25 shows the results using same hyperparameters but with different hidden state sizes.

BERT+BiLSTM+Attention: The best results we got using this architecture were on the B3 dataset with 4e-6 learning rate, 5 epochs, and 600 LSTM hidden state size. Table 26 shows the results using same hyperparameters but with different hidden state sizes.

A.7 More Information about the Datasets:

Total number of tweets and percentage of sarcastic and non-sarcastic tweets in each dataset for subtask A is shown in Table 27, and the total number of tweets and percentage of sarcastic labels in each dataset for subtask B is shown in Table 28.

External datasets for subtask A (English) is shown in Table 29, for subtask A (Arabic) is shown in Table 30, and for subtask B (English) is shown in Table 31.

Density of the number of words in tweets and their rephrases in the original datasets is shown in Figure 3 for English and in Figure 4 for Arabic.

In addition to this, information about dialects for Arabic subtasks is presented in Figure 5 and Table 32.

Information about the distribution of sarcastic and non-sarcastic tweets in the original datasets is presented in Figure 6.

Information about sarcastic labels, for subTask B, in the original datasets is shown in Figure 7.

Hidden State Size	Val	Test
50	0.6027	0.4741
100	0.5882	0.4765
300	0.5813	0.4627
600	0.561	0.4702
900	0.5831	0.4516

Table 25: F1-score of the BERT+BiLSTM model for SubTask A (English) on the B4 dataset.

Hidden State Size	Val	Test
50	0.5726	0.4685
100	0.5978	0.468
300	0.5935	0.4627
600	0.6087	0.4625
900	0.5777	0.4618

Table 26: F1-score of the BERT+BiLSTM+Attention model for SubTask A (English) on B3 dataset.

Dataset	Split	Lang	Total	S%	NS%
Original	Train	En	3468	25	75
Original	Train	Ar	3102	24	76
Original	Test	En	1400	14.3	85.7
Original	Test	Ar	1400	14.3	85.7
Twitter API	Train	En	2841	16.8	83.2
Twitter API	Test	En	713	16.8	83.2
SemEval 2018	Train	En	3834	49.8	50.2
ArSarcasm-v2	Train	Ar	12548	17.3	82.7
ArSarcasm-v2	Test	Ar	3000	27.4	72.6

Table 27: Total number of tweets and percentage of sarcastic and non-sarcastic tweets in each dataset for subtask A.

Dataset	Split	Total	Sarcasm% Under-statement%	Irony% Over-statement%	Satire% Rhetorical question%
Original	Train	867	68.3	14.8	2.4
			1	3.8	9.7
	Test	1400	66.4	7.4	18.1
			0.4	3.7	4
Twitter API	Train	477	41.5	30.6	10.9
			1	8.2	7.8
	Test	120	41.7	33.3	11.7
			3.3	6.7	3.3

Table 28: Total number of tweets and percentage of sarcastic labels in each dataset for subtask B.

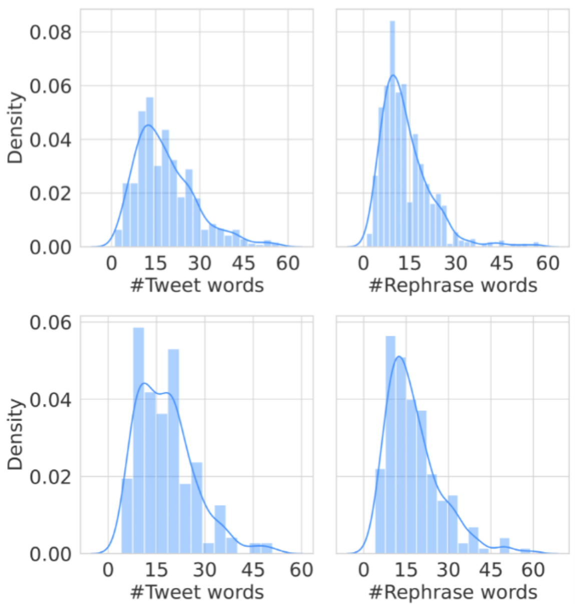


Figure 3: Density of the number of words in the original English train (top) and test (bottom) datasets.

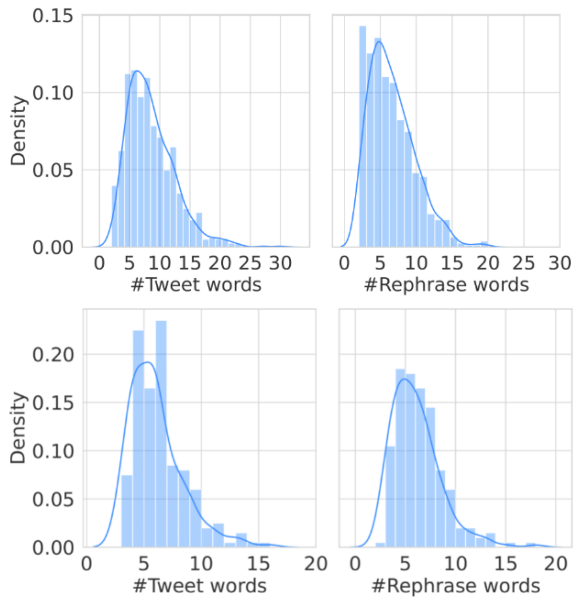


Figure 4: Density of the number of words in the original Arabic train (top) and test (bottom) datasets.

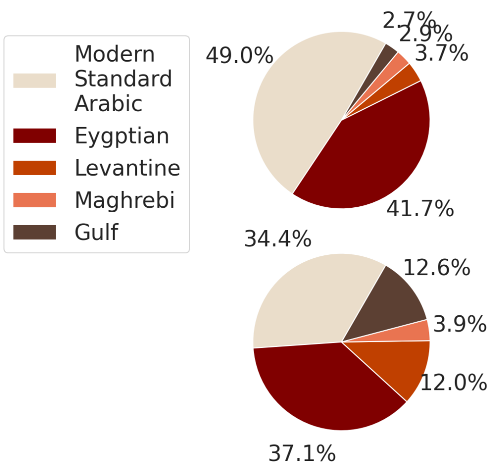


Figure 5: Percentage of dialects of tweets in the original Arabic train (top) and test (bottom) datasets.

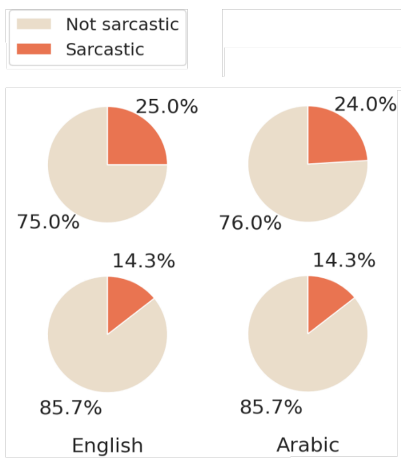


Figure 6: Percentage of sarcastic and non-sarcastic tweets in the original English and Arabic train (top) and test (bottom) datasets.

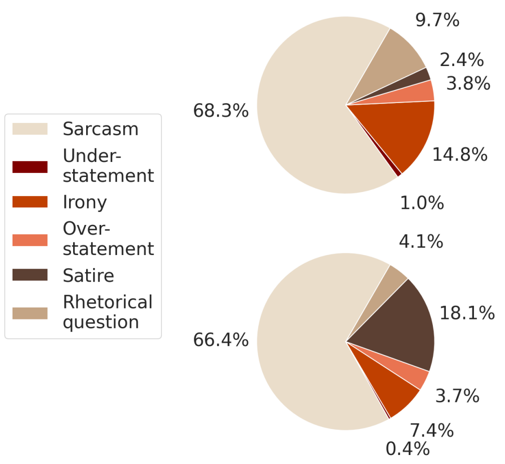


Figure 7: Percentage of tweets under each sarcastic label in the original English train (top) and test (bottom) datasets.

Dataset	Contributing Datasets	Additional Non Sarcastic Tweets from SemEval 2018-Train	Total	S%	NS%
B0	Original Train + Twitter API Train (only sarcastic) + Twitter API Test (only sarcastic) + SemEval 2018 Train (1911 sarcastic)	0	4578	66	34
B1		145	4723	64	36
B2		290	4868	62	38
B3		435	5013	60	40
B4		580	5158	59	41
B5		725	5303	57	43
B6		870	5448	55	45
B7		1015	5593	54	46
B8		1160	5738	53	47
B9		1305	5883	51	49

Table 29: External datasets for subtask A (English).

Dataset	Contributing Datasets	Additional Non Sarcastic Tweets from ArSarcasm-v2 Train	Total	S%	NS%
B0	Original Train + ArSarcasm-v2 Test (only sarcastic) + ArSarcasm-v2 Train (2168 sarcastic)	0	4850	71	29
B1		202	5052	68	32
B2		404	5254	65	35
B3		606	5456	63	37
B4		808	5658	61	39
B5		1010	5860	59	41
B6		1212	6062	57	43
B7		1414	6264	55	45
B8		1616	6466	53	47
B9		1818	6668	52	48

Table 30: External datasets for subtask A (Arabic).

Dataset	Balanced	Contributing Datasets	Total	Sarcasm Irony Satire	Under-statement Over-statement Rhetorical question
Ext-NB	Not Balanced	Original Train + Twitter API Train + Twitter API Test	1203	55.90%	1.20%
				22.30%	5.50%
				6.40%	8.70%
Ext-UW	Using Word Embedding		4336	16.50%	16.60%
				16.50%	16.80%
				16.60%	17%
Ext-UR	Using Repetition		4336	16.50%	16.60%
				16.50%	16.80%
				16.60%	16.90%
Ext-EB	Not Balanced		5314	15%	18.80%
				15.80%	17%
				16.70%	16.80%

Table 31: External datasets for subtask B (English).

Dataset	Split	Total	MSA	Eygptian	Levantine	Maghrebi	Gulf
Original	Train	3102	49	41.7	3.7	2.9	2.7
	Test	1400	34.4	37.1	12	3.9	12.6
ArSarcasm-v2	Train	12548	68.2	21.2	5	0.3	5.1
	Test	3000	77.4	10.2	1.6	0.1	10.7

Table 32: Distribution of tweets over dialects in Arabic Datasets.

IISERB Brains at SemEval-2022 Task 6: A Deep-learning Framework to Identify Intended Sarcasm in English

Tanuj Singh Shekhawat[*1], **Manoj Kumar**[*2], **Udaybhan Rathore**[*3], **Aditya Joshi**[4], **Jasabanta Patro**[5]

[1235] Indian Institute of Science Education and Research, Bhopal
[4] SEEK, Australia
{[1]tanuj19, [2]manoj19, [3]udaybhan19, [5]jpatro}@iiserb.ac.in, [4]ajoshi@seek.com.au

Abstract

This paper describes the system and the resulted models submitted by our team "IISERB Brains" to SemEval-2022 Task 6 competition. We participated in the three sub-tasks for English language datasets. Our submitted models use BERT-based classifiers along with data augmentation. We ranked **19th** out of **43 teams** for sub-task A, **8th rank** out of **22 teams** for sub-task B, and **13th rank** out of 16 teams for sub-task C. In this paper, we describe details of our submissions, related evaluation and additional experiments conducted post the termination of the shared task. Our code and used additional resources are present in GitHub[1] for reproducibility. *Authors with equal contributions are marked by *.*

1 Introduction

Sarcasm in spoken or written form is a type of verbal irony that indicates the difference between the literal and intended meanings of an utterance (Joshi et al., 2017). While many people use it as a bitter remark to mock or ridicule a target(Patro et al., 2019), some also use it as a joke to amuse others(Joshi et al., 2015a). Sarcasm often used together or interchangeably with other ironic categories, is considered an essential component of human communication. A large portion of the web and social media text is sarcastic, which creates a challenge for traditional natural language processing (NLP) tasks like sentiment classification, opinion mining, harassment detection, author profiling etc. Since these systems are deployed widely across various industries, administration, data analysts etc, designing a robust sarcasm detecting component would help the downstream tasks substantially. The SemEval-2022 task 6(Abu Farha et al., 2022) identifies some of the challenges persisting till now, particularly in English and Arabic texts. Our team participated in the tasks floated for the English language under the name "IISERB Brains". The organizers have floated three sub-tasks: (i)Sub-task A to detect whether a given text is sarcastic, (ii)Sub-task B to identify which ironic category the sarcastic tweet belongs to, and (iii) Sub-task C to given two pieces of text, identify the sarcastic one.

In this paper, we report the details of our participating systems and their performance on the evaluation data. The paper also contains details of additional experiments that were a part of our participation in the shared task. The paper is organised as follows. Section 2 describes past work related to the sub-tasks, while section 3 presents the dataset and task details. Following that, Section 4 reports the details of our systems followed by experiments and results in Section 5. Finally, in section 6, we conclude the paper with a summary of the overall results.

With our submitted systems, we obtained the **19th rank** out of **43 teams** for sub-task A, **8th rank** out of **22 teams** for sub-task B, and **13th rank** out of 16 teams for sub-task C. Our code is made public to ensure the reproducibility of our results.

2 Related Work

Sarcasm has been an interesting research topic for computational linguists for a long time(Joshi et al., 2017; Pilling et al., 2017). Researchers have studied the topic from depth(sarcasm sub-categories, linguistic nuances etc.) and breadth (multi-modal sarcasm(Castro et al., 2019a), multi-lingual sarcasm (Liu et al., 2021; Bansal et al.), sarcasm targets (Patro et al., 2019), etc). A considerable amount of work has been reported on the task 'sarcasm detection' itself. This task particularly deals with identifying whether a given text is sarcastic. The sub-task A proposed by the organizers also belongs to this task.

[1]https://github.com/manojmahan/iSarcasmEval-Intended-Sarcasm-Detection-In-English-main

While the traditional approaches have used feature-based machine learning models, recent approaches mostly relied on deep-neural models to report state of art performance. Lexical features like emoticons, special characters, word-patterns have always been preferred features along with semantic features like parts-of-speech tags, contrasting sentiments etc(Davidov et al., 2010)(Veale and Hao, 2010) (Riloff et al., 2013)(Joshi et al., 2015b)(Ghosh et al., 2018) in traditional approaches. Deep-neural methods, on the other hand, learn latent features for the same task. They rely on architectures like RNNs, CNNs, transformers etc. In their models (Joshi et al., 2017; Pilling et al., 2017; Tay et al., 2018; Tarunesh et al., 2021; Jaiswal, 2020). Pre-trained language models like BERT, RoBERTa, XLNet etc, have been extensively used as token-encoders in these models.

Identifying sarcastic intention has always been a challenging task, even humans sometimes have difficulties. Recently, researchers have started focusing on contextual information such as author context (Ghosh et al., 2020), multi-modal context (Ghosh et al., 2020), eye-tracking information (Govindan et al., 2018), or conversation context (Ghosh et al., 2020; Srivastava et al., 2020) to capture it. The sub-task A is related to shared tasks in the domain of figurative analysis such as a SemEval task on irony detection in Twitter that focuses on the utterances in isolation.

3 Background

3.1 Dataset details

We used the English dataset provided by the task organizers(Abu Farha et al., 2022). It has 3468 English tweets. It has 867 sarcastic and 2601 non-sarcastic tweets, which indicates that the dataset is highly unbalanced. For each sarcastic tweet, the organizers have also provided the ironic sub-classes to which the tweet belongs. The sub-classes are sarcasm, irony, satire, understatement, overstatement, and rhetorical question. Also, there are many tweets with multiple labels assigned to them. The distribution of tweets over their labels are shown in Table 1.

Several sarcasm detection datasets are often annotated by a person who is not the author of a piece of text. However, in this dataset, the sarcastic label of each tweet is marked by its author. This makes the dataset unique in terms of capturing the sarcastic intention of the tweet-author(s). Additionally, for each sarcastic tweet, the organizers have asked the authors to rephrase the tweet-text to convey the same message without using sarcasm. However, the organizers have relied on linguistic experts to annotate the sub-categories. Experts referred Leggitt and Gibbs (2000) for the definitions of sub-categories.

Labels	**|N|**
Non-sarcastic	2601
Only-sarcasm	568
Only-irony	122
Sarcasm and irony	1
Sarcasm and satire	21
Sarcasm and overstatement	31
Sarcasm and understatement	6
Sarcasm and rhetorical questions	86
Irony and satire	4
Irony and overstatement	9
Irony and understatement	4
Irony and rhetorical question	15
Understatement and rhetorical question	2
Irony, understatement and rhetorical question	1
Sarcasm, understatement and rhetorical question	1
Total	***3468***

Table 1: Label-wise distribution of tweets. |N| refers to the sample size

3.2 Task details:

Based on the dataset they have released, the task organizers have formulated three challenges as sub-tasks. The details of the sub-tasks are,

- **Sub-task A:** Given a text, determine whether it is sarcastic or non-sarcastic;
- **Sub-task B:** This sub-task is designed for particularly English dataset. It is a binary multi-label classification task. Here, given a text, we have to determine which ironic speech category it belongs to, if any;
- **Sub-task C:** Given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which of the two is the sarcastic.

For all of the three sub-tasks, the organizers have informed us that precision, recall, accuracy, and macro-F1 of the participating models will be reported. According to them the main metrics of evaluation for the sub-task A is the F1-score for the sarcastic class. Similarly, for sub-task B and sub-task C, it is the macro-F1 score and the accuracy, respectively.

4 System overview

4.1 Additional resources

As shown in Table 1, the dataset is highly unbalanced, and the sample size is small. To mitigate this issue, we considered additional publicly available datasets published earlier for a similar task with some synthetically generated text. The details of such datasets are following,

- **SemEval-2018 task 3:** We used the training and test data provided by SemEval-2018 task on irony detection (Van Hee et al., 2018) as an additional resource for training of our models.
- **MUStARD:** We used the textual part of the multi-modal sarcasm detection dataset provided by Castro et al. (2019b) as an additional resource for training of our models.
- **FigLang 2020 Sarcasm:** We used the sarcasm dataset[2] released as a part of shared task of FigLang2020 [3] workshop as additional data for training of our models.

  Augmentation: For increasing the instances labeled with sub-categories in the train data provided by the task organizers, we performed data augmentation using the python NLPAUG library[4]. NLPAUG applies a set of transformations to textual datasets in order to create augmented data for deep learning models that rely on high volumes of data. We took sarcastic tweets given by organizers used the word-replacement procedure provided by NLPAUG to synthesize three additional tweets from each input tweet. We used 'distilbert-base-uncased'[5] contextual embedding as the input embedding for this process.

[2]https://github.com/EducationalTestingService/sarcasm/releases

[3]https://sites.google.com/view/figlang2020/home

[4]https://nlpaug.readthedocs.io/en/latest/overview/overview.html

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

Source	|N|
SemEval-2022 Task 6	3468
SemEval-2018 Task 3 training	3398
SemEval-2018 Task 3 test	780
MUStARD	690
FigLang20 Sarcasm	9400
Data Augmentation (867x3)	2601

Table 2: Basic statistics of additional sources. |N| refers to the sample size.

The statistical distribution of additional resources is shown in Table 2. After elimination of the duplicates, the final dataset had 19986 tweets.

4.2 Data preprocessing

We followed the following preprocesing steps for every instance in our dataset.

- **Case conversion:** We converted the dataset into lowercase except for those words in which the whole word is in uppercase.
- **Stop-word removal:** We removed all stop-words as they contain low information. We did this using python NLTK library [6].
- **Data cleaning:** We did basic data cleaning which include removal of links, punctuation marks, floating point(.) characters and user-name. However, we did not apply stemming and lemmatization techniques because we believe they will distort the meaning of instances.
- **Special Tokens:** We added special tokens at the starting and ending of the instances as required by different tokenizers for respective transformer based models.

4.3 Model description

We relied on Transformer-based architectures to design our models for all sub-tasks since Transformers are regarded as state-of-the-art in NLP. We built our models using Hugging Face's Transformer library. They support generic transformer based architectures with the ability to seamlessly initialize the tokens with different pre-trained embeddings.

- **Sub-task A:** For this sub-task, we deployed the binary classifier versions of different transformer based architectures provided by the

[6]https://www.nltk.org/

Hugging Face transformer library. We particularly experimented with BERT(Devlin et al., 2019), RoBERTa(Liu et al., 2019), XLNet(Yang et al., 2019) and DistilBERT(Sanh et al., 2019) architectures. Apart from initializing the tokens with respective pre-trained embeddings, we fine-tuned the last layers of the models according to our training data. Further, we added the non-sarcastic versions of 867 sarcastic tweets given by the organizers to the training set.

- **Sub-task B:** Here, instead of using a multi-label classifier, we used six binary classifier versions of the transformer based architectures provided by the same Hugging Face library. As in the previous sub-task, here we have also experimented with BERT, RoBERTa, XLNet, and DistilBERT architectures. We constructed training data(refer Table 6) for label-wise models to fine-tune it. We merged the predictions of all six models to get the final prediction labels.
- **Sub-task C:** We formulated this sub-task as a parallel combination of two sub-task A models. We considered the same architecture for both parallel sub-components in all of our experiments.

5 Experiments and results

5.1 Sub-task A

For this sub-task, in addition to the data given by the organizers, we considered the other datasets as mentioned in Table 2. We applied the pre-processing(details in section 4.2) steps before sending them to the respective tokenizers of the considered architectures. The tokens are initialized by respective pre-trained embeddings. The dataset is divided into three parts i.e. training, validation and test set with a ratio of 0.7:0.2:0.1 for the parameter and hyper-parameter tuning. We generated the predictions for the unlabeled data provided by the organizers and submitted them in the codalab submission site[7] for evaluation. As the organizers have later released the labels for their evaluation data, we can compare all our models by ourselves too. The performance of our models for the evaluation data is reported in Table 3. The number of epochs for which the models are trained are different for different models. We trained until our models started over-fitting. Note that the best performing result reported in the table is different from that we have submitted in the codalab site. We experimented with our models even after the evaluation period and the results in Table 3 show the best performance we have achieved till date. The models submitted as a part of competition is reported in Table 5. The hyper-parameters for all models are reported in Table 4.

[7]https://codalab.lisn.upsaclay.fr/competitions/1340

5.2 Sub-task B

As stated in the previous section, for this sub-task we considered separate binary classifiers for each label. The sample size of the datasets created for individual classifiers are shown in Table 6. Note that we did not include the non-sarcastic tweets provided by the organizers in our new datasets. Rather, we added the synthetic tweets generated by the nlpaug library (see section 4.1) to amplify the label for which it is created. We didn't generate any additional synthetic text for the dataset corresponds to 'sarcasm' label as it is the dominant class in the provided 'sarcastic' data. Thus, the dataset created for the binary classification of 'sarcasm' class has the original 867 sarcasic tweets. In other label-specific datasets we increased the corresponding label tweets with the help of nlpaug library. As stated in section 4.1, we did this by generating three similar texts for each tweet tagged with the considered label. Thus, the newly created label-specific datasets have different sample size as shown in Table 6. We fine-tuned BERT model for each category of ironic speech. The organizers have evaluated sub-task B based on macro-F1 score. The results are reported in Table 7.

5.3 Sub-task C

As reported in Section 4.3, we formulated the task as two parallel combination of sub-task A models. The considered same model across the parallel sub-components. The accuracy(evaluating measure considered by the organizers) of different models are reported in Table 8. As we can infer, BERT based fine tuned model performed best on the evaluation data among all with an accuracy of **0.62**.

6 Conclusion

In this paper, we discussed our models for different sub-tasks proposed by SemEval 2022 task 6 organizers for the English dataset. We experimented

Model	Precision	Recall	F-1 score	F-1 sarcastic	accuracy
DistilBERT	0.57	0.64	0.53	0.34	0.62
XlNet	0.63	0.61	0.62	0.34	0.82
BERT	0.60	0.69	0.60	0.39	0.70
RoBERTa	***0.70***	0.68	***0.69***	***0.47***	***0.86***

Table 3: Performance measures of our models submitted for sub-task A

Model	Learning-rate	MAX_SEQ_LEN	BATCH_SIZE	EPOCHS
RoBERTa	2e-6	256	16	10
BERT	2e-5	128	32	3
Xlnet	2e-5	128	32	4
DistilBERT	5e-5	1213	16	5

Table 4: Hyper-parameters of our models.

SubTask A	
Model	*F1*
BERT	0.34
SubTask B	
Model	*Macro-F1*
BERT	0.0751
SubTask C	
Model	*Accuracy*
BERT	0.62

Table 5: Performance of our submitted models for the three tasks.

with Transformer architectures with different pre-trained language models in our submitted systems.

With our submitted models, our team "IISERB Brains", obtained the **19th rank** out of **43 teams** on sub-task A, **8th rank** out of **22 teams** on sub-task B and **13th rank** out of 16 teams in sub-task C. All of our code and links to considered data are uploaded in GitHub[8] for reproduciblity.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Srijan Bansal, G Vishal, Ayush Suhane, Jasabanta Patro, and Animesh Mukherjee. Code-switching patterns can be an effective route to improve performance of downstream nlp applications: A case study of humour, sarcasm and hate speech detection.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019a. Towards multimodal sarcasm detection (an _obviously_ perfect paper). *arXiv preprint arXiv:1906.01815*.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019b. Towards multimodal sarcasm detection (an _Obviously_ perfect paper). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4619–4629, Florence, Italy. Association for Computational Linguistics.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Coling 2010: Posters*, pages 241–249.

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

Debanjan Ghosh, Alexander R Fabbri, and Smaranda Muresan. 2018. Sarcasm analysis using conversation context. *Computational Linguistics*, 44(4):755–792.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. *arXiv preprint arXiv:2005.05814*.

Kannan Govindan, TC Edwin Cheng, Nishikant Mishra, and Nagesh Shukla. 2018. Big data analytics and application for logistics and supply chain management.

[8] https://github.com/manojmahan/iSarcasmEval-Intended-Sarcasm-Detection-In-English-main

Source	|S|	|I|	|O|	|Rq|	|St|	|U|
SemEval-2022 Task 6	867	867	867	867	867	867
Data Augmentation	0	465	120	303	75	30
Total	*867*	*1332*	*906*	*1170*	*942*	*98*

Table 6: Dataset sources used for sub-task B. Sample size for categories sarcasm, irony, overstatement, rhetorical question, satire and understatement are represented by |S|, |I|, |O|, |Rq|, |St|, |U| respectively.

Results	BERT
Macro F-1	0.075
F1-Sarcasm	0.23
F1-irony	0.096
F1-satire	0.083
F1-understatement	0.00
F1-overstatement	0.00
F1-rhetorical question	0.0414

Table 7: Macro-F1 and class-wise F1 scores for sub-task B

Model	Accuracy
RoBERTa	0.47
XlNet	0.49
DistilBERT	0.57
BERT	***0.62***

Table 8: Accuracy of our models on evaluation data provided for sub-task C.

Nikhil Jaiswal. 2020. Neural sarcasm detection using conversation context. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 77–82.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5):1–22.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015a. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.

Ankur Joshi, Saket Kale, Satish Chandel, and D Kumar Pal. 2015b. Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396.

John S Leggitt and Raymond W Gibbs. 2000. Emotional reactions to verbal irony. *Discourse processes*, 29(1):1–24.

Yaochen Liu, Yazhou Zhang, Qiuchi Li, Benyou Wang, and Dawei Song. 2021. What does your smile mean? jointly detecting multi-modal sarcasm and sentiment using quantum probability. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 871–880.

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

Jasabanta Patro, Srijan Bansal, and Animesh Mukherjee. 2019. A deep-learning framework to detect sarcasm targets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6336–6342, Hong Kong, China. Association for Computational Linguistics.

Luke C Pilling, Chia-Ling Kuo, Kamil Sicinski, Jone Tamosauskaite, George A Kuchel, Lorna W Harries, Pamela Herd, Robert Wallace, Luigi Ferrucci, and David Melzer. 2017. Human longevity: 25 genetic loci associated in 389,166 uk biobank participants. *Aging (Albany NY)*, 9(12):2504.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Himani Srivastava, Vaibhav Varshney, Surabhi Kumari, and Saurabh Srivastava. 2020. A novel hierarchical bert architecture for sarcasm detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 93–97.

Ishan Tarunesh, Somak Aditya, and Monojit Choudhury. 2021. Trusting roberta over bert: Insights from checklisting the natural language inference task. *arXiv preprint arXiv:2107.07229*.

Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. *arXiv preprint arXiv:1805.02856*.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Tony Veale and Yanfen Hao. 2010. Detecting ironic intent in creative comparisons. In *ECAI 2010*, pages 765–770. IOS Press.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

connotation_clashers at SemEval-2022 Task 6: The effect of sentiment analysis on sarcasm detection

Patrick Hantsch and **Nadav Chkroun**
University of Tübingen
{patrick.hantsch, nadav.chkroun}@student.uni-tuebingen.de

Abstract

We investigated the influence of contradictory connotations of words or phrases occurring in sarcastic statements, causing those statements to convey the opposite of their literal meaning. Our approach was to perform a sentiment analysis in order to capture potential opposite sentiments within one sentence and use its results as additional information for a further classifier extracting general text features, testing this for a Convolutional Neural Network, as well as for a Support Vector Machine classifier, respectively.

We found that a more complex and sophisticated implementation of the sentiment analysis than just classifying the sentences as positive or negative is necessary, since our implementation showed a worse performance in both approaches than the respective classifier without using any sentiment analysis.

1 Introduction

According to Cambridge dictionary, sarcasm is defined as the use of remarks that mean the opposite of what they say, in order to hurt someone's feelings or to criticize something in a humorous way. While detecting sarcasm can be very difficult even beyond the text level, it is a real challenge to detect sarcasm in textual data. When having conversations in person, it is easier for people to detect sarcasm because they have additional information in form of the speaker's emphasis of words, his facial expressions and body gestures. All of this information is not available in written language. Yet, detecting sarcasm from text becomes more and more important, since everyday life nowadays takes place on social media platforms to a large extent, unfortunately including things like hate speech in order to offend people. Sarcasm is often used as a tool for that (Frenda, 2018), so operators need ways to detect and remove offensive material expressed in a sarcastic way in large amounts of messages and posts released by millions of people every day.

The shared task (described in Abu Farha et al., 2022) consisted of three sub tasks:

- Sub task A: Determining if a given input text is sarcastic or not
- Sub task B: Determining if a given input text belongs to a ironic speech category and if so, to which one exactly
- Sub task C: Given two input texts, one being sarcastic and the other one being its non-sarcastic rephrase, determining which is the sarcastic one

We only participated in sub task A of the shared task. Our strategy was to additionally perform a sentiment analysis on the given text data, using the contradictory nature of a sarcastic statement, between the actual utterance and its true meaning/intent. For example in the sentence "I love how ill i became last night…", a sentiment analysis might recognize the clash of connotations between the positive connotated word "love" and the negative connotated fact of "being ill". Van Hee et al. (2018) also mentioned the inversing effect of using irony (a form of sarcasm) in a sentence on the overall sentiment of that sentence and even included this notion in form of a class called "Verbal irony by means of a polarity contrast" in one of their sub tasks of the task they provided for the SemEval competition in 2018.

In an earlier work described in Poria et al. (2017) a similar approach to our strategy was used, combining several Convolutional Neural Networks (CNNs), pre-trained on recognizing and classifying sentiment, emotion and personality features respectively, with a Support Vector Machine classifier for classification only. However, for the best of our knowledge there was no earlier work investigating the impact of sentiment analysis on sarcasm detection for linear models. Inspired by the previously mentioned work, we combined the result of our sentiment analysis with all other features subtracted from the given text and investigated the influence of sentiment analysis on the final sarcasm detection for both a linear model, as well as for a Deep Learning model, also comparing the performance of both approaches.

2 Training Data

According to sub task A, our system should classify an input text as either sarcastic, or non-sarcastic. If, for example, the string "I love how it rains for the seventh day in a row" was fed as input to our system, the output would be either "1" (sarcastic), or “0” (non-sarcastic). As training data, two collections of tweets were given, one in English and one in Arabic. In table 1 you can see the amount of sarcastic and non-sarcastic tweets in both datasets.

	English	**Arabic**
total amount of tweets	**3468**	**3102**
`sarcastic`	867	745
`non-sarcastic`	2601	2357

Table 1: Amount of sarcastic and non-sarcastic tweets in training data

3 System Overview and Experimental Setup

Both approaches we decided to test for solving the task were implemented in Python 3.8. In terms of sentiment analysis we decided to focus on English, for which we tested each of those approaches both combined with a sentiment analysis and without it. The predictions contained in our official submission were created by our Deep Learning model not being combined with sentiment analysis, for both English and Arabic.

3.1 Official Submission

We chose to build a CNN as a Deep Learning Model for sarcasm detection. CNNs are several layers of convolutions with non-linear activation functions like ReLU (Brownlee, 2019) or tanh applied to the results.

3.1.1 Preprocessing

All tweets were split into a sarcastic and a non-sarcastic tweet collection. We used the tokenizer class from Keras library (Chollet et al., 2015), which allows vectorization of a text corpus, by turning each text into a sequence of integers, while calculating the maximum number of words to keep based on their frequency. In addition, we split the training data into training set (80%) and validation set (20%) for hyperparameter optimization. We were setting each tweet a length, and padding or truncating it, based on the length of 100 words, respectively.

3.1.2 Training

In our model, we converted words to vocabulary indices. We did not use pre-trained embeddings. Instead, the embedding was done by the embedding layer of our model. In total, 6 layers were used, which are described in table 2.

name of layer	**parameters**
embedding	vocab_size=8879, embedding_dim=32, input_length=100
convolution 1D	filters=128, kernel_size=3, activation= “relu”
pooling 1D	-
Dense	units=100 activation=“relu”
Dense	units=32 activation=“relu”
Dense	units=1 activation=“sigmoid”

Table 2: Description of the model’s architecture

We compiled the model using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.01. The loss function for the optimizer was Binary Cross Entropy, which compares each of the predicted probabilities to the actual class labels which can be either 0 or 1. For the tuning, random search was used to find the optimal hyperparameters. This method sets up a grid of hyperparameter values and selects random combinations. We tuned the parameters with the respective values as shown in table 3.

In total, 5 different hyperparameter settings were tried by our random search.

3.1.3 Evaluation

According to the official metrics, which was F1-score for the sarcastic class, our CNN trained on the English data scored 0.2024, unweighted mean precision being 0.5016 and unweighted mean recall being 0.5029. All in all, our CNN achieved a

Description	Name	Values
learning rate of the optimization algorithm	learning_rate	0.01, 0.001, 0.0001
length of the convolution window	kernel_size	3, 5
size of the dense layer	units	32-128 (interval: 16)
number of convolution units	filters	32-256 (interval: 32)

Table 3: Hyperparameters and values to which the random search was applied

bad rank (37/43). The CNN trained on the Arabic training data achieved a F1-score of 0.3013, a unweighted mean precision score of 0.5647 and the unweighted mean recall was 0.5954. It ranked 23rd of 32 total submissions.

3.2 CNN Influenced by Sentiment Analysis

The preprocessing used for the CNN we combined with the sentiment analysis is the same as for the CNN which created our official submission.

3.2.1 Sentiment Analysis & Transfer Learning

We implemented sentiment analysis for sarcasm detection, used as a "transfer learning" method. The sentiment analysis assigns each tweet a label of either being "positive" or "negative". The sentiment analysis dataset we used was the IMDB movie review sentiment classification dataset of Keras. It consists of 25000 movie reviews, labeled as 1 ("positive") or 0 ("negative"). We split it into a training set consisting of 20000 of the reviews and a test set containing 5000 reviews.

First, we trained our CNN model on the training set for sentiment analysis. We saved the weights and then trained the model on the sarcasm detection training set that we created previously. It is important to note that the sentiment analysis scores were used in a way that although the model labeled each sentence as a float number between 0 to 1. The final labels (0 or 1) were given to each sentence according to its proximity to 0.5.

3.2.2 Training

In order to get a more fine-tuned model for the combination with the sentiment analysis, we changed the model's architecture. The reason for this change is a result that another random search, which we conducted after we applied the transfer learning, picked for our combined model. The same hyperparameters and values were tested as in our first random search described in section 3.1.2. We changed the hyperparameters accordingly, leading to the architecture depicted in table 4.

name of layer	parameters
embedding	vocab_size=88584, embedding_dim=32, input_length=100
convolution 1D	filters=128, kernel_size=3, activation="relu" kernel_regularizer=l2(0.001)
dropout	rate=0.1
convolution 1D	filters=128, kernel_size=3, activation="relu" kernel_regularizer=l2(0.001)
dropout	rate=0.1
pooling 1D	-
dropout	rate=0.1
Dense	units=32 activation="relu" kernel_regularizer=l2(0.001)
dropout	rate=0.1
Dense	units=1 activation="sigmoid" kernel_regularizer=l2(0.001)

Table 4: Description of the model's architecture for the transfer learning

3.2.3 Evaluation

Our CNN which was influenced by the sentiment analysis performed worse than the CNN on its own. The F1 score on the sarcastic class was 0.1679. Unweighted mean precision was 0.5109, unweighted mean recall 0.5117.

3.3 Comparing Results of both CNN's

Comparing the scores of both the CNNs shows that the CNN augmented by the sentiment analysis performed worse than the CNN on its own, as shown in table 5.

3.4 Linear Model Approach

In the following subsections we discuss our linear model based approach. The preprocessing steps, as

model	f1-score	precision (unweighted mean)	recall (unweighted mean)
CNN without transfer learning of sentiment analysis	0.2024	0.5016	0.5029
CNN including transfer learning of sentiment analysis	0.1679	0.5109	0.5117

Table 5: Both CNN model's scores on the English test set

well as the implementation of sentiment analysis differ from what we did in our Deep Learning based approach.

3.4.1 Preprocessing

One challenge we were confronted with was a huge amount of data imbalance. The English data consisted of only 25% sarcastic tweets, the Arabic data also of around 24%, while the rest of the data was non-sarcastic, respectively. To account for this in our linear model approach, in each language respectively, first, all tweets were split into a sarcastic and a non-sarcastic tweet collection. After shuffling both collections, we chose 867 tweets each, i.e all sarcastic tweets were used, for the further training process. We put 694 tweets each (80%) into the training set, while keeping the other 173 tweets (20%) of each category as held-out data for testing. This way, we had a guaranteed sarcastic/ non-sarcastic ratio of 50% each to avoid bias due to the previously mentioned imbalance. After shuffling both the training set and the test set again, we obtained our final datasets.

3.4.2 Feature Extraction

We extracted the features from our training data by using a simple count vectorizer, while applying tf-idf weighting, both as implemented in scikit-learn (Pedregosa et al., 2011). The vectorizer considered word n-grams from unigrams up to 4-grams. This feature extraction process was used in the grid search we performed for choosing a linear classifier as described in section 3.4.3, as well as for training both with and without sentiment analysis, described in section 3.4.5.

3.4.3 Model Choice

For the linear model approach we chose three different linear models and performed a grid search on them, trying to optimize F1-score for the sarcastic class on a held-out dataset (20% of all training data, see Section 3.4.1), to find the best hyperparameter settings for fitting on our training data. The three models chosen were a Naive Bayes Classifier, a Support Vector Machine model and a Random Forest classifier. The following hyperparameters of the respective models were included in the grid search:

- For Naive Bayes Classifier:

Description	Name	Values
Parameter for add-k smoothing	alpha	0.1, 1.0, 2.0, 5.0, 10.0, 20.0, 50.0, 100.0

- For Support Vector Machine

Description	Name	Values
Probability estimates for classification	probability	True, False
Kernel type	kernel	linear, poly, rbf, sigmoid
Kernel coefficient	gamma	scale, auto
Degree of the polynomial kernel function	degree	0-6 (interval: 1)

- For Random Forest Classifier:

Description	Name	Values
Number of decision trees in the forest	n_estimators	10, 20, 50, 100, 200, 300, 400, 500
Minimum number of samples needed to split a node (make a decision)	min_samples_split	3, 5, 10, 20, 30, 50
Maximum depth of a tree	max_depth	None, 3, 5, 15, 25, 50, 100
Maximum number of features considered when looking for each split	max_features	3, 5, 10, 20

The F1-score of the sarcastic class was calculated using cross-validation. Features of the used training data during the grid search were extracted as described in section 3.4.2. Both the grid search and the classifiers were implemented via scikit-learn. We chose the classifier which achieved the highest F1-score for the sarcastic class with its best hyperparameter settings respectively, obtained from the previously mentioned grid search. This classifier was the Naive Bayes Classifier. However, after making predictions on the organizer‘s test data, we decided to finally use the Support Vector Machine model, since almost all test instances were classified sarcastic by the Naive Bayes Classifier, while the predictions made by the Support Vector Machine model were mixed and seemed more reasonable. The best hyperparameter setting found for the Support Vector Machine classifier were its default values for all possible hyperparameters, as defined in scikit-learn.

3.4.4 Sentiment Analysis

To implement the sentiment analysis for English we used a pre-trained model from the Flair NLP library (Akbik et al., 2019). Flair is a NLP framework providing a lot of different models for several common NLP tasks, including sentiment analysis. We can feed an input text to the pre-trained Flair model to get a classification for the text being positive or negative, as well as a score indicating how confident the model was about the classification between 0.5 and 1.0.

Our idea for how to use this result to help us solve the task at hand was to feed each input sentence into the Flair model and obtain a sentiment prediction, "positive" or "negative", together with the model's confidence score. We then assign one of six categories to the respective sentence. We created three categories, each resembling a certain range of the model's confidence score for both the positive and negative class. Thus, we obtained the following six categories:

confidence score	**classified "positive"**	**classified "negative"**
> 0.95	very positive	very negative
0.75 - 0.95	quite positive	quite negative
< 0.75	rather positive	rather negative

Depending on the sentence's category we then created a one-hot encoded vector. For example, if a sentence would be categorized as very positive, the respective vector would be "[1 0 0 0 0 0]", with each integer resembling one of the categories and its value showing if it is the sentence's category (1) or not (0).

Applying this to each input sentence results in a matrix, which we attached in the training process (see section 3.4.5) to the feature matrix that was created by our feature extraction described in section 3.4.2.

3.4.5 Training

For training in both cases (with and without sentiment analysis), we preprocessed our training data as described in section 3.4.1 and obtained our feature matrix as described in section 3.4.2.

For training with sentiment analysis, we attached the sentiment matrix, obtained in the process described in section 3.4.4, horizontally. That means the first row of the sentiment matrix was appended to the first row of the feature matrix etc. such that the final matrix resembles all input sentences of the training set (rows) with all extracted features, including the result of our sentiment analysis (columns).

To get our final predictions we fit our Support Vector Machine classifier on our respective feature matrix, depending on training with or without sentiment analysis.

3.4.6 Results

The linear model trained without the input of our sentiment analysis scored 0.2738 according to the official metrics, which was F1-score for the sarcastic class. The model trained with the input of the sentiment analysis scored only 0.2721, so the model influenced by the sentiment analysis performed slightly worse.

model	**f1-score**	**precision (unweighted mean)**	**recall (unweighted mean)**
SVM with sentiment analysis	0.2721	0.5318	0.564
SVM without sentiment analysis	0.2738	0.5336	0.5673

Table 6: Scores of the linear model with and without sentiment analysis for the English test set

4 Conclusions

The Support Vector Machine model (SVM) performed better than the CNN both when sentiment analysis was included, as well as on its own. The SVM without sentiment analysis also showed the highest F1 score of all four tested models. However, in general our results show that our ways of implementing sentiment analysis even had a negative impact on the classification for sarcasm on both systems.

One reason for the bad performance of the sentiment analysis could have been that we performed the sentiment analysis on the input sentences as a whole for both training and classifying. In order to recognize a contradiction within one sentence, as explained in Section 1, improvements might be achieved if the sentence is split in several sub parts and sentiment analysis is performed on those parts. This way, positive and negative classifications with high confidence scores might be observed, clearly indicating the aforementioned clash of connotations. Potential challenges that could come up in that case are to determine where to split the sentences exactly, or deciding which parts of even the sub parts the sentiment analysis should be performed on, since not all word types convey sentiment (e.g. stop words could be considered neutral). To account for this, alternatively to our approaches, a sentiment lexicon could be used to calculate scores on sub parts of sentences.

Even without splitting the input sentences, we could have implemented our idea better for both approaches. For example, for the Deep Learning approach, sentiment analysis datasets might be more useful, if their genre is the same or closer to our training data. An additional way to adjust our implementation for this approach would be to shift our decision boundary, classifying sentences contradictory, if output values of the sigmoid function between for example 0.35 and 0.65 can be observed, since clearly positive and negative sub parts might lead to a rather neutral sentiment in total.

Similarly, for the linear model approach we could have weighted the "rather positive" and the "rather negative" category as more important, compared to the more extreme categories.

Acknowledgements

We would like to thank Çağrı Çöltekin for his advice and support while creating this work. We would also like to thank everyone who reviewed an earlier version of this paper and provided helpful comments and suggestions for improvement.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art in natural language processing. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Jason Brownlee. 2019. A gentle introduction to the rectified linear unit (relu). `https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/`.

François Chollet et al. 2015. Keras. `https://keras.io`.

Simona Frenda. 2018. The role of sarcasm in hate speech. pages 14–16.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2017. A deeper look into sarcastic tweets using deep convolutional neural networks.

Cambridge University Press. 2022. Cambridge dictionary.

TUG-CIC at SemEval-2021 Task 6: Two-stage Fine-tuning for Intended Sarcasm Detection

Jason Angel
CIC, Instituto Politécnico Nacional
Mexico City, Mexico
ajason08@gmail.com

Segun Taofeek Aroyehun
TU Graz
Graz, Austria
aroyehun.segun@gmail.com

Alexander Gelbukh
CIC, Instituto Politécnico Nacional
Mexico City, Mexico
www.gelbukh.com

Abstract

We present our systems and findings for the iSarcasmEval: Intended Sarcasm Detection In English and Arabic at SEMEVAL 2022. Specifically we take part in the Subtask A for the English language. The task aims to determine whether a text from social media (a tweet) is sarcastic or not. We model the problem using knowledge sources, a pre-trained language model on sentiment/emotion data and a dataset focused on intended sarcasm. Our submission ranked third place among 43 teams. In addition, we show a brief error analysis of our best model to investigate challenging examples for detecting sarcasm.

1 Introduction

According to the Freedictionary[1], sarcasm is a cutting statement to express contempt or ridicule, often using words to convey a meaning that is the opposite of their literal or actual meaning.

Due to its ambiguous nature, sarcasm plays an important role for resolution of several NLP tasks, such as Sentiment Analysis (Liu et al., 2010; Joshi et al., 2017; Maynard and Greenwood, 2014), Hate speech (Frenda et al., 2022), disagreement classification (Ghosh et al., 2021), Opinion Mining (Kannangara, 2018) among others. One example of benefits of modeling sarcasm is presented by Bouazizi and Ohtsuki (2015) about the use of sarcastic tweets to improve Sentiment Analysis.

In this work we present our submission to iSarcasmEval at Semeval-2022 for subtask A (Given a text, determine whether it is sarcastic or non-sarcastic). Our solution system is at the top three teams among 43 teams. The proposed approach uses the pre-trained language model well informed of this task, because it uses sentiment/emotion data and we enhanced it with a dataset about intended sarcasm texts.

The rest of the document goes as follows: Section 2 overviews some related works about sarcasm detection tasks. In Section 3, we give a detailed description of our conducted experiments, including the dataset used. Section 4 summarizes our results and offers an interpretation to our findings. Finally, Section 5 presents our conclusion, contributions and future work.

2 Related works

Beside the importance of detecting sarcasm for industry applications related to understanding comments on social networks or commercial products reviews (Yavanoglu et al., 2018), sarcasm detection has received great attention from the NLP community, influencing the creation of diverse approaches, benchmark datasets and evaluation methods.

The methods for modeling sarcasm range from rule-based system and statistical approaches by exploiting handcrafted features using traditional machine learning (Joshi et al., 2015; Wicana et al., 2017) to modern techniques using deep learning architectures together with word embeddings and pretrained language models (Mehndiratta and Soni, 2019; Srivastava et al., 2020; Wang et al., 2021).

Previous attempts for benchmarking sarcasm detection were proposed by "SemEval-2018 Task 3: Irony Detection"[2], ALTA Shared Task 2019[3], FigLang 2020[4] and WANLP 2021[5]. They crafted their datasets either using weak supervision tech-

[1] https://www.thefreedictionary.com/

[2] https://competitions.codalab.org/competitions/17468

[3] http://www.alta.asn.au/events/sharedtask2019/description.html

[4] https://sites.google.com/view/figlang2020/shared-tasks

[5] https://sites.google.com/view/wanlp2021

niques like scraping tweets having the #irony, #sarcasm hashtags, or by manual labeling from third party annotators, which leads to several shortcomings as exposed in Oprea and Magdy (2020), instead the dataset used in this work (iSarcasm) was crafted by asking the authors themselves to provide the sarcastic/non-sarcastic labeling for their tweets.

	Training	validation
Sarcastic	867	86
Non-Sarcastic	2601	259
Total examples	3468	345

Table 1: Number of classes in train and validation splits

3 Experiments

Our experiments aim to determine whether a given text is intended to be sarcastic or non-sarcastic. The metric for ranking systems on the competition is the F1-score for the sarcastic class (positive label) only.

Datasets. As part of the iSarcasm shared task (Abu Farha et al., 2022), the organizers provide training and test datasets for English and Arabic, however, we only participated in the English track. Table 1 displays the number of samples and the distribution over sarcastic and non-sarcastic texts for train and validation splits. As observed sarcastic texts roughly account for the 25% of data in each split.

In addition to the iSarcasmEval dataset, we also employed the SPIRS dataset[6] which is a collection of sarcastic and non-sarcastic tweet ids (15,000 for each category) gathered using "reactive supervision", a new data capturing method (Shmueli et al., 2020). From SPIRS only intended and negative examples were considered which amount to 15,950 and 1,773 examples in the training and development splits, respectively. Therefore, tweets perceived as sarcastic were discarded, since perceived sarcasm accounts for a related but different task.

Pre-processing steps. We perform minimal preprocessing which includes conversion of user mentions and links to @USER and URL, respectively. Also, consecutive whitespaces are normalized to a single occurrence and punctuation marks are surrounded with a single space character.

Our approach. Our approach is informed by insights from the literature on the effectiveness of sentiment/emotion information for sarcasm detection. Additionally, the creators of the SPIRS dataset also provided perspectives (intended vs. perceived) for their heuristically-labeled sarcasm dataset. Building on these, we examine the effect of these two knowledge sources on the sarcasm detection task. Specifically, we combine pre-training on sentiment/emotion with a second pre-training step on the intended sarcasm subset of the SPIRS dataset. In the final step, we fine-tune the pretrained model on the dataset provided for the shared task.

Pre-trained models. We employed four publicly available pre-trained models from the Huggingface model hub. The models namely: BERTweet-base[7], BERTweet-sentiment[8], BERTweet-emotion[9], and BERTweet-large[10] are trained on twitter data using masked language modeling as well as sentiment and emotion detection where applicable.

Training details/hyperparameters. We use the adapter-transformers library for the experiments. We add a classification layer (consisting of two dense layers) on top of the pooled output of the last transformer layer and optimize this layer jointly with the pre-trained layers. We optimize the model using Adamw with a batch size of 64 on a single Nvidia V100 GPU (32GB) and a maximum learning rate of 1e-5. We use a warmup ratio of 0.1 and set the maximum number of epochs to 15 with earlystopping on the validation performance metric (F1) using a patience of 5 evaluation runs. We evaluate the performance of the model every 20 steps on the validation set. Furthermore, we employ three regularization techniques: weight decay with a factor of 0.01, dropout applied to the pooled output of the last transformer layer with a probability of 0.2, and label smoothing with a factor of 0.1.

[6]https://github.com/bshmueli/SPIRS

[7]https://huggingface.co/vinai/bertweet-base

[8]https://huggingface.co/cardiffnlp/bertweet-base-sentiment

[9]https://huggingface.co/cardiffnlp/bertweet-base-emotion

[10]https://huggingface.co/vinai/bertweet-large

Model	Validation			Test		
	P	R	F1	P	R	F1
BERTweet-emotion-spirs	0.515	0.581	0.546	0.323	0.710	0.444
BERTweet-sentiment-spirs	0.545	0.558	0.552	0.330	0.705	0.450
BERTweet-spirs	0.526	**0.593**	0.557	0.349	0.710	0.468
BERTweet-large-spirs	**0.667**	0.558	**0.608**	**0.412**	**0.740**	**0.530**
Post-evaluation runs*						
BERTweet	0.573	0.547	0.560	0.395	0.680	0.500
BERTweet-emotion	0.529	0.535	0.532	0.378	0.630	0.473
BERTweet-sentiment	0.527	0.570	0.547	0.348	0.675	0.459
BERTweet-large	**0.690**	0.570	**0.624**	**0.420**	0.735	**0.535**
BERTweet-irony	0.581	0.581	0.581	0.399	0.665	0.499
BERTweet-irony-spirs	0.515	0.581	0.546	0.323	0.710	0.444

Table 2: Performance scores on the validation and test sets (All metrics are for the sarcastic class).
*Suggested by one of the reviewers for completeness and better comparison.

Case	Example	Explanation
Not enough context to determine the intention of the phrase(s)	- JUSTICE HAS BEEN DONE . - i ' ve never had protected sex - i'm dying	Chances are that the required context are on other parts of the post, e.g. in the comments
Common sense or Knowledge of real world is required	Mad how many cars they make now without indicators [happy-emoji]	It is not that new cars had a design without indicators, it is just that for many reasons, some people refuse to use them, and that make other drivers angry because of the difficulties while driving.
Common sense or Knowledge of real world is required	Vaccinated this morning . Not sure it ' s worked - still committed to Apple and no extra autism detected . Bloody science .	Those 'secondary effects' are not even related with vaccines; it seems that the user is just joking with that concept.
Common sense or Knowledge of real world is required	hello to all three of my followers, this is my big return to twitter	People who are familiar with Twitter knows that having three followers is not actually impressive. Instead this author is making fun of it.
Sentiment contradiction between key words and emojis	- I hate it here [happy emoji] - Headaches that last all day nonstop [happy emoji] - @ user thanks for shutting down the only Disney Store in northeastern Ohio . The other two stores went out of business yrs ago . It ' s not the same with shipping costs online not to mention LE doll sales are an absolute nightmare online . So unbelievably disappointing . [sad emoji]	Not all examples use positive emoji to end sarcastic messages. Here, the contradiction lies in the pair thanks-[sad emoji]

Table 3: Error analysis of model predictions on the validation set

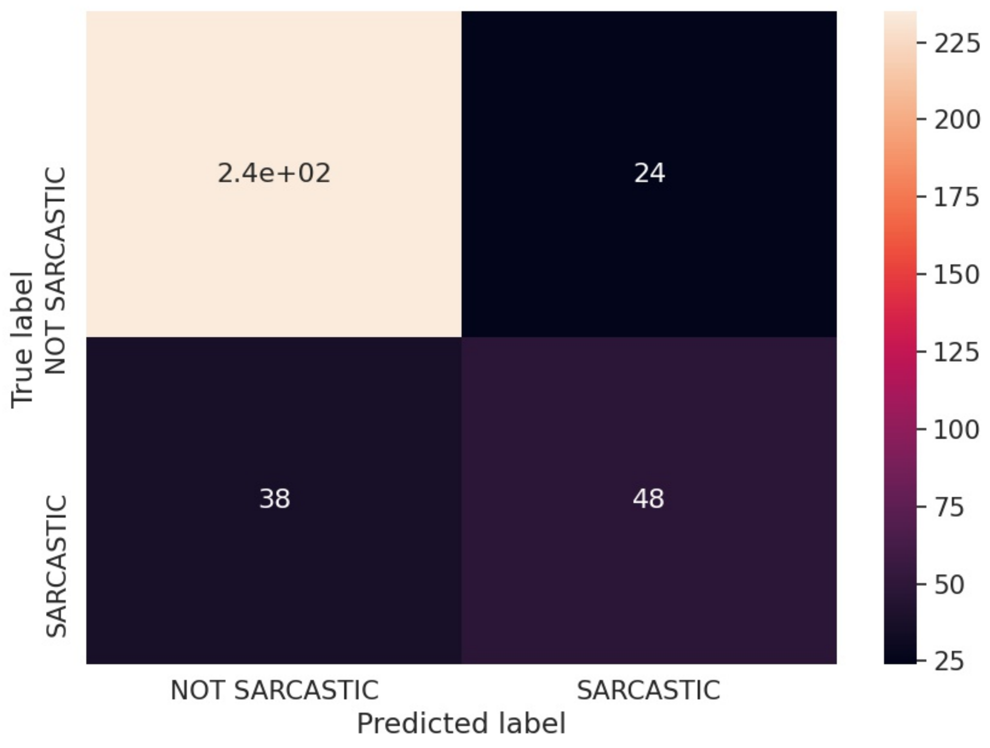


Figure 1: Confusion matrix for the best model predictions on the validation set

4 Results and error analysis

Our system ranks third on the leaderboard for the competition with an F1-score for the sarcastic class (positive label) of 0.530. The performance scores for our submissions on the validation and test sets are in Table 2. The BERTweet-large model achieved the best score out of our four submissions. It seems that pre-training on sentiment or emotion detection is not beneficial in our experimental settings. This observation can be confirmed in our post-evaluation runs where the BERTweet-base model shows superior performance (up to 5 F1 points on the test set) when compared to the same model that has been further pre-trained on sentiment or emotion detection task. Further investigation of this observation is required to identify alternative approaches to incorporate these sources of information for transformer-based models. Figure 1 shows the confusion matrix of the predictions made by our best model on the validation set.

As for error analysis, we use the validation set to identify possible explanations for why the model fails at predicting correctly positive labels (sarcastic tweets). Table 3 shows three cases as challenges for detecting sarcasm, which are usually hard to overcome in many NLP tasks as well.

These results show that still modern language model techniques struggle to grasp the pragmatics of messages expressed in social media, which are particularly difficult.

5 Conclusion

We present our systems for the task of sarcasm detection using a variety of knowledge sources that explore the capabilities of pre-trained language models to understand pragmatic phenomena such as sarcasm. We found that although these knowledge sources help to shed light over the sarcasm detection task, still more external knowledge would be required to correctly classify difficult cases that require a deeper understanding of real world context.

Based on our analysis, we plan to further examine the impact of including say emoji modelling to measure its influence, especially over cases that show a contradiction on expressed sentiments.

Acknowledgements

The work was done with partial support from the Mexican Government through the grant A1-S-47854 of the CONACYT, Mexico, grants 20211784, 20211884, and 20211178 of the Secretaría de Investigación y Posgrado of the Instituto Politécnico Nacional, Mexico. The authors thank the CONACYT for the computing resources brought to them through the Plataforma de Aprendizaje Profundo para Tecnologías del Lenguaje of the Laboratorio de Supercómputo of the INAOE, Mexico and acknowledge the support of Microsoft through the Microsoft Latin America PhD Award.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mondher Bouazizi and Tomoaki Ohtsuki. 2015. Opinion mining in twitter how to make use of sarcasm to enhance sentiment analysis. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1594–1597.

Simona Frenda, Alessandra Teresa Cignarella, Valerio Basile, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2022. The unbearable hurtfulness of sarcasm. *Expert Systems with Applications*, page 116398.

Debanjan Ghosh, Ritvik Shrivastava, and Smaranda Muresan. 2021. " laughing at you or with you": The role of sarcasm in shaping the disagreement space. *arXiv preprint arXiv:2101.10952*.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5):1–22.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.

Sandeepa Kannangara. 2018. Mining twitter for fine-grained political opinion polarity classification, ideology detection and sarcasm detection. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 751–752.

Bing Liu et al. 2010. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2(2010):627–666.

Diana G Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Lrec 2014 proceedings*. ELRA.

Pulkit Mehndiratta and Devpriya Soni. 2019. Identification of sarcasm using word embeddings and hyperparameters tuning. *Journal of Discrete Mathematical Sciences and Cryptography*, 22(4):465–489.

Silviu Vlad Oprea and Walid Magdy. 2020. The effect of sociocultural variables on sarcasm communication online. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–22.

Boaz Shmueli, Lun-Wei Ku, and Soumya Ray. 2020. Reactive Supervision: A New Method for Collecting Sarcasm Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2553–2559, Online. Association for Computational Linguistics.

Himani Srivastava, Vaibhav Varshney, Surabhi Kumari, and Saurabh Srivastava. 2020. A novel hierarchical bert architecture for sarcasm detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 93–97.

Haiyang Wang, Xin Song, Bin Zhou, Ye Wang, Liqun Gao, and Yan Jia. 2021. Performance evaluation of pre-trained models in sarcasm detection task. In *International Conference on Web Information Systems Engineering*, pages 67–75. Springer.

Setra Genyang Wicana, Taha Yasin İbisoglu, and Uraz Yavanoglu. 2017. A review on sarcasm detection from machine-learning perspective. In *2017 IEEE 11th International Conference on Semantic Computing (ICSC)*, pages 469–476. IEEE.

Uraz Yavanoglu, Taha Yasin Ibisoglu, and Setra Genyang Wıcana. 2018. Sarcasm detection algorithms. *International Journal of Semantic Computing*, 12(03):457–478.

YNU-HPCC at SemEval-2022 Task 6: Transformer-based Model for Intended Sarcasm Detection in English and Arabic

Guangmin Zheng, Jin Wang and Xuejie Zhang
School of Information Science and Engineering
Yunnan University
Kunming, China
Contact: gmzheng@mail.ynu.edu.cn, {wangjin, xjzhang}@ynu.edu.cn

Abstract

In this paper, we (a YNU-HPCC team) describe the system we built in the SemEval-2022 competition. As participants in Task 6 (titled "iSarcasmEval: Intended Sarcasm Detection In English and Arabic"), we implement the sentiment system for all three subtasks in English and Arabic. All subtasks involve the detection of sarcasm (binary and multilabel classification) and the determination of the sarcastic text location (sentence pair classification). Our system primarily applies the sequence classification model of a bidirectional encoder representation from a transformer (BERT). The BERT is used to extract sentence information from both directions for downstream classification tasks. A single basic model is used for single-sentence and sentence-pair binary classification tasks. For the multilabel task, the Label-Powerset method and binary cross-entropy loss function with weights are used. Our system exhibits competitive performance, obtaining 12/43 (21/32), 11/22, and 3/16 (8/13) rankings in the three official rankings for English (Arabic).

1 Introduction

Satirical text is a rhetorical device for implicitly expressing emotions by using words that are contrary to the actual intention to achieve a satirical or humorous linguistic effect. The true semantics of satirical texts cannot be directly inferred from the text vocabulary, and contradictions exist between their literal meaning and the true intention. Therefore, the detection of sarcasm and its sentiment discrimination are more challenging in natural language processing (NLP) problems.

Task 6 in the SemEval-2022 competition is a sarcasm detection task that consists of three subtasks (Abu Farha et al., 2022).

- Subtask A: Detect sarcastic meaning from a given tweet.
- Subtask B: Identify the tweet as "no sarcasm" or one or more of the six given sarcastic speech categories.
- Subtask C: Determine the position of the satirical text from the given tweets and their non-satirical restatements (0 means the first sentence is satirical, and 1 means the second sentence is satirical).

In previous sarcasm detection tasks, researchers used supervised learning methods based on the support vector machines and logistic regression (González-Ibáñez et al., 2011) to study ironic and non-ironic tweets that directly express positive and negative views. However, these traditional machine learning methods cannot mine the deep semantic information hidden in the text. Relying only on the surface semantic information can easily result in an incorrect judgment regarding irony. In contrast, deep learning methods show excellent results in deep semantic mining. By using the context information of the text to be detected (Bamman and Smith, 2015), we can further mine the behavior information of social users. We can achieve better performance by using a bidirectional recurrent neural network to capture the syntactic and semantic information of the target tweet text and the automatic learning features of historical tweets related to the target tweet for sarcasm detection (Zhang et al., 2016). The convolutional long-term and short-term memory network (CNN-LSTM-DNN) (Ghosh and Veale, 2016) achieved remarkable results.

In this paper, we propose a deep learning system for Task 6 in SemEval-2022, titled "iSarcasmEval: Intended Sarcasm Detection In English and Arabic." We use a pre-trained bidirectional encoder representation from a transformer (BERT) (Devlin et al., 2019) sequence classification model as the base model. For single-sentence and sentence-pair binary classification tasks, we use fine-tuning methods on a basic model. We employ the Label-

Powerset (Nazmi et al., 2020) approach and train the basic model by applying a binary cross-entropy loss function with weights for the multi-label task. The contributions of this study are as follows.

- For the sentiment analysis problem, we propose a basic model using a pre-trained BERT sequence classification model.
- The use of a binary cross-entropy loss function with weights is more advantageous for performing multi-label classification with uneven label distribution task.
- Fine-tuning on multilingual datasets (Pires et al., 2019) leads to a significant improvement in the scores of prediction results.

The remainder of this paper is organized as follows. In Section 2, we describe the proposed system and model in detail. The experiments and results are discussed in Section 3. Finally, conclusions are presented in Section 4.

2 System Description

The general structure of our system consists of four modules, described as follows.

Input layer. In this layer, we build text-processing tools to perform pre-processing of text and embedding of words. A large amount of useless information in the given text, such as user (@user), URL (http://ie.com), and escape symbols (\s, \j, etc.), increases the computational effort and complexity of a model. We remove the useless information in advance by using regular expressions and convert all words to lowercase, without breaking the structure of hashtags (#hashtag). We believe that the search and tagging of these tweets depends mainly on their hashtags (Peng et al., 2018). The preservation of the hashtag structure improves the accuracy of sarcasm detection. Tokenizers provided by HuggingFace[1] are used to process information such as punctuation, emoticons, non-English (or non-Arabic) letters, numbers, and hashtags (#hashtag) that are still present in the text and to rapidly perform word separation. Tokenizers can also perform word embedding using continuous low-dimensional vectors to represent word features (Mikolov et al., 2013). In this layer, we obtain the sequence of representations to be used as inputs to the subsequent modules.

[1]https://huggingface.co/.

Context encoder. Devlin et al. proposed a new natural language representation model named BERT in 2018, which successfully achieved state-of-the-art results in 11 NLP tasks, winning a plethora of accolades from the NLP community. The model is based on a bidirectional transformer for large-scale pre-training and can be fine-tuned by users to handle different text-processing tasks. One variant of the BERT is RoBERTa (Liu et al., 2019), which enhances the effect of the BERT by improving its pre-training method without affecting the structure of the BERT. Unlike the BERT static mask approach, RoBERTa uses dynamic masks, where the tokens masked for each sequence are changed in different epochs of training. RoBERTa removes the NSP task and uses the FULL-SENTENCES training approach. RoBERTa also uses a larger minibatch, more data, and larger number of sentences. Therefore, the RoBERTa-pretrained model yields better results. In this module, we mainly use the pre-trained BERT model and its variant RoBERTa model to complete the contextual encoder.

Fully connected layer. Fully connected networks are used for downstream classification tasks.

Output layer. The output of the fully connected layer is processed to complete the label prediction. Different tasks require different processing methods.

The details of each subtask in each module are discussed in the following.

2.1 Subtask A: Sentence Classification

The first part of this task was to extract semantic information from a given tweet text. We termed this sentence classification (Dao et al., 2020), where we predict whether a sentence is sarcastic. For this purpose, our approach generated 768-dimensional word embeddings for each word in a sentence by using a pre-trained BERT model. We selected the first token (i.e., "[CLS]") of the sentence into the sequence classification because it integrated the semantic information of the entire sentence. The word embeddings obtained from the previous step were then connected to a fully connected layer, which transformed the 768-dimensional input into two-dimensional values. These values were then fed into Softmax to calculate the probability that the sentence is sarcastic. Finally, the probability results were fed into argmax to form labels; in our experimental setup, 1 indicated sarcasm and 0 indicated non-sarcasm. The model architecture is

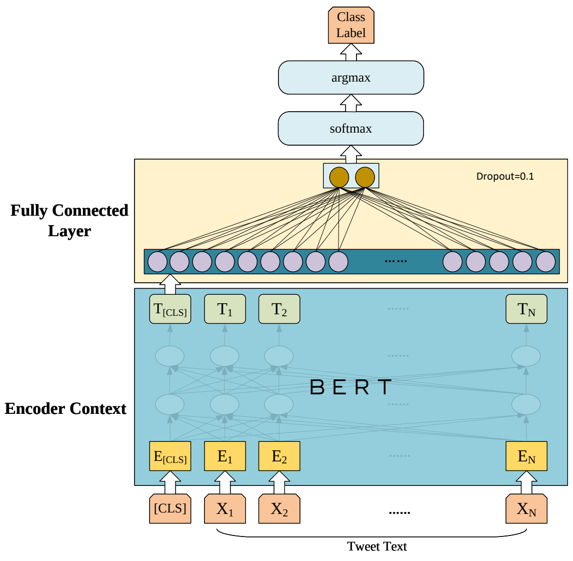


Figure 1: System of binary classification for sentence

Table 1: Quantity of labels by category in English dataset.

Category	Quantity
sarcasm	713
irony	155
satire	25
understatement	10
overstatement	40
rhetorical_question	101

illustrated in Figure 1.

2.2 Subtask B: Multi-Label Classification

This task was also a sentence classification task, with the difference being that it involved the prediction of multiple binary targets simultaneously based on a given input. For the upstream task of sentence information extraction, we used the pre-trained BERT model. For the downstream classification task, we used the Label-Powerset method (Tsoumakas et al., 2011), which sets the number of labels to the number of output neurons in the network. We can directly apply an arbitrary binary classification loss function to the neural network model, and the model can simultaneously output all targets. At this point, we only need to train one model; the training time is shorter, and the network can also learn the relevance of different labels through the output neurons. We counted the number of various types of labels in the English dataset (Table 1) and discovered that the number of labels is dozens of times different. To solve the problem of label imbalance, we appropriately added a network layer to improve the classification effect. We employed the RELU activation function for fast computation between two linear layers to boost the sparsity of the simultaneous network and reduce the interdependence of parameters to alleviate the overfitting problem. In addition, we utilized a binary cross-loss function with weights to focus the model on the sparse labels, as stated in Section 2.4.

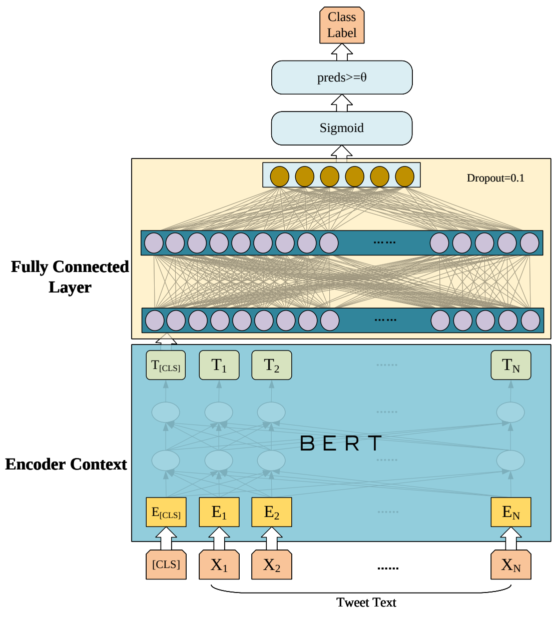


Figure 2: Multi-label classification system for sentence

We selected the values generated by the aforementioned work into sigmoid and mapped their range to (0, 1) to reply to the confidence level of each label. We set a threshold of $\theta = 0.5$, and when the predicted value was greater than or equal to θ, we considered that the prediction contained this label; otherwise, it did not. The model architecture is shown in Figure 2.

2.3 Subtask C: Sentence-Pair Classification

This task is a sentence pair classification task, similar to subtask A, but with two sentences as the input. In the input layer, in addition to the work done in Subtask A, we must split the two sentence representations using the "[SEP]" token. The inputs are then passed to the pre-trained BERT model to obtain the "[CLS]" token, which integrates the semantic information of the entire sentence. The subsequent model structure was the same as that for Subtask A. The "[CLS]" token is passed to the subsequent module to obtain the location label of

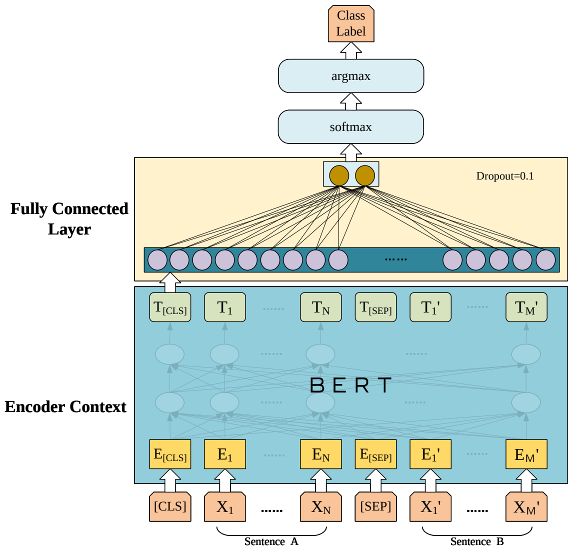


Figure 3: System of binary classification for sentence pair

the sarcastic text, with 0 indicating sentence A and 1 indicating sentence B. The model architecture is illustrated in Figure 3.

2.4 Training and Hyper-parameters

We used a binary cross-entropy loss function for subtasks A and C and a binary cross-entropy loss function with weights for subtask B to train the model. For the sample imbalance between labels in the multilabel classification task of subtask B, we introduced parameter w_i. We increased the attention of the system to scarce labels by assigning higher weights to labels that appear less frequently. For batch data D(x, y) containing N samples with M labels per sample, the loss function is calculated as follows:

$$loss = \frac{1}{\text{N}} \sum_{n=1}^{\text{N}} l_n, \tag{1}$$

$$l_n = \frac{1}{\text{M}} \sum_{i=1}^{\text{M}} l_n^i, \tag{2}$$

where l_n^i is the loss corresponding to the i-th label of the n-th sample.

$$l_n{}^i = -w_i[y_n^i \cdot \log \sigma(x_n^i) + (1 - y_n^i) \cdot \log(1 - \sigma(x_n^i))] \tag{3}$$

where σ is the sigmoid function. w_i is the weight parameter of the i-th label, which is calculated from the i-th label number, $Label_i$, by applying the following equation.

$$w_i = \frac{\sum Label}{Label_i} \tag{4}$$

For all the subtasks, we used the AdamW (Loshchilov and Hutter, 2019) optimizer to train the model with a batch size of 16.

Hyperparameters. The dimensionality (d) of the word embedding was 100, learning rate was 2e-5, weight decay was 0.01, and dropout ratio was 0.1 for all layers in all models.

3 Experimental

Datasets. The datasets used were the English dataset (3467 data) and Arabic dataset (2602 data) provided by the competition, with no other external corpus. Only one dataset in English was used for subtask B, except for subtasks A and C, in which two sub-datasets were used. We thank the organizers for their contributions to the data.

Evaluation Metrics. The main evaluation metrics are as follows:

- SubTask A: F1-score for the sarcastic class.
- SubTask B: Macro-F1 score.
- SubTask C: Accuracy.

Implementation Details. To solve the data imbalance problem, we sampled the data selected for training such that the number of 0/1 tags was as similar as possible. For the sentence pair classification task, we take the tweet text in the dataset as sentence A and the rephrase text as sentence B, and assign label 0. After switching the positions of tweet text and rephrase text, we assign label 1. We divided the training data into a training set and a development set at a ratio of 8:2. We trained our models on the training set, evaluated the predictions on the development set using evaluation measures, and saved the models with the highest evaluation scores during the process. We used the Pytorch framework provided by the Huggingface library for the pre-trained BERT model and bert-base-multilingual-uncased and roberta-base for the binary classification, multi-label classification, and sentence pair classification included in this task.

Results and Analysis. Tables 2, 3, and 4 present comparative results for each task. The bolded scores are those assessed by participating in the competition leaderboard. On the competition leaderboard, our system ranked 12/43 (21/32) in

Table 2: Comparable results of experiments for subtask A.

Model	Fine-tune	Test	F1
bert-base-multilingual-uncased	English	English	0.306
	English+Arabic		0.285
	Arabic	Arabic	0.202
	English+Arabic		0.245
roberta-base	English	English	**0.392**
	Arabic	Arabic	**0.323**

Table 3: Comparable results of experiments for subtask B.

Model	Loss	Macro-F1
6-binary	BCE	0.0702
bert-base-multilingual-uncased	BCE	0.0672
	BCEwithWeight	0.0795
	Dice	0.0379
	Focal	**0.0646**

Table 4: Comparable results of experiments for subtask C.

Model	Fine-tune	Test	Acc
bert-base-multilingual-uncased	English	English	0.815
	English+Arabic		**0.805**
	Arabic	Arabic	0.5
	English+Arabic		**0.755**
roberta-base	English	English	0.860
	Arabic	Arabic	0.5

English (Arabic) in Task A, 11/22 in Task B, and 3/16 (8/13) in Task C.

As shown in the tables, our approach achieves significant results. This is mainly because the BERT model is a multilayer bidirectional transformer encoder that can be integrated into various NLP downstream tasks and obtains the best results. In addition, the training results using the roberta-base model are significantly better, indicating that the pre-training approach of the BERT can be improved. Moreover, fine-tuning with multiple languages significantly improved the training results, particularly for Arabic. As revealed in Table 4, fine-tuning the model on the Arabic dataset individually shows worse results, which may be due to the fact that very little information is extracted from the Arabic dataset only. In the future, we can improve the method of fine-tuning in multiple-language datasets by balancing the vectors of different languages to improve the model effect. For multi-label classification, the Label-Powerset method is not as effective as stand-alone dichotomous classifier training; however, combined with a dichotomous cross-entropy loss function with weights, the results can be slightly improved while saving a lot of training time.

4 Conclusion

In this paper, we describe a deep learning model for the sentiment analysis task in the SemEval-2022 competition (Task 6: iSarcasmEval: Intended Sarcasm Detection In English and Arabic). A BERT sequence classification model was used as the base model. The final submitted system performed admirably and ranked third in one of the rankings. However, there is still considerable potential for improvement. Therefore, in future investigations, we will attempt to extend this model with improved capabilities to achieve better outcomes.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61966038. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

David Bamman and Noah A. Smith. 2015. Contextualized sarcasm detection on twitter. In *Proceedings of the 9th International Conference on Web and Social Media (ICWSM 2015)*, pages 574–577.

Jiaxu Dao, Jin Wang, and Xuejie Zhang. 2020. YNU-HPCC at SemEval-2020 task 11: LSTM network for detection of propaganda techniques in news articles. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1509–1515, Barcelona (online). International Committee for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th Workshop on Computational Approaches to*

Subjectivity, Sentiment and Social Media Analysis, pages 161–169, San Diego, California. Association for Computational Linguistics.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in Twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586, Portland, Oregon, USA. Association for Computational Linguistics.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *arXiv preprint arXiv: 1711.05101*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv: 1301.3781*.

Shabnam Nazmi, Xuyang Yan, Abdollah Homaifar, and Emily Doucette. 2020. Evolving multi-label classification rules by exploiting high-order label correlations. *Neurocomputing*, 417:176–186.

Bo Peng, Jin Wang, and Xuejie Zhang. 2018. YNU-HPCC at SemEval-2018 task 3: Ensemble neural network models for irony detection on Twitter. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 622–627, New Orleans, Louisiana. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2011. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2449–2460, Osaka, Japan. The COLING 2016 Organizing Committee.

UTNLP at SemEval-2022 Task 6: A Comparative Analysis of Sarcasm Detection Using Generative-based and Mutation-based Data Augmentation

Amirhossein Abaskohi, Arash Rasouli*, Tanin Zeraati*, Behnam Bahrak
School of Electrical and Computer Engineering, College of Engineering
University of Tehran, Tehran, Iran
{amir.abaskohi, arash.rasouli, t.zeraati, bahrak}@ut.ac.ir

Abstract

Sarcasm is a term that refers to the use of words to mock, irritate, or amuse someone. It is commonly used on social media. The metaphorical and creative nature of sarcasm presents a significant difficulty for sentiment analysis systems based on affective computing. The methodology and results of our team, UTNLP, in the SemEval-2022 shared task 6 on sarcasm detection are presented in this paper. We put different models, and data augmentation approaches to the test and report on which one works best. The tests begin with traditional machine learning models and progress to transformer-based and attention-based models. We employed data augmentation based on data mutation and data generation. Using RoBERTa and mutation-based data augmentation, our best approach achieved an F1-sarcastic of 0.38 in the competition's evaluation phase. After the competition, we fixed our model's flaws and achieved an F1-sarcastic of 0.414.

1 Introduction

Billions of internet users use social networks not only to stay in touch with friends, meet new people, and share user-generated content but also to express their opinions on a wide range of topics using a variety of methods such as posting comments, videos, photos, etc. with specific groups of people (Tungthamthiti et al., 2016). In these platforms, users could submit information on whatever topic they wanted, with no restrictions on the sort of content they may share. The lack of constraints and individuals' anonymity on these networks led to humorous sarcastic texts.

Because sarcasm indicates sentiment, detecting sarcasm in a text is critical for anticipating the text's accurate sentiment, making sarcasm detection a valuable tool with multiple applications in domains such as security, health, services, product evaluations, and sales. Sarcasm detection is an essential aspect of creative language comprehension (Veale et al., 2019) and online opinion mining (Kannangara, 2018). Even for humans, identifying sarcasm is difficult due to heavily contextualized expressions (Walker et al., 2012). There are few labeled data resources for sarcasm detection. Any available texts that can be collected (for example, Tweets) contain many issues, such as an evolving dictionary of slang words and abbreviations, requiring many hours of human annotation to prepare the data for any potential use. Furthermore, the nature of sarcasm identification adds to the task's difficulty, as sarcasm may be considered relative and varies significantly across people, depending on a variety of criteria such as the context, area, time, and events surrounding the statement.

In an attempt to solve this issue, we participated in SemEval-2022 shared task 6 (Abu Farha et al., 2022), which aims to recognize whether a tweet is sarcastic or not. Our contributions are as follows:

1. We experiment with simple machine learning models like Support Vector Machine (SVM) and various word encodings.
2. To discover the optimum data preprocessing method, we tested the effect of various data preprocessing.
3. We put several data augmentation techniques to the test.
4. On our best dataset, we evaluated Long Short Term Memory (LSTM) based models, Bidirectional Encoder Representations from Transformers (BERT) based models, and attention-based models. Different neural network topologies are compared, and the model with the highest performance is reported.

With RoBERTa (A Robustly Optimized BERT Pretraining Approach), no preprocessing, and mutation-based data augmentation, our top result gets an F1-sarcastic of 0.38. However, we obtain

*equal contribution

better outcomes, with a 0.414 F1-sarcastic after fixing the problems of our proposed method.

The rest of this paper is organized as follows. In Section 2, we discuss the related work, Section 3 introduces the dataset. In Sections 4 and 5, we present our methodology and results, respectively. Finally Section 6 concludes the paper.

2 Related Work

We give a quick review of previous works on sarcasm detection in this part, followed by works on data augmentation.

2.1 Sarcasm Detection on Twitter

Sarcasm detection has been represented as a binary classification issue, with most tweets labeled with specific hashtags (e.g., #sarcasm, #sarcastic) being considered sarcastic. Many techniques in various languages have been proposed using this framework.

In (Davidov et al., 2010), Semi-supervised sarcasm detection experiments were done using a Twitter dataset (5.9 million tweets) and 66,000 Amazon product evaluations. On the product review dataset, they acquired an F-measure of 0.83. On the Twitter dataset, they obtained an F-measure of 0.55 using 5-fold cross-validation on their k-Nearest Neighbor (kNN) like classifier.

(González-Ibánez et al., 2011) used 900 messages from Twitter sorted into three groups (sarcastic, positive sentiment, and negative sentiment). To find sarcastic tweets, they utilized the hashtags #sarcasm and #sarcastic. SVM with Sequential Minimum Optimization (SMO) and logistic regression were employed as classifiers. The best accuracy for the sarcastic class was 0.65.

(Reyes et al., 2012) presented elements to capture ambiguity, polarity, unexpectedness, and emotive situations in figurative language. F1-sarcastic of 0.65 was the best result in categorizing irony and general tweets.

The representativeness and significance of conceptual elements have been investigated in (Reyes et al., 2013). Punctuation marks, emoticons, quotations, capitalized words, lexicon-based features, character n-grams, skip-grams, and polarity skip-grams are all examples of these characteristics. Each of the four categories (irony, comedy, education, and politics) in their corpus has 10,000 tweets. Using the Naive Bayes and decision trees algorithms, they evaluated two distributional scenarios: balanced distribution and unbalanced distribution (25% ironic tweets and 75% tweets from the three non-ironic categories). The decision trees classified the balanced distribution with an F1-sarcastic of 0.72 and the unbalanced distribution with an F1-sarcastic of 0.53.

One sort of sarcasm identified by (Riloff et al., 2013) is the difference between a good mood and a bad scenario. Using a bootstrapping approach, the authors gathered collections of positive sentiment phrases and negative circumstance words from sarcastic tweets. They suggested a method for classifying tweets as sarcastic if they contain a positive predictive close to a negative context phrase. They used a SVM classifier using unigrams and bigrams as features to evaluate a human-annotated dataset of 3000 tweets (23% sarcastic), getting an F1-sarcastic of 0.48. The F1-sarcastic of the hybrid strategy, which combined the findings of the SVM classifier with their baseline method, was 0.51.

(Lukin and Walker, 2017) used bootstrapping, syntactic patterns, and a high precision classifier to classify sarcasm and nastiness in online chats. On their snark dataset, they got an F1-sarcastic of 0.57.

In (Oprea and Magdy, 2019), LSTM, Att-LSTM, CNN, SIARN, MIARN, 3CNN, and Dense-LSTM models were used to assess the task dataset that was introduced in (Oprea and Magdy, 2019), which is an unbalanced dataset and labeled by the tweets' writers. Using Multi-Dimension Intra-Attention (MIARN) (Tay et al., 2018) Network, they could get an F-score of 0.364.

In (Guo et al., 2021), the Latent Optimized Adversarial Neural Transfer (LOANT) model was suggested as a novel latent-optimized adversarial neural transfer model for cross-domain sarcasm detection. LOANT surpasses classical adversarial neural transfer, multitask learning, and meta-learning baselines using stochastic gradient descent (SGD) with a one-step look-ahead and sets a new state-of-the-art F-score of 0.4101 on the iSarcasm dataset.

2.2 Data Augmentation

Natural Language Processing(NLP) encompasses a wide range of tasks, from text categorization to question answering, but no matter what you do, the quantity of data you have to train your model has a significant influence on the model's performance. Using the data you already have, data augmentation techniques are used to produce extra, synthetic data. Augmentation techniques are widely used in computer vision applications, but they may also be used in natural language processing.

In the instance of Twitter, (Van Hee et al., 2018) and (Ilić et al., 2018) found that adding more data from the same domain did not improve the performance for recognizing sarcasm and irony. Although their result is not general for all sarcasm detection tasks and the result of data augmentation depends on the data and augmentation method.

(Lee et al., 2020)'s idea is to make a new datapoint out of the context sequence [c1, c2,, cn] and label it "NOT SARCASM." The sequence could not be identified as "SARCASM" without the answer [r1]. They believe that the newly created negative samples will aid the model in focusing on the link between the response [r1] and its contexts [c1, c2, cn]. They also create positive samples using back-translation procedures(Berard et al., 2019; Zheng et al., 2019) in French, Spanish, and Dutch to balance out the quantity of negative examples.

In (Feng et al., 2020) different data augmentation methods were tested on Yelp Reviews dataset(Yel, 2014) for GPT-2 generative model(Radford et al., 2019). They used "Random Insertion, Deletion, & Swap", "Semantic Text Exchange (STE)", "Synthetic Noise", and "Keyword Replacement". They showed in some case data augmentation could help them to reach better performance.

This paper is the first to look at generative-based and mutation-based data augmentation strategies in sarcasm detection.

3 Dataset

We mostly used the iSarcasm (Oprea and Magdy, 2019) dataset in this study. In specific experiments, we integrated the primary dataset with various secondary datasets, including the Sarcasm Headlines Dataset (Misra and Arora, 2019) and Sentiment140 dataset (Go et al., 2009) to increase the quantity of data and compensate for the lack of sarcastic data. For each dataset, the details are further discussed. It is worthy to mention that all of the supplementary datasets we included had a negative impact on our model's performance. We believe this was the result of a different data gathering method. Because to the differing labeling process and domain, the distribution diverged from that of iSarcasm. As a result, the following sections are solely dependent on the iSarcasm dataset, with no other datasets being used.

3.1 Main Task Dataset: iSarcasm

According to (Oprea and Magdy, 2019), the sarcasm labeling using hashtags to build datasets captures just the sarcasm that the annotators were able to detect, leaving out the intended sarcasm. When the author intends for the content to be sarcastic, it is called intended sarcasm. The iSarcasm dataset includes 4484 tweets: 3707 non-sarcastic and 777 sarcastic. Because some tweets had been erased, we only had access to 3469 tweets for the job. The unbalanced dataset and the scarcity of sarcastic data were two of the most significant issues we encountered. Table 1 displays some of the dataset's annotated remarks.

3.2 Sarcasm Headlines Dataset

Sarcasm Headlines Dataset (Misra and Arora, 2019; Misra and Grover, 2021) was gathered from two news websites. It is beneficial since it overcomes the constraints of Twitter datasets due to noise. As the second edition of this dataset includes more data and a greater variety of data than the first version, we chose the second version.

3.3 Sentiment140 Dataset

We needed to compensate for the limited data to train our model successfully. As a result, we chose the sentiment140 dataset (Go et al., 2009) because it has a large quantity of data and is based on Twitter. The sentiment tweet message is labeled using an automated classification approach in this dataset. The accuracy is more than 80% when using a machine learning algorithm.

4 Methodology

In this study we examined and analyzed various models and data augmentation strategies for sarcasm detection. First, we go through data augmentation methods; then, we discuss the structure and hyperparameters of these models in this section. The codes of all models are available on GitHub[1].

4.1 Data Augmentation

4.1.1 Generator-based

For this augmentation method, we used GPT-2 (Radford et al., 2019) generative model to generate 4000 tweets for both sarcastic and non-sarcastic classes. Then we selected 2000 tweets of each

[1]https://github.com/AmirAbaskohi/SemEval2022-Task6-Sarcasm-Detection

Table 1: Example of Sarcastic and Non-Sarcastic tweets.

Tweet	Sarcastic	Sarcasm Type
Oh my goodness. It's the first week of the summer holidays and @name has found his recorder Give.Me.Strength.	Sarcastic	['Sarcasm']
90% of adulthood is just refilling your @name pitcher.	Sarcastic	['Irony', 'overstatement']
True bliss is laying in an ice cold bath during the hottest part of the year	Non-Sarcastic	[]

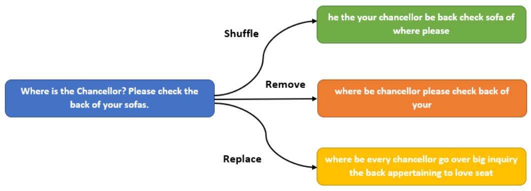

Where is the Chancellor? Please check the back of your sofas.
Shuffle
he the your chancellor be back check sofa of where please
Remove
where be chancellor please check back of your
Replace
where be every chancellor go over big inquiry the back appertaining to love seat

Figure 1: Effect of shuffling, word elimination, and replacing with synonyms on a tweet sample.

class randomly to increase dataset quantity and have more sarcastic samples.

4.1.2 Mutation-based

We used three distinct ways to change the data in this method: eliminating, replacing with synonyms, and shuffling. These processes were used in the following order: shuffling, deleting, and replacing. The removal and replacement were carried out systematically. We used the words' roots to create a synonym dictionary. Synonym dictionary is created by scarping the Thesaurus website[2]. When a term was chosen to be swapped with its synonyms, we chose one of the synonyms randomly (Figure 1). We tried each combination of these processes to find the best data augmentation combination (a total of seven).

4.2 Models

4.2.1 Support Vector Machine (SVM)

We utilized SVM to discover the optimal approaches for dataset preprocessing and word embeddings. For data augmentation, we employed both generator-based and mutation-based methods. We also put other data preprocessing approaches to the test, such as link removal, emoji removal, stop word removal, stemming, and lemmatizing. We utilized TF-IDF, Word2Vec (Mikolov et al., 2013), and BERT (Devlin et al., 2018) for word embedding. We found that using a regularization value of 10 and a Radial Basis Function (RBF) kernel, BERT word embedding, and no data preprocessing will give us the best results.

4.2.2 LSTM-based Methods

We begin with the intuition that a memory model can help us reach a better result. So we started with Long Short Term Memory (LSTM) model (Hochreiter and Schmidhuber, 1997). We used one LSTM layer followed by time distributed dense layer. We repeated these two layers one more time, and then we used another LSTM layer followed by two dense layers. This model and all of the following models in this section were trained in 10 epochs.

In addition, we used Bidirectional Long Short Term Memory (BLSTM). Using bidirectional will run the inputs in two directions, one from past to future and the other from future to past. We used one BLSTM layer for this network, followed by a time-distributed dense layer. We repeated these two layers one more time, and then we used another BLSTM layer followed by two dense layers.

Furthermore, we combined LSTM and BLSTM with Convolutional Neural Networks (CNNs). CNN layers for feature extraction on input data are paired with LSTM to facilitate sequence prediction in the CNN-LSTM architecture. Although this model is often employed for video datasets, (Rehman et al., 2019) demonstrated that it could perform better in sentiment analysis tasks. We used three 1D convolutional layers followed by a 1D global max-pooling layer for the convolutional part. We used these layers at the end of LSTM-based networks.

[2]https://www.thesaurus.com

4.2.3 BERT-based Methods

The use of bidirectional training of transformer and a prominent attention mode for language modeling is BERT's fundamental technological breakthrough (Devlin et al., 2018). The researchers describe a new Masked Language Model (MLM) approach that permits bidirectional training in previously tricky models. They found that bidirectionally trained language models can have a better understanding of language context and flow than unidirectional ones.

Robustly Optimized BERT or RoBERTa has a nearly identical architecture to BERT, however, the researchers made some minor adjustments to its architecture and training technique to enhance the results on BERT architecture (Liu et al., 2019).

We used both RoBERTa with twitter-roberta-base, which has been trained on near 58 million tweets and finetuned for sentiment analysis with the TweetEval benchmark and BERT with bert-base from Huggingface (Wolf et al., 2019). For both models, we employed five epochs, batch size of 32, 500 warmup steps, and a weight decay of 0.01.

4.2.4 Attention-based Methods

One of the most important achievements in deep learning research in the recent decade is the attention mechanism (Vaswani et al., 2017). The Encoder-Decoder model's restriction of encoding the input sequence to one fixed-length vector to decode each output time step is addressed via an attention mechanism. This difficulty is thought to be more prevalent when decoding extended sequences.

We start with the assumption that if a model with an attention layer is trained to identify sarcasm at the sentence level, the sarcastic words will be the ones the attention layer learns to value. As a result, we added an attention layer to our LSTM-based and BERT-based models. The results will be discussed further.

4.2.5 Google's T5

Google's T5 (Raffel et al., 2019) text-to-text model outperformed the human baseline on the GLUE, SQuAD, and CNN/Daily Mail datasets and earned a remarkable 88.9 on the SuperGLUE language benchmark.

We fine-tuned T5 for our problem and dataset by giving the sarcastic label the target and the tweets as the source. We used two epochs, batch size of 4, 512 tokenization max length, Adam epsilon of 1e-8, word decay of 0, no warmup steps, and learning rate of 3e-4 (Figure 2)[3].

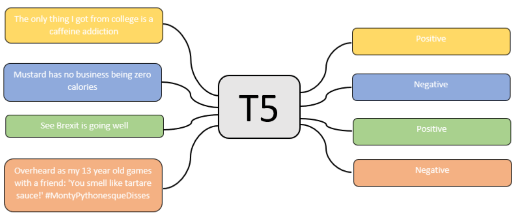

The only thing I got from college is a caffeine addiction
Mustard has no business being zero calories
See Brexit is going well
Overheard as my 13 year old games with a friend: 'You smell like tartare sauce!' #MontyPythonesqueDisses
T5
Positive
Negative
Positive
Negative

Figure 2: Fine-tuning T5 model for sarcasm detection problem.

Table 2: F1-sarcastic and accuracy for different data augmentation methods on SVM model with BERT word embedding and no preprocessing.

Data Augmentation	F1-sarcastic	Accuracy
Shuffling	0.305	0.7471
Shuffling + Replacing	0.301	0.741
Shuffling + Removing	0.306	0.747
Removing	0.301	0.747
GPT-2	0.292	0.675

5 Results

In this section we report the results of our models introduced in Section 4.

It's important to note that after the competition, we discovered that none of our preprocessing strategies improved the performance of our model. So we were able to get an F1-sarcastic of 0.414 without using any preprocessing methods, which was 0.034 higher than our performance in the competition, which was based on the best combination of preprocessing methods.

5.1 Support Vector Machine (SVM)

The optimum augmentation technique, preprocessing method, and word embedding were all determined using the SVM model. Without any augmentation, BERT obtained the greatest F1-sarcastic of 0.2862, compared to 0.2541 and 0.0924 for Word2Vec and TF-IDF, respectively.

We have also looked at several ways of data augmentation. The F1-sarcastics for shuffling with replacing words, only word elimination, just shuffling, and shuffling with word elimination were the highest in the mutation-based augmentation (Table 2). We also tried these data augmentation and

[3]We were not able to test a larger version of the model with better hyperparameters due to resource constraints

Table 3: Best results for each model using iSarcasm dataset and mutation-based data augmentation.

Model	F1-sarcastic	Accuracy
SVM	0.3064	0.7478
LSTM-based	0.2751	0.7251
BERT-based	0.414	0.8634
Attention-based	0.2959	0.7793
Google's T5	0.4038	0.8124

GPT-2 data augmentation on RoBERTa because the results were close, and we found that merely word removal was the best data augmentation. The following results are based on no data preprocessing, BERT word embedding, and mutation-based data augmentation utilizing only word removal.

5.2 LSTM-based Methods

LSTM obtained an F1-sarcastic of 0.2176 using BERT word embeddings, mutation-based data augmentation, and no preprocessing, whereas BLSTM's F1-sarcastic was 0.2439 using BERT word embeddings, mutation-based data augmentation, and no preprocessing. By adding CNN layers, the F1-sarcastic of the LSTM was increased to 0.2453, and the BLSTM was increased to 0.2751. The CNN model's F1-sarcastic was 0.2263.

5.3 BERT-based Methods

We employed a mutation-based data augmentation approach with no preprocessing for BERT-based procedures. We got an F1-sarcastic of 0.323 using BERT. We achieved our best result with RoBERTa with an F1-sarcastic of 0.414, which was better than LOANT (Guo et al., 2021) model on the same dataset.

5.4 Attention-based Methods

Adding attention layers to this job was not helpful, and it decreased our models' performance. RoBERTa's F1-sarcastic dropped to 0.2959 using the attention layer. LSTM model with the attention layer earned an F1-sarcastic of 0.2145. The F1-sarcastic of BLSTM with attention layer was 0.2336.

5.5 Google's T5

Based on the hyperparameters listed in the Section 4, our F1-sarcastic for this model is 0.4038. However, we believe that we may get better results by increasing the tokenization max length, increasing the batch size, and utilizing the t5-large pre-trained model.

6 Conclusion

In this study, we reviewed and contrasted a number of sarcasm detection methods. To improve the performance of our model, we experimented with two different types of augmentation. In the job of sarcasm detection, we observed that mutation-based data augmentation can assist us in achieving better results than generative-based data augmentation. Additionally, we tested with other deep-learning techniques, including RNN and BERT-based models. Our best system, an ensemble model, has an F1-sarcastic of 0.414.

Acknowledgements

We want to convey our heartfelt gratitude to Prof. Yadollah Yaghoobzadeh, who provided us with invaluable advice during our research.

References

2014. Yelp. yelp open dataset. `http://www.yelp.com/dataset_challenge`.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alexandre Berard, Ioan Calapodescu, and Claude Roux. 2019. Naver labs europe's systems for the wmt19 machine translation robustness task. *arXiv preprint arXiv:1907.06488*.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcasm in twitter and amazon. In *Proceedings of the fourteenth conference on computational natural language learning*, pages 107–116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Steven Y Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. 2020. Genaug: Data augmentation for finetuning text generators. *arXiv preprint arXiv:2010.01794*.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586.

Xu Guo, Boyang Li, Han Yu, and Chunyan Miao. 2021. Latent-optimized adversarial neural transfer for sarcasm detection. *arXiv preprint arXiv:2104.09261*.

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

Suzana Ilić, Edison Marrese-Taylor, Jorge A Balazs, and Yutaka Matsuo. 2018. Deep contextualized word representations for detecting sarcasm and irony. *arXiv preprint arXiv:1809.09795*.

Sandeepa Kannangara. 2018. Mining twitter for fine-grained political opinion polarity classification, ideology detection and sarcasm detection. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 751–752.

Hankyol Lee, Youngjae Yu, and Gunhee Kim. 2020. Augmenting data for sarcasm detection with unlabeled conversation context. *arXiv preprint arXiv:2006.06259*.

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

Stephanie Lukin and Marilyn Walker. 2017. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. *arXiv preprint arXiv:1708.08572*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Rishabh Misra and Prahal Arora. 2019. Sarcasm detection using hybrid neural network. *arXiv preprint arXiv:1908.07414*.

Rishabh Misra and Jigyasa Grover. 2021. Sculpting data for ml: The first act of machine learning.

Silviu Oprea and Walid Magdy. 2019. isarcasm: A dataset of intended sarcasm. *arXiv preprint arXiv:1911.03123*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Anwar Ur Rehman, Ahmad Kamran Malik, Basit Raza, and Waqar Ali. 2019. A hybrid cnn-lstm model for improving accuracy of movie reviews sentiment analysis. *Multimedia Tools and Applications*, 78(18):26597–26613.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering*, 74:1–12.

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Language resources and evaluation*, 47(1):239–268.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading inbetween. *arXiv preprint arXiv:1805.02856*.

Piyoros Tungthamthiti, Kiyoaki Shirai, and Masnizah Mohd. 2016. Recognition of sarcasm in microblogging based on sentiment analysis and coherence identification. *Journal of Natural Language Processing*, 23(5):383–405.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tony Veale, F Amílcar Cardoso, and Rafael Pérez y Pérez. 2019. Systematizing creativity: A computational view. In *Computational Creativity*, pages 1–19. Springer.

Marilyn Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012. A corpus for research on deliberation and debate. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 812–817.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Renjie Zheng, Hairong Liu, Mingbo Ma, Baigong Zheng, and Liang Huang. 2019. Robust machine translation with domain sensitive pseudo-sources:

Baidu-osu wmt19 mt robustness shared task system report. *arXiv preprint arXiv:1906.08393*.

FII_UAIC at SemEval-2022 Task 6: iSarcasmEval - Intended Sarcasm Detection in English and Arabic

Tudor Manoleasa
Faculty of Computer Science, Alexandru Ioan Cuza University of Iasi, Romania
tudor.manoleasa@info.uaic.ro

Iustin Sandu
Faculty of Computer Science, Alexandru Ioan Cuza University of Iasi, Romania
iustin.sandu@info.uaic.ro

Daniela Gifu
Faculty of Computer Science, Alexandru Ioan Cuza University of Iasi, Romania
Institute of Computer Science, Romanian Academy - Iasi Branch
daniela.gifu@info.uaic.ro

Diana Trandabat
Faculty of Computer Science, Alexandru Ioan Cuza University of Iasi, Romania
diana.trandabat@info.uaic.ro

Abstract

The "iSarcasmEval - Intended Sarcasm Detection in English and Arabic" task at the SemEval 2022 competition focuses on detecting and rating the distinction between intended and perceived sarcasm in the context of textual sarcasm detection, as well as the level of irony contained in these texts. In the context of SemEval, we present a binary classification method which classifies the text as sarcastic or non-sarcastic (task A, for English) based on five classical machine learning approaches by trying to train the models based on this dataset solely (i.e., no other datasets have been used). This process indicates low performance compared to previously studied datasets, which indicates that the previous ones might be biased.

1 Introduction

One of the most challenging tasks facing natural language processing (NLP) is the automatic figurative language detection (Gifu Daniela, Samson Mihai, 2021), (C. Van Hee, E. Lefever, and V. Hoste, 2018), such as humor, sarcasm or irony. In general, this way of expressing takes advantage of linguistic elements in order to project complex and explainable meanings. In this paper, we investigate the binary classification models for figurative language, as is sarcasm (Dan Alexandru and Daniela Gîfu, 2020). In general, it is omnipresent on the public space, being disruptive of computational systems that harness this data to perform tasks such as opinion mining and sentiment analysis in elections (Gîfu, Daniela, 2010). The political actors themselves introduce a specific language based on sarcasm or irony, making their message analysis very challenging (Reyes, Antonio and Rosso, Paolo and Buscaldi, Davide, 2012). In order to identify the figurative meaning of a specific message, it is required to encode each sentence separately. The research question guiding this paper is what are the most efficient sarcasm detection algorithms? We propose an approach based on three classical machine learning (ML) approaches by trying to train the models based on this dataset solely (i.e., no other datasets have been used). Furthermore, we experimented with architectures ranging from Naïve Bayes (multinomial, complement and bernoulli variants), Support Vector Machines (SVMs with linear, polynomial and RBF kernels) to Logistic Regression which we tried to train using only the dataset provided by the SemEval-2022 Task 6 competition (Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy, 2022). The rest of the paper is organized as follows: section 2 briefly presents studies related to sarcasm detection, section 3 presents the dataset, the required pre-processing and plausible methods for it, section 4 resumes the results of the conducted experiments, with their interpretations, followed by section 5 with the conclusions.

2 Related work

This topic is a widely researched subject in recent years, evidenced in this competition at several workshops (e.g., SemEval-2017 Task 4: Using Sarcasm Detection for Enhancing Sentiment Classification and Quantification or SemEval-2018 Task 3: Irony Detection in English Tweets). Such a competition is challenging, especially since the problem of labeled data is time consuming and not cheap. Moreover, the automatic sarcasm detection depends on the annotation process, which always introduce some biases to the data. For the binary task, as in this case, there are many computational models to solve it or to capture the (actual) sarcasm or subcategories of it (John S. Leggitt and Raymond W. Gibbs Jr. , 2000) (Silviu Oprea and Walid Magdy, 2020). Thus, work on this topic was never followed by high results, as this problem is still debatable and text classification even for humans is very controversial and biased. It is a task that can be considered as sentiment analysis for which most of the authors used LSTM (Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, 2014), (Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy, 2016) or CNN (Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su, 2018) and (Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang, 2018). New approaches concentrate on using attention-based methods (Joshi et al., 2017), in particular transformer architectures such as BERT (Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina, 2018), RoBERTa, spanBERT (Amardeep Kumar, Vivek Anand, 2020) etc. These transformers are pre-trained on unlabeled data to be later fine-tuned for a variety of tasks like single sentence classification, sentence pair classification, etc. to understand role of context for sarcasm detection. Here, we used five machine learning approaches by trying to train the models based on this dataset solely (i.e., no other datasets have been used).

3 Dataset and Methods

This section is focused on two issues: Twitter dataset in English and two types of methods for classification task, both binary and multi-class. For the first type, we mostly tried classical machine learning approaches that resulted in satisfactory outcomes when evaluated on the training set. On the second one, a RNN with a few GRU layers was the architectural choice. At this moment, the results were modest, probably due to the lack of to much data (it is empirically known that neural networks require a high volume of data to work properly)

3.1 Dataset

The dataset that this competition provided consists of 3468 samples. Each sample is made of a short tweet and 8 binary columns which specify whether or not the text belongs to a certain class. The target column in the binary classification case is ”sarcastic” while the other 7 (sarcasm, irony, satire, understatement, overstatement, rhetorical_question, ambiguity) compose the outputs for the multi-class scenario. At a first glance, we can clearly see that the data is heavily unbalanced. However, this is pretty standard and conforms to the reality where, for example, only a few people are sarcastic, ironic or satirical and so on. Sure, data imputation methods could have been useful (for example, replacing the words of each text with the corresponding synonyms and adding the new texts to the dataset or doing some sort of oversampling on the positive label texts that are heavily lacking in examples) to make the set a little bit bigger but we chose to just stick to what was given! In the following picture (Figure 1), you can see a bar-chart of how the output column labels are distributed across the dataset.

One big inconvenience that we observed is the presence of many 0s in the target columns. This is difficult to solve since replacing nans with certain labels is not tractable. In the end, we went forward with the missing values. Before getting into the proper implementation details, we want to present an image of the overall architecture (Figure 2) so that every step is clear right from the start. The right arrows in the image define transitions from one step to another. A step title is written inside a cloud and the boxes under it are nothing but its definitory operations.

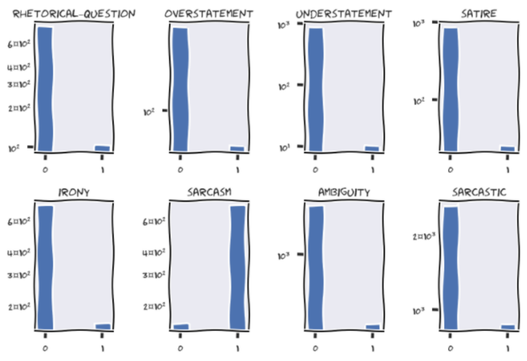


Figure 1: Bar charts of the labels distribution. The Y axis represents the counts for each value that the labels take.

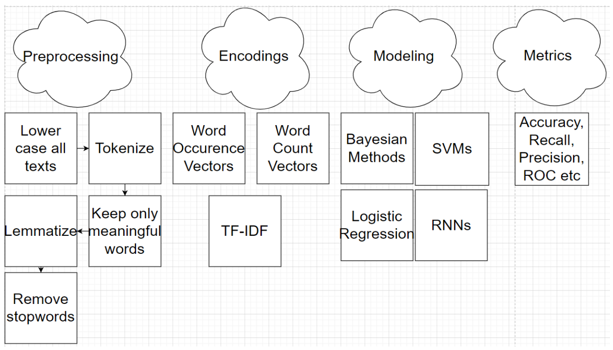


Figure 2: The architecture is made out of 4 parts. The separation between them is made through arrows.

3.2 Methods

3.2.1 Pre-processing

Before talking about the machine learning models, we will explain the pre-processing step that had to be done in order to ”clear” the data. First, we lowered all the training texts and tokenized them into words. Since many compound words could exist (e.g. ”state-of-the-art” or ”equality”; note that these are not actual words from the set) the natural thing to do is to split them by space. After that, for each sample, we are left with lists of words only, hence getting rid of the non alphanumeric words is advisable (the punctuation marks do not add any real value to our task). Another requirement is the lemmatization of the words. We want the root forms of our words in order not to consider multiple derivations of the same word as different (e.g. ”goes” should be the same with ”go”, ”going” or ”gone” and so on). Now, as a final touch, we eliminate the stopwords (the stopwords list is taken from the corpus class of the nltk library) since they don’t add any value to our models. We also built a vocabulary which contains all the unique words of the corpus and this will help us when constructing the numerical representations. Given the processed texts, we can start the well known one hot, doc-term and tf-idf encodings. For each one of them, we define a matrix where the rows represent the indexes of the texts and the columns are nothing but the words from our vocabulary. In the case of one-hot representation, one cell is either 1 or 0, noting the apppearance of a word (column) in the text (row). Alternatively, doc-term counts the number of appearances of a word in a text while tf-idf is built on top of doc-term and gives some scores according to the frequencies of the terms. All of these encodings were used with different machine learning methods.

3.2.2 Classical Machine Learning Approaches

As a first approach to classification we tried bayesian methods. We chose them because they are fast to build and have low variance. As another plus, the performance is very good with both small training sets, as well as big ones. Having a reputation as a bad estimator, but a decent classifier, we decided that it’s a good starting point (please note that the previous statements can be deducted from the books (Tom M. Mitchell, 1997), the Bayesian Learning Chapter and (Christopher Bishop, 2006), the Graphical Models chapter). Throughout time, NB has been the most used one when it comes to NLP tasks and we thought that our numerical representations could work very well on it. Bernoulli Naïve Bayes (BNB), Multinomial Naïve Bayes (MNB) and Complement Naïve Bayes (CNB) are the options we have through the scikit-learn library. BNB deals with binary features, hence one-hot encoding is the only choice here. Multinomial Naïve Bayes works with word counts or word scores, therefore doc-term and tf-idf representations are the way to go in this direction. As a last resort, we also tried a CNB because, according to the scikit documentation (Complement Naive Bayes documentation of scikit), CNB is an adaptation of the standard MNB algorithm that is particularly suited for imbalanced data sets, just like ours. The inventors of CNB show empirically that the parameter estimates for CNB are more stable than those for MNB. Further, CNB regularly outperforms MNB (often by a considerable margin) on text classification tasks (Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, David R. Karger). It is well known that logistic regression is the discriminative correspondent of Naïve Bayes. Moreover, if the conditional independence supposition of NB is true and the amount of training data tends to infinite, the 2 models should give similar results. Therefore, we tried to fit a logistic regression on our data to check the previous statement (these statements are based on the Generative and discriminative classifiers (Tom M. Mitchell) book chapter). Moving onto the next level, support vector machines (SVMs) come into play. Considering that each numerical representation has lots of features (words) and only a few lines (samples), the dual form is preferred, since the complexity is O(rows) (rows are the number of samples). Besides this, we work with the soft-margin SVM because it allows us to avoid overfitting through the slack variable. When it comes to kernels, we had 3 options. The first one we chose is the linear kernel (a.k.a the ”no kernel”) because in a very high dimensional space as ours, it's plausible to find a good separating hyperplane. After that, we decided to try a polynomial kernel as well. This one maps our data into a higher dimension, hoping to find a better suited hyperplane. The degrees used were 2 and 3 and we compared the results with the previous model. In the end, the RBF kernel seemed to be the natural step forward because it's the only one that is guaranteed to find a separating hyperplane by mapping the features into an infinite dimensional space. The hyper-parameters gamma and C (the weight of the slack variables) were chosen with a randomized grid search (no specific random seed, we just used the scikit default option) with cross validation. In other words, different values from 2 exponential distributions have been generated to each hyper-parameter and then, a SVM was applied. The one that gave the best results at cross validation was kept.

3.2.3 Deep Learning Approaches

Even though it was not the main target of the project, we chose to take a closer look at the other classes of the dataset and fit a neural network that could be the starting point of some future work. On the multi-class classification, as we've said, we fitted a RNN. However, this was not a classic RNN, since those lose to much early information. To mediate this problem, we thought that GRU hidden layers (HL) are a good option. Moreover, due to the lack of data, we had to stick to a number of 3 hidden layers only. Each one of them has 10 neurons. The output layer is, of course, a 7 neurons dense layer with softmax activation. To avoid overfitting, we also added a dropout of 10% on each HL. When it comes to the input layer, the situation is a bit different. Given that each text sample has a different number of words, some padding had to be done. Therefore, we calculated the length of the longest words sequence sample and padded the rest of the samples with 0 until we would obtain that maximum length for each training observation. The last requirement was a word2vec matrix where the indexes represent a word and the columns are the dimensions of our numerical mappings. This number of dimensions was chosen by us to be 5 because we don't have that many words. To speed up the training process, a batch normalization layer was used immediately after the input layer. The results were not the best, as we will see in the next section. We definitely needed more training data.

4 Results and Interpretations

We would like to make it clear that the results have been obtained on the training set here, since we didn't have a test set available at the time we were working on the project. Moreover, the results should be viewed with caution since the models have been trained on a small sized data set that has not to many samples, hence not to many words. A big problem arises: what do we do when the future testing samples don't contain words we've seen before? To answer this, one could simply drop the unseen words and focus on those seen only. Moving on, the main metrics we have used throughout the project are: recall, precision, F-score, balanced accuracy, accuracy and the ROC graph. It's worth noticing that the metrics have been evaluated on all the numerical representations we have mentioned before. We will start with the Naïve Bayes algorithms since these were the ones that surprisingly produced the best results (Table 1).

Bear in mind that the results have obtained on a multinomial variant with doc-term numerical representation. We can clearly see that there is a difference between the accuracy and the balanced accuracy of 8%. This is quite a big percentage, but not as big as other models resulted. Of course, the standard accuracy is not very relevant since the data is imbalanced, hence one should pay attention to the balanced accuracy more. When it comes to precision and recall on the negative labels (i.e., the not sarcastic texts), the system is very precised. Both a high sensitivity and high precision are what you would normally want from a system. In other words, the non-sarcastic texts are very well classified. On the positive labels, although not as big, the rate on recall and precision maintains at a strong above 80% percentage. Judging by the F-scores, we are pleased with the result, however, as we've said in the previous chapter, Naïve Bayes is a good classifier, but a bad estimator, so even though the results are great, the independence supposition in this case is very general and wide and tricks the statistical principles. The results for the rest of the other NB variants are given in Table 2 and Table 3.

Moving on to logistic regression, we can see that the results are really far away from NB. Therefore, we can definitely say that the Naïve Bayes independence supposition here is strongly untrue. We can see that the recall on the positive labels is very small, but the precision is almost close to 100%. In other words, among all the texts that are positive in reality, only a small percentage of them are classified as positive, but those that are predicted positive, are done so with very high certainty. So, we could say that someone who is interested in getting a very precised sarcastic prediction should choose this model. Also, an interesting observation that we can draw from all the analysed algorithm results is that a high recall on the positive class leads to a high precision on the negative class and alternatively a high precision on the positive class leads to a high recall on the negative class (Table 4).

The support vector machines are the ones that behaved the most poorly among the tried models. This is somehow weird since most of the articles out there indicate SVMs as the best option for binary classification in NLP when having little data. As mentioned before, we tried different kernel variants, but nothing good came up, unfortunately.

The polynomial kernel seems to have given the best results, but these are weak. Table 5 and Table 6 are associated to the linear and polynomial variants. Judging by the F-scores and balanced accuracy, we can easily conclude that the bayesian methods are behaving way better. What we found to be very curious is the fact that the randomized grid search didn't give any good SVM, even after 200 iterations and generous exponential distributions for "C" and "gamma" hyper-parameters. For this reason, we chose to skip showing the results for the RBF kernel. In the end, we will look at the ROC curve (Figure 9) that sums up all the models that we fitted in our project. Again, the curve with the highest area under it is the Multinomial Naïve Bayes one. The other curves just seem to overlap, firmly bellow the best one!

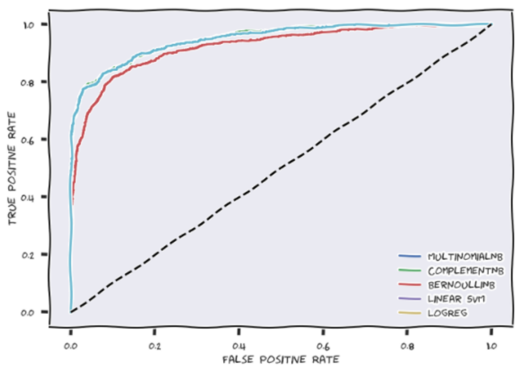


Figure 3: ROC curve

5 Conclusion

For sarcasm classification task (binary or multiclass), a small dataset is a really challenging. The classical machine learning approaches we have presented so far cannot answer convenient to it. This lack of data is a big downside because it means that not many words are numerically encoded, hence having the same vocabulary on a testing set is absolutely mandatory. In practice, it is very rarely. We may conclude that the best model is the Multinomial Naïve Bayes one, but again, it is not guaranteed that it has not overfitted the training data. In the future, some more attention could be payed to the neural network if data imputation is done and the dataset reaches a reasonable number of samples.

References

Amardeep Kumar, Vivek Anand. 2020. Transformers on Sarcasm Detection with Context. *Proceedings of the Second Workshop on Figurative Language Processing*, pages 88–92.

C. Van Hee, E. Lefever, and V. Hoste. 2018. Exploring the fine-grained analysis and automatic detection of irony on Twitter. *Lang Resources Evaluation Vol. 52, pages 707–731*.

Christopher Bishop. 2006. Pattern Recognition and Machine learning. *Springer, New York*.

Complement Naive Bayes documentation of scikit. https://scikit-learn.org/stable/modules/naive_bayes.html#complement-naive-bayes.

Dan Alexandru and Daniela Gîfu. 2020. Tracing humor in Edited News Headlines. *Ludic, Co-design and Tools Supporting Smart Learning Ecosystems and Smart Education, pages 187–196. Springer Singapore.*

Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR*.

Gîfu, Daniela. 2010. The Discourse of the Written Press and the Violence of Symbols. *PhD thesis, Faculty of Philosophy and Political Studies, "Alexandru Ioan Cuza" University of Iaşi.*

Gifu Daniela, Samson Mihai. 2021. FII FUNNY at SemEval-2021 Task 7: HaHackathon: Detecting and rating Humor and Offense. *Proceedings of the 15th International Workshop on Semantic Evaluation, (SemEval-2021), ACL, Bangkok, Thailand, pages 1226–1231, DOI: 10.18653/v1/2021.semeval-1.174.*

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics*.

Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, David R. Karger. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. Massachusetts Institute of Technology.

John S. Leggitt and Raymond W. Gibbs Jr. . 2000. Emotional reactions to verbal irony.

Reyes, Antonio and Rosso, Paolo and Buscaldi, Davide. 2012. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering*, 74:1–12.

Silviu Oprea and Walid Magdy. 2020. The effect of sociocultural variables on sarcasm communication online. *Proceedings of the 23rd ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW).*

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. *AAAI*.

Tom M. Mitchell. http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf.

Tom M. Mitchell. 1997. Machine Learning,. *McGraw Hill Education, New York.*

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in between. *ACL*, pages 1010–1020.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

	Precision	Recall	F1-Score	Support	Overall
0	0.90	0.99	0.94	2601	-
1	0.97	0.66	0.78	867	-
balanced accuracy	-	-	-	3468	0.8254
accuracy	-	-	-	3468	0.9094
macro avg	0.93	0.83	0.86	3468	-
weighted avg	0.92	0.91	0.90	3468	-

Table 1: Multinomial NB with doc-term encoding

	Precision	Recall	F1-Score	Support	Overall
0	0.89	0.99	0.94	2601	-
1	0.98	0.65	0.78	867	-
balanced accuracy	-	-	-	3468	0.8208
accuracy	-	-	-	3468	0.9077
macro avg	0.93	0.82	0.86	3468	-
weighted avg	0.91	0.91	0.90	3468	-

Table 2: Multinomial NB with one hot encoding

	Precision	Recall	F1-Score	Support	Overall
0	0.76	1.00	0.87	2601	-
1	1.00	0.07	0.13	867	-
balanced accuracy	-	-	-	3468	0.5340
accuracy	-	-	-	3468	0.7670
macro avg	0.88	0.53	0.50	3468	-
weighted avg	0.82	0.77	0.68	3468	-

Table 3: Multinomial NB with tf-idf encoding

	Precision	Recall	F1-Score	Support	Overall
0	0.96	1.00	0.98	2601	-
1	0.99	0.87	0.93	867	-
balanced accuracy	-	-	-	3468	0.9323
accuracy	-	-	-	3468	0.9653
macro avg	0.98	0.93	0.95	3468	-
weighted avg	0.97	0.97	0.96	3468	-

Table 4: Logistic Regression Results

	Precision	Recall	F1-Score	Support	Overall
0	0.85	1.00	0.92	2601	-
1	1.00	0.49	0.66	867	-
balanced accuracy	-	-	-	3468	0.7452
accuracy	-	-	-	3468	0.8722
macro avg	0.93	0.75	0.79	3468	-
weighted avg	0.89	0.87	0.86	3468	-

Table 5: SVM with linear kernel results

	Precision	**Recall**	**F1-Score**	**Support**	**Overall**
0	0.87	1.00	0.93	2601	-
1	1.00	0.54	0.70	867	-
balanced accuracy	-	-	-	3468	0.7710
accuracy	-	-	-	3468	0.8855
macro avg	0.93	0.77	0.82	3468	-
weighted avg	0.90	0.89	0.87	3468	-

Table 6: SVM with polynomial kernel results

MarSan at SemEval-2022 Task 6: iSarcasm Detection via T5 and Sequence Learners

Maryam Najafi[1,*], **Ehsan Tavan**[1,*]
[1] NLP Department, Part AI Research Center, Tehran, Iran
`{maryam.najafi, ehsan.tavan}@partdp.ai`

Abstract

The paper describes SemEval-2022's shared task "Intended Sarcasm Detection in English and Arabic." This task includes English and Arabic tweets with sarcasm and non-sarcasm samples and irony speech labels. The first two subtasks predict whether a text is sarcastic and the ironic category the sarcasm sample belongs to. The third one is to find the sarcastic sample from a sarcastic sample and its non-sarcastic paraphrase. Deep neural networks have recently achieved highly competitive performance in many tasks. Combining deep learning with language models has also resulted in acceptable accuracy. Inspired by this, we propose a novel deep learning model on top of language models. On top of T5, the architecture uses an encoder module of the transformer, followed by LSTM and attention to utilizing past and future information, concentrating on informative tokens. Due to the success of the proposed model, we used the same architecture with a few modifications to the output layer in all three subtasks.

1 Introduction

Sarcasm is a sophisticated form of expression that implicitly conveys the content of a sentence. Automated sarcasm detection focuses mainly on the lexical, syntactic, and semantic levels of text analysis Hazarika et al. (2018).

Natural language understanding(NLU), dialogue systems, and text mining can benefit from sarcasm detection. Arguably, the most challenging part of sarcasm is its rarity, infrequency, difficulty in detecting, and ambiguity in meaning. Sarcasm, for instance, can imply a negative meaning with the use of positive words. For example, "Taxes are just the best, and I cannot wait to pay more 🥰 🥺 😇" a sarcastic sentence that uses positive words but carries a negative meaning of "I dislike paying taxes." As a result, detecting sarcasm poses a challenging task due to the nature of sarcastic texts, which are influenced by several factors, such as context, region, and mentality.

A sarcasm detection algorithm goes beyond sentiment analysis, and instead of looking at sentiment in a sample, it focuses on sarcasm. The purpose of this field is to identify whether a given text is sarcastic or not.

Recently, it has been shown that neural language models trained on unstructured text can implicitly store and retrieve knowledge. Studies have revealed that the Text-To-Text Transfer Transformer (T5) architecture Raffel et al. (2019) can achieve high performance for various NLP applications. An essential step in creating NLP models is choosing an appropriate embedding vector. In this research, the T5 and Multilingual T5 (MT5) (Xue et al., 2020) Encoder module for English and Arabic was implemented, respectively. Our task is comprised of three Subtasks:

- Subtask A: Predict the sarcastic nature (sarcastic or non-sarcastic) of a sample.
- Subtask B: Determine which one of the irony speech categories the sample belongs to.
- Subtask C: Given two samples, determine which one is sarcastic.

There are insufficient instances in the dataset for this task, particularly the low number of sarcastic instances which make deep learning models unsuitable for extracting text features. Fine-tunning the language model on two large open-source datasets in English and Arabic is the key to solving this challenge. A fine-tuning step may provide the model with valuable awareness of task context. For the English task, the dataset of 4,484 tweets named iSarcasm (Oprea and Magdy, 2019) was selected, while for the Arabic task, the dataset of 10,547 tweets Farha and Magdy (2020) was chosen.

[*]Equal contribution. Listing order is random.

This research uses the t5 language model as a word embedding layer to design the sarcasm detection model. After T5, the encoder module of the transformer, the Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) layer, and scaled dot-product attention (Vaswani et al., 2017) were implemented sequentially for extracting the informative knowledge. Lastly, the model was fed an output from max-pooling. The same model has implemented all three subtasks, changing the input and output layer types.

To participate in Task 6 of SemEval-2022 Abu Farha et al. (2022), we submitted the results of 5 subtasks. The ranking of our model in subtask A ranked 7th in English and 31st[1] in Arabic. Our model placed 9th out of 22 teams in subtask B. In subtask C, our model placed 7th out of 16 teams in English and 3rd out of 13 teams in Arabic. Our code is available at GitHub[2] for researchers.

The remaining of this paper is organized as follows: Section 2 reviews related work. Section 3 describes both tasks and provided dataset. Section 4 presents the theoretical background of the proposed neural model. Implementation details are provided in Section 5, while experiments and results are presented in Section 6. Section 7 presents both quantitative and qualitative error analysis. Section 8 contains paper conclusions.

2 Background

It is pointed out in Javdan et al. (2020) that sarcasm can alter the meaning of a phrase, making opinion analysis error-prone. Subsequently, a model by BERT and aspect-based sentiment analysis is employed to address the issue. Based on the context dialogue sequence, this system can determine whether a response is sarcastic or not, with an F1-score of 0.73 on Twitter.

In Dadu and Pant (2020), as in Javdan et al. (2020), the researcher used two Reddit and Twitter datasets and applied Roberta-large to detect sarcasm in both datasets. González-Ibánez et al. (2011) investigates lexical (like uni-grams and dictionary-based) and pragmatic (like positive or negative emotions) features and compares the performance of machine learning techniques and human judges.

[1]There was an error in submitting this subtask. In the results section, we provide the true results of the proposed model for subtask A .

[2]`https://github.com/MarSanTeam/Sarcasm_Detection`

Since the meaning of sarcasm differs for each individual and may lead to misunderstandings in everyday communications, Hazarika et al. (2018) claims that user embedding can encode the stylistic and personality attributes of users, and combined with Convolutional Neural Networks (CNNs) (LeCun et al., 1999) that extracts localized information, the results are reasonable.

Kumar et al. (2020) introduce a binary classification deep learning model for sarcasm detection. Kumar et al. use Bi-LSTM and Multi-Head Attention Mechanism to obtain sentence embedding and classify input text with a softmax layer.

RoBERTa network architecture is utilized in Potamias et al. (2020) to map words onto a rich embedding space efficiently. To improve RoBERTa performance and capture temporal reliance information, use the RCNN network.

3 Task Description

Task 6 of SemEval-2022 presents a Sarcasm Detection dataset in English and Arabic.

Statistical information on the number of samples in the train, dev, and test data is shown in Table 1. Since there was no official dev set at the evaluation phase, we randomly selected 10% of the dataset as the dev data.

Dataset	train	dev	test
Subtask A (English)	3121	347	1400
Subtask A (Arabic)	2792	310	1400
Subtask B	780	87	1400
Subtask C (English)	780	87	200
Subtask C (Arabic)	670	76	200

Table 1: Statistical information of datasets

A description of the subtasks is provided in the following sections.

3.1 Subtask A

In this subtask, the model should determine the sarcastic or non-sarcastic nature of the text. Arabic and English texts can be submitted for this subtask.

Figure 1 shows the distribution of sarcastic and non-sarcastic classes in subtask A. There is an imbalance with this data, as only 25% of the samples are categorized as sarcasm, making the training process more challenging.

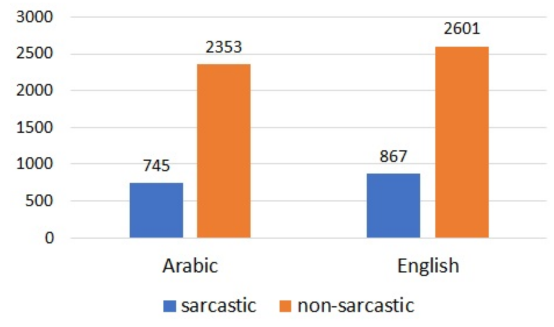


Figure 1: Label distribution in Subtask A

3.2 Subtask B

This subtask is a multi-label binary classification task.

Ideally, the model should predict which category of ironic speech (sarcasm, irony, satire, understatement, overstatement, and rhetorical question) input data falls into. In this subtask, all data is available in English only.

Figure 2 illustrates how irony speech categories (sarcasm, satire, understatement, overstatement, and rhetorical questions) have been distributed. A high percentage of samples are labeled sarcasm, and the lowest, with 10 samples, is for understatement.

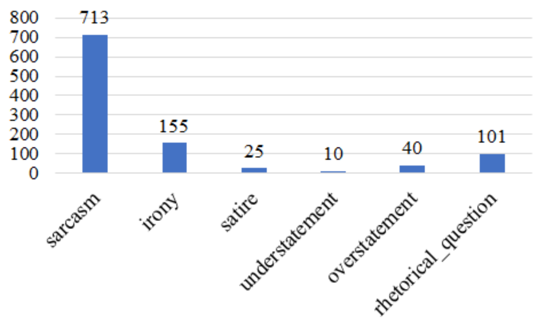


Figure 2: Label distribution in Subtask B

3.3 Subtask C

As part of subtask C, a sarcastic text and its non-sarcastic paraphrase is given to the model. The proposed model should determine which one is sarcastic.

In this Subtask, we have 867 samples in English and 745 samples in Arabic.

4 System overview

Contextual features can be extracted very efficiently with pre-trained language models. In NLP tasks, T5 has proven to be an efficient encoder-decoder framework. Using the encoder layers within the T5 language model, we can fine-tune pre-trained encoder-decoder T5 models efficiently for classification and regression tasks. Pre-trained models ease fine-tuning downstream tasks by reducing the reliance on large task-specific training datasets. Their results can be further enhanced if other deep learning architectures, such as LSTM, CNN, and attention, are applied on top of them (Tavan et al., 2021).

The framework we developed uses the encoder module of T5 and Transformer, Bi-LSTM layer, and scaled dot-product attention to determine whether a text is sarcastic. Also, we apply the same architecture to subtasks B and C but make some changes to the output and input layers, respectively. The proposed model architecture is shown in Figure3.

As part of subtasks A and B, the model gets a sequence of $S = \{s_1, s_2, s_3, ..., s_N\}$, where s_n is the nth token of input text. In subtask C, the inputs are sequence of $P = \{p_1, p_2, p_3, ..., p_I\}$, and $Q = \{q_1, q_2, q_3, ..., q_J\}$, where p_j is the jth token of the first text and q_I is the ith token of the second. The input sequence $S = \{P, < /s >, Q\}$ are the final inputs for subtask C.

Subtask A estimates a sarcastic label based on the probability distribution of $P_r(y|S)$. Within subtask B, the model estimates the probability distribution $P_r(y|S)$ for each category of ironic speech. Finally, subtask C estimates the probability distribution $P_r(y|P, Q)$, predicting whether P indicates sarcasm or Q.

4.1 Word Representation

To obtain vector representations of input tokens, the T5 encoder is used. As mentioned, this model can obtain a contextualized embedding vector for each token in the input text by fine-tuning the T5 encoder.

4.2 Encoder Architecture

The encoder module of the transformer and a Bi-LSTM layer are used sequentially on top of the T5 encoder to extract the contextualized feature.

As a way to extract the relation between tokens, the encoder architecture used in the transformer Vaswani et al. (2017) employs a multi-head attention mechanism. This capability of multi-head attention allows the model to extract relationships between input tokens, which can help identify sarcasm context.

Sarcasm detection can be enhanced by identifying contradictions and long-term dependencies in

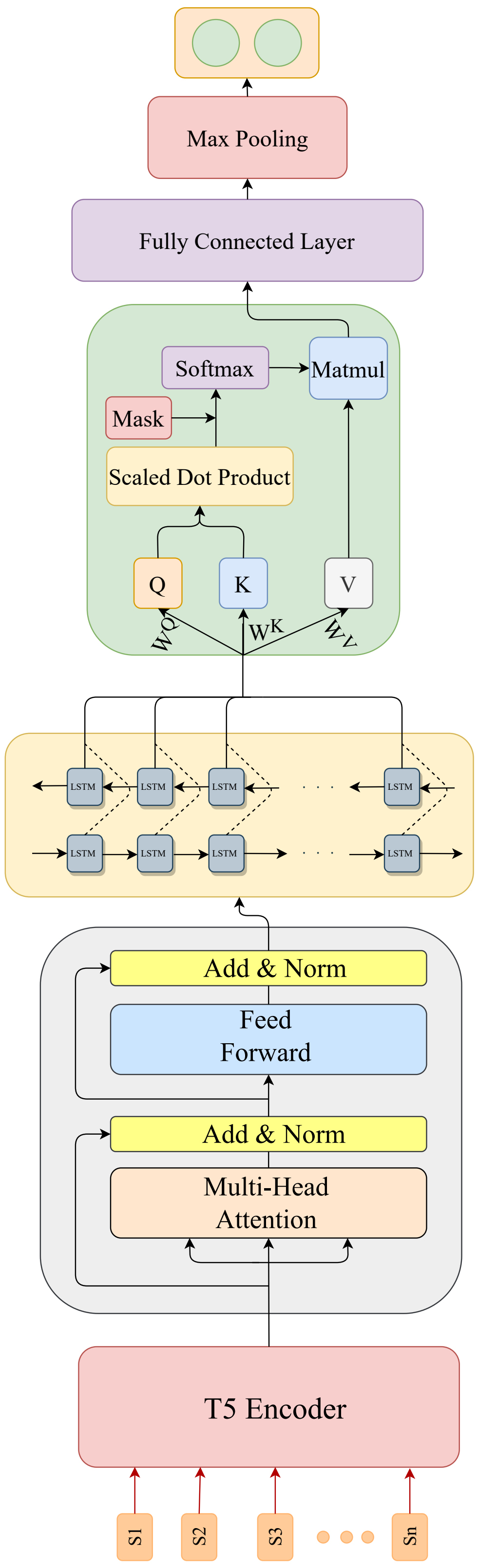


Figure 3: proposed model

a sentence. This information can not be extracted properly from the transformer encoder because of its structure. The challenge can be tackled by using the RNNs layer to extract temporal information and long-term dependencies. Using a Bi-LSTM layer, this can be done in long sequences. The output vector of the encoder layer is calculated by concatenating the Bi-LSTM layer and the transformer encoder output.

4.3 Attention Module

In Bi-LSTM networks, close words are more likely to be correlated with the extracted attribute than words located farther away. In order to extract rich features, scaled dot-product attention assigns different weights to each token. Each token is given a weight based on its importance and relation to a class. Hence, attention can determine the relevance and importance of tokens to identify the label correctly. The scaled dot-product attention above the encoder layer enables the model to capture the importance and relationship between tokens regardless of their distance. The attention module consists of the following components:

$$W_i^Q, W_i^K, W_i^V \in R^{d_{model} \times d_k}$$

$$Q = XW_Q, K = XW_K, V = XW_V \qquad (1)$$

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Among which W_Q, W_K, and W_V are trainable parameters. The input vector X is multiplied by the matrices Q, K, and V in order to create three matrices Q, K, and V. To prevent the dot-product between Q and K from getting too large, the dot-product between Q and K is divided by $\sqrt{d_k}$.

4.4 Prediction Module

To accurately predict sarcasm in a text by utilizing the most relevant and informative extracted features, a fully connected layer with a tanh activity function was first employed. The general representation is then obtained using a max-pooling layer from the same dimensions of different tokens. The max-pooling modules formulate in Equation 2.

$$Z = Max([h_1, ..., h_l]) \qquad (2)$$

Finally, to determine the probabilities of the labels, the softmax classification method is used. The module is a simple softmax classifier that generates probabilities of distributions based on input features. A softmax classifier is used to predict a label $\hat{y}$ from a set of discrete classes(sarcastic or not-sarcastic) for an input sequence S. The softmax classifier takes R as input:

$$P\left(y \mid Z\right) = softmax(WR + b) \tag{3}$$

$$\widehat{y} = argmax P\left(y \mid Z\right) \tag{4}$$

We used six softmax layers to predict the label of each input text in subtask B. The difference between the implementation of the proposed model in single label and multi-label is shown in Figure 4.

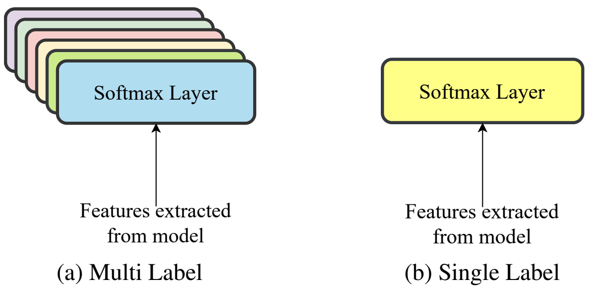


Figure 4: Outline of utput layer used in subtasks.

5 Experimental Setup

In this section, we first describe data pre-processing. Following that, we discuss the implementation details.

Pre-processing Data analysis revealed some samples containing URLs and user mentions. Due to the possibility of these items confusing the model in identifying the correct label, these items were removed in the pre-processing phase. Tokens such as punctuation and emojis were not removed since they could be excellent indicators of sarcasm.

Implementation Details PyTorch was used to implement the model, and we trained it on Nvidia V100 GPUs. Each subtask's hyper-parameters were tuned using the development set. The AdamW optimizer with the learning rate of 2e -5 is used to train the network using the back-propagation algorithm. A training method of early stopping with the patience of 5 and monitoring validation loss of sarcastic class in min mode is used. As for regularization, we evaluate the effect of the dropout on the transformer encoder and the Bi-LSTM layer; the model has a better performance when the dropout rate is set as 0.2, 0.3 respectively.

Each subtask has a batch size of 32. The number of attention heads in the transformer encoder is eight, and the hidden size in the Position-wise feed-forward Layer is 2048. One layer of Bi-LSTM with 128 LSTM units was used. Over the first fully connected layer, we applied tanh, and the output size of this layer was 256. In the case of data imbalance, the cross-entropy loss can be used with class weights. The maximum length used in T5 and MT5 tokenizers is 100. Other parameters are randomly initialized.

6 Results

This section reviews the different baselines and compares them with the proposed model. Due to the success of language models in recent years, we have evaluated different language models to select the most appropriate language model.

Since there was no official test data at the evaluation phase, experiments were conducted on the dev set to select the best architecture. Finally, the performance of the models on the test data is also evaluated.

6.1 Subtask A

Subtask A was evaluated on the F1-score of the sarcastic class. Table 2 Shows the results for subtask A. As can be seen, BERT, T5, and RoBERTa have been evaluated to determine the most appropriate language model. The T5-large and MT5-large achieve a score of 49.47% and 71.53%, respectively, outperforming other language models. It could be caused by the differences in objective learning among the models. In addition, it is worth noting that the BERT-based models predict masked words from the vocabulary and are auto-encoding models, while the T5 uses a text-to-text framework for training, and it is an auto-regressive model.

Table 3 shows the results of implementing the deep learning model. Since T5-large and MT5-large performed well in the initial experiments, they were subsequently used in subtask A. Adding transformer encoders help increase the F1-score on the sarcastic class in subtask A by capturing and focusing the most informative data on top of the language model. In English and Arabic, there was an increase in the F1-score using the transformer encoder due to the presence of multi-head attention in the transformer and the ability of this module to extract the dependencies.

Combining a Bi-LSTM layer results in a higher F1-score. This increase is due to the ability of the recurrent network to extract temporal information

Model	Dev		Test	
	sarcastic	non-sarcastic	sarcastic	non-sarcastic
Subtask A (English)				
T5-base	42.29	82.45	25.33	76.29
T5-large	**49.47**	81.05	**32.85**	75.89
BERT-base	45.89	76.55	31.40	73.40
RoBERTa-base	26.41	83.46	9.34	86.99
RoBERTa-large	39.64	85.92	25.12	88.59
Subtask A (Arabic)				
MT5-base	67.88	89.64	29.12	56.85
MT5-large	**71.53**	87.23	**31.89**	65.33
MBERT-base	58.89	76.55	24.40	73.40
XLM-RoBERTa-base	58.70	89.80	29.16	63.32
XLM-RoBERTa-large	63.93	86.52	27.87	42.67

Table 2: subtask A: Evaluation result of Language models

Model	Dev		Test	
	sarcastic	non-sarcastic	sarcastic	non-sarcastic
Subtask A (English)				
T5-large + Transformer	52.41	81.36	33.68	75.26
T5-large + Transformer + Bi-LSTM	52.55	79.49	34.85	77.89
T5-large + Transformer + Bi-LSTM + Attention	55.11	84.48	40.31	84.53
T5-large + Transformer + Bi-LSTM + Attention + fc (ours)	57.18	87.43	42.51	83.76
ours + finetune T5	**58.89**	88.39	**43.42(7)**	84.31
Subtask A (Arabic)				
MT5-large + Transformer	72.16	88.94	33.42	55.12
MT5-large + Transformer + Bi-LSTM	73.98	88.14	33.64	56.28
MT5-large + Transformer + Bi-LSTM + Attention	75.16	86.98	**34.06**	55.47
MT5-large + Transformer + Bi-LSTM + Attention + fc (ours)	75.87	90.34	31.88	53. 76
ours + finetune MT5 (Error in submission)	**76.29**	91.76	18.79(31)	34.38
ours + finetune MT5 (True result)	**76.29**	91.76	32.24	54. 43

Table 3: subtask A: Baseline results. Our rank is shown in parentheses.

and long-term dependencies. Next, adding scaled dot-product attention improved the F1-score in English and Arabic, reaching 55.11% and 75.16% on the sarcastic class, respectively. A scaled dot-product attention module could improve the model's accuracy by detecting dependencies between words, which is mainly helpful in assisting the Bi-LSTM architecture in identifying spatially spaced apart words.

Finally, as mentioned before, we have also fine-tuned T5 and MT5 on two large datasets in English and Arabic, owing to the deficient number of samples. Using this method, the F1-scores obtained in English and Arabic have reached 58.89% and 76.29%. An improvement in the F1-score can be achieved by the task awareness of the language model.

6.2 Subtask B

The results of different models for subtask B are shown in Table 4. The evaluation metric in this subtask is macro F1-score. As a result of the experiments, it was found that RoBERTa achieves a macro F1-score of 11.34% on test data, which is the highest F1-score. Due to the nature of multi-label binary classification, each of the six classes in this subtask is classified by a separate classifier within the same architecture as subtask A. Our T5-based implemented model in the competition achieved the rank of 9 and F1-score of 7.43%.

Model	Dev	Test
T5-base	12.41	2.41
T5-large	19.68	5.34
Bert-base	18.07	5.76
RoBERTa-base	22.01	**11.34**
RoBERTa-large	21.28	9.94
ours in competition	**24.67**	7.43(9)

Table 4: Subtask B: Evaluation result of Language models. Our rank is shown in parentheses.

6.3 Subtask C

Several experiments were performed to select the most appropriate language model for subtask C. Table 5 shows the results of implementing different models on English and Arabic. For this subtask, the evaluation metric is accuracy. Among the tested language models, on dev, the T5 and MT5 reach the highest accuracy among other language models.

For this reason, these models have been used in subsequent experiments. The results of language models are shown in Table 5.

Model	Dev	Test
Subtask C (English)		
T5-base	**92.24**	**72.57**
T5-large	90.95	69.43
Bert-base	77.01	68.18
RoBERTa-base	82.75	68.86
RoBERTa-large	49.42	47.27
Subtask C (Arabic)		
MT5-base	53.33	52.75
MT5-large	**88.19**	**78.64**
MBert-base	80.64	69.83
XLM-RoBERTa-base	45.12	49.73
XLM-RoBERTa-large	45.33	50.16

Table 5: Subtask C: Evaluation result of Language models

In Table 6, the model that was developed in this research is implemented, and the results are shown. To compete in English, the T5-large language model and its combination with the proposed deep network architecture have been used and we have reached an accuracy of 76.50%, 87.50% in English and Arabic, respectively.

Model	Dev	Test
Subtask C (English)		
T5-base + Transformer	93.10	73.86
T5-base + Transformer + Bi-LSTM	94.25	76.64
T5-base + Transformer + Bi-LSTM + Attention	95.34	77.13
T5-base + Transformer + Bi-LSTM + Attention + fc (ours)	92.55	**79.32**
T5-Larg + Transformer + Bi-LSTM + Attention + fc (ours in competition)	**93.10**	76.50(9)
Subtask C (Arabic)		
MT5-large + Transformer	88.99	81.38
MT5-large + Transformer + Bi-LSTM	91.25	83.18
MT5-large + Transformer + Bi-LSTM + Attention	92.34	86.32
MT5-large + Transformer + Bi-LSTM + Attention + fc (ours in competition)	**96.55**	**87.50(3)**

Table 6: Subtask C: Proposed model result. Our rank is shown in parentheses.

7 Error And Performance Analysis

In this section, we analyze the performance of several components of our system. This section will examine our model's ability to correctly label samples in sarcastic or non-sarcastic contexts.

7.1 Subtask A

There are 1400 samples in the English test set. The model correctly identified 924 samples as non-sarcasm, which means True Negative (TN), and correctly identified 132 samples as sarcasm, known as True Positive(TP). Despite this, 68 sarcasm samples were incorrectly predicted as non-sarcasm, resulting in False Negatives(FN). Lastly, the False Positive(FP) value is very high. There were 276 input samples in this case that were incorrectly predicted as sarcasm samples. According to the results, since about 75% of the data is the negative sample, Data imbalance leads to an FP error rate of about 276 among all predictions.

Error analysis Some samples of post-evaluation on the model's output were examined, in Table 7. According to studies, the most significant effect on the accurate prediction of sarcasm samples is obtained from sentiment and emoji in samples (Type A, B). According to the implemented architecture, the model is more robust for long samples. However, in very short samples (Type C), weaknesses in the estimation are observed.

In addition to analyzing the strengths of the model and the factors that affected them, this section also examined its weaknesses in predicting the sample and the factors that affected them. As mentioned, a major reason is the short length of the input sample. Another reason could be the presence of misspellings (Type D) in the data and the unfamiliarity of the model with the incorrect word. A major reason for the error in estimation can be found in the absence of sentiment and emotion in the text and the fact that the sample is completely based on "human knowledge" (Type E).

As a result, the model has weaknesses in short, humane, and emotionless examples.

7.2 Subtask B

Detailed model results on test data for each label are presented in Table 8. Since the dataset only included one sample of the "understatement" class, the model could not correctly identify the sample's label. The "overstatement" class contains only ten samples, and due to its scarcity and complexity, the model could not predict any of these, so the F1-score on the "understatement" and "overstatement" classes is 0.

Among the remaining 4 classes, the recall value is much higher than the precision of the model, which indicates that the model performed well at

Sample	Prediction type
Type A: Emoji	
Love it when someone with no mask chooses to sit next to me on the bus...😑	TP
Weathers wonderful today! 🌂🌂🌂	TP
Type B: Sentiment	
Wow the Prime Minister is so good at the telling the truth	TP
You on your best behaviour is me on my worst	TP
It makes me feel a lot safer knowing the MET Police don't investigate crimes after they happen.	TP
Yeah, feeding children sweets before bedtime is an awesome way of getting lots of sleep	TP
I swear stupid people were put on this earth to test my anger management skills	TP
Type C: Short samples	
Proof of unjustified victimisation!	FP
Rubbish	FP
Type D: Misspellings	
if you listen carefully, you can hear me not carig	FP
Type E: Human Speech	
Masks work, that's why we don't have to wear them in pubs but do in shops!	FP
Boris Johnson is a great leader and all his team stick to the covid rules rules	FP
I was waiting at the bus stop when the driver pulled up and said you waiting for a bus? I said no mate im waiting for a plane. He drove off.	FP

Table 7: subtask A: Result analysis

Class	Number of samples	TP	FP	FN	TN	Precision	Recall	F1-Score
sarcasm	180	102	748	78	472	12.00	56.67	19.81(19)
irony	20	13	365	7	1015	3.44	65.00	6.53(8)
satire	49	7	135	42	1216	4.93	14.29	7.33(5)
understatement	1	0	10	1	1389	0	0	0(3)
overstatement	10	0	203	10	1187	0	0	0(6)
rhetorical-question	11	9	145	2	1244	5.84	81.82	10.91(3)

Table 8: Subtask B: Error analysis. Our rank is shown in parentheses.

identifying samples that belong to each class. The precision value is very low, which indicates that a significant number of positive predictions are incorrectly categorized as positive.

7.3 Subtask C

Table 9 provides detailed information results for English test data. According to the three evaluation metrics, the model performed well in distinguishing sarcasm samples from non-sarcasm paraphrases.

We have decided not to explain further in this section due to the model's acceptable performance and space limitation.

Sarcasm-id	Precision	Recall	F1-Score
0	79.82	81.31	80.56
1	78.02	76.34	77.17

Table 9: Subtask C: Error analysis

8 Conclusion

We implement a novel deep learning approach based on language models. The last hidden state of T5 is used as an embedding layer in this architecture. On top of this layer, a bidirectional LSTM is used to extract future and past contexts as representations of the input text. LSTM output is processed using an attention mechanism, which focuses more on the valuable tokens to predict.

The main challenge in this study was that there were not enough samples in the dataset. To solve this, we fine-tuned the T5 language model with other large open-source datasets to have a language model that had a pre-awareness of the task and a higher accuracy. This fine-tuned language model was then used as an embedding representation in our deep architecture.

To evaluate the performance of the developed model and find the best language model, many experiments were conducted. The experimental results show that the T5 language model covers a reasonable range of results and is most appropriate for our architecture.

The same architecture was used in all three subtasks, as mentioned earlier. Due to the multi-label nature of subtask B, six separate classifiers were used instead of one to produce the output. We identified specific sarcasm challenges through error analysis, creating immediate future tasks.

As a final point, our architecture has already been created for English and Arabic, but it could be easily extended to other languages.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tanvi Dadu and Kartikey Pant. 2020. Sarcasm detection using context separators in online discourse. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 51–55.

Ibrahim Abu Farha and Walid Magdy. 2020. From arabic sentiment analysis to sarcasm detection: The arsarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 32–39.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. *arXiv preprint arXiv:1805.06413*.

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

Soroush Javdan, Behrouz Minaei-Bidgoli, et al. 2020. Applying transformers and aspect-based sentiment analysis approaches on sarcasm detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 67–71.

Avinash Kumar, Vishnu Teja Narapareddy, Veerubhotla Aditya Srikanth, Aruna Malapati, and Lalita Bhanu Murthy Neti. 2020. Sarcasm detection using multi-head attention based bidirectional lstm. *Ieee Access*, 8:6388–6397.

Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. 1999. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer.

Silviu Oprea and Walid Magdy. 2019. isarcasm: A dataset of intended sarcasm. *arXiv preprint arXiv:1911.03123*.

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas-Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Ehsan Tavan, Ali Rahmati, Maryam Najafi, and Saeed Bibak. 2021. Bert-dre: Bert with deep recursive encoder for natural language sentence matching. *arXiv preprint arXiv:2111.02188*.

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

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

LT3 at SemEval-2022 Task 6: Fuzzy-Rough Nearest Neighbor Classification for Sarcasm Detection

Olha Kaminska
Computational Web Intelligence
Department of Applied Mathematics,
Computer Science and Statistics
Ghent University, Ghent, Belgium

Chris Cornelis
Computational Web Intelligence
Department of Applied Mathematics,
Computer Science and Statistics
Ghent University, Ghent, Belgium

Veronique Hoste
LT3 Language and Translation Technology Team
Ghent University, Ghent, Belgium
{Olha.Kaminska,Chris.Cornelis,Veronique.Hoste}@UGent.be

Abstract

This paper describes the approach developed by the LT3 team in the Intended Sarcasm Detection task at SemEval-2022 Task 6. We considered the binary classification subtask A for English data. The presented system is based on the fuzzy-rough nearest neighbor classification method using various text embedding techniques. Our solution reached 9th place in the official leader-board for English subtask A.

1 Introduction

Sarcasm (or irony) can be defined as a trope or figurative language use whose actual meaning is different from what is literally enunciated (Chandler and Munday, 2011). The task of sarcasm detection can be connected with various challenges in the Natural Language Processing (NLP) field, from sentiment analysis to hate speech detection. However, this task is more complicated by its nature. Even for a human, sarcasm detection could be a challenging issue. It can be represented in different shapes, with voice, gestures, mimic, etc. So, text alone may not be sufficient to detect whether a given utterance is sarcastic or not. It makes the labeling of such datasets quite complicated (Ghanem et al., 2020).

The SemEval competition is an annual event that provides a set of challenges for researchers in different aspects of the NLP field. This year we participated in SemEval-2022 Task 6 called "iSarcasmEval" that considers sarcasm detection in two languages: English and Arabic (Abu Farha et al., 2022). We tackled subtask A for English, where for a given text, we should determine whether it is sarcastic. As described by the authors of the dataset, text writers provided the labels by themselves to exclude subjective labeling. The dataset contains text, its binary label for subtask A (sarcastic or not), a non-ironical rephrase of the provided text with an explanation of why is it sarcastic for subtask C (we did not use it in our experiments), and a set of binary classes for six types of sarcasm (irony, satire, rhetorical question, etc.) for subtask B. We note that the size of the non-irony class for subtask A is three times bigger than the size of the irony class (2,601 instances and 867 instances).

A task related to sarcasm detection is emotion detection issue, which we considered in our previous papers. In (Kaminska et al., 2021b) and (Kaminska et al., 2021a), we addressed the intensity task provided by SemEval-2018 Task 1 (Mohammad et al., 2018). Our first paper used the weighted k Nearest Neighbor (wkNN) classification approach with corresponding text cleaning and embedding steps. In the second paper, we tuned the preprocessing steps and, instead of wkNN, we considered the Fuzzy-Rough Nearest Neighbor (FRNN) classification model with Ordered Weighted Average (OWA) operators ((Jensen and Cornelis, 2011), (Vluymans et al., 2019), (Lenz et al., 2019)). The final approach has the shape of an ensemble of FRNN models based on several strong embedding techniques. The solution described in this paper is based on the methodology presented in (Kaminska et al., 2021a), but it is fine-tuned for the iSarcasmEval dataset and task. Our code is provided at the GitHub repository[1].

The remainder of this paper has the following parts: in Section 2 we provide a step-by-step description of our system, including text preprocessing and the used embedding methods, the classification model and its ensemble, and the evaluation metric. In Section 3 we provide results obtained by cross-validation, and identify the best setup ap-

[1]https://github.com/olha-kaminska/frnn_emotion_detection/tree/iSarcasmEval

plied on the test data. It also provides an error analysis of the output and post-competition experiments for model improvement. Section 4 presents conclusions and future steps.

2 System Description

Figure 1 gives an overview of our full method, which encompasses the following steps: we embed the raw text with the Twitter-roBERTa-base for Irony Detection model (Barbieri et al., 2020), then we use the output vectors in the OWA-FRNN classification method to obtain a prediction of classes.

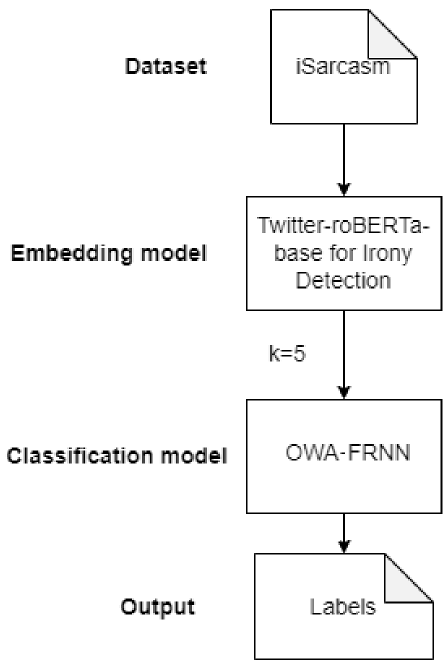


Figure 1: Schematical overview of our architecture.

2.1 Text preprocessing

We manually explored part of the provided dataset and concluded that it has attributes of regular social media posts, like tweets, including user tags, emojis, hashtags, etc. Hence, the first step in our experiments was text preprocessing before we applied text embedding techniques. Initially, we considered three options: no preprocessing, basic cleaning, and extra stop-words removal.

The basic cleaning included deleting the "#" symbol before hashtags and emojis transformation. We did not delete the text of hashtags because it could contain important information about the whole text. Similarly, we kept emojis but replaced them with textual descriptions provided by *"emoji"* package[2].

The third approach of text preprocessing involves the same steps, additionally deleting the stop-words using the *"NLTK"* package[3].

We tried all three text preprocessing setups for each text embedding technique in order to detect the most suitable for each.

2.2 Embedding methods

As a next step of text preparation before classification we investigate different text embedding methods. This technique represents a fragment of text (symbol, word, collocation, sentence, or even paragraph) as a vector (or a set of them). The obtained vector corresponds to the actual text in multi-dimensional vector space with the idea that neighboring vectors represent similar text pieces.

Embedding techniques came a long way from simple bag-of-words and pre-trained dictionaries in a word-vector format to the current state-of-the-art transformer-based solutions and context-based language models. In our method, we explored various types of text embedding techniques: vocabulary Word2Vec from Gensim package[4], sentiment-based DeepMoji[5], Universal Sentence Encoder (USE) by the TensorFlow[6], Bidirectional Encoder Representations from Transformers (BERT), as proposed by (Devlin et al., 2019) and two methods related to BERT - Sentence-BERT (SBERT) by (Reimers and Gurevych, 2019) and the Twitter-roBERTa-based model for irony recognition presented by (Barbieri et al., 2020). As we will see in Section 3.1, the latter performed much better than the others, hence we will describe it in more detail.

The robustly optimized BERT pre-training approach (roBERTa) is comparable to the original BERT model but has a few training technique and architecture differences. In (Barbieri et al., 2020), the authors described several roBERTa-based models for different tasks, for example, hate speech recognition and emotion detection. We considered the one for irony classification[7] that was trained on nearly 58M tweets and fine-tuned on the dataset from Subtask A of the SemEval2018 challenge for Irony Detection presented by (Van Hee et al., 2018).

[2]https://pypi.org/project/emoji/

[3]https://pypi.org/project/nltk/

[4]https://radimrehurek.com/gensim/models/word2vec.html

[5]https://deepmoji.mit.edu/

[6]https://www.tensorflow.org/hub/tutorials/semantic_similarity_with_tf_hub_universal_encoder

[7]https://huggingface.co/cardiffnlp/twitter-roberta-base-irony

This model can perform classification directly, but instead we used it to extract tweets' weights from the inner model's layer as their embedding vectors. We will discuss this further in Section 2.3.

2.3 OWA-FRNN classifier

Most state-of-the-art approaches for tasks such as irony detection belong to the deep learning family, and therefore remain black-box solutions. Our idea was to try a more explainable technique based on the Fuzzy-Rough Nearest Neighbor (FRNN) method that already showed promising results in our previous work (Kaminska et al., 2021a).

The FRNN classification model was introduced in (Jensen and Cornelis, 2011). It is an instance-based approach that uses lower (L) and upper (U) fuzzy-rough approximations inside the classification process. In (Vluymans et al., 2019) and (Lenz et al., 2019), FRNN extensions were described based on the Ordered Weighted Average (OWA) operators with the aim of making the method more robust. OWA operators are used to define the membership of a data instance to the lower and upper approximation through an aggregation process.

We will use the following notation: V is the set of OWA aggregation values, where $v_{(i)}$ is the i^{th} largest element of the set V; the weight vector is denoted as $\overrightarrow{W} = \langle w_1, w_2, ..., w_{|V|}\rangle$, where $(\forall i)(w_i \in [0,1])$ and $\sum_{i=1}^{|V|} w_i = 1$. Then, we will have the following formula for the OWA operator:

$$OWA_{\overrightarrow{W}}(V) = \sum_{i=1}^{|V|}(w_i v_{(i)}) \quad (1)$$

We used additive OWA operators (Vluymans et al., 2019), as they performed the best in our previous paper. They are linearly increasing for lower and linearly decreasing for upper approximations. Additive weights are presented by the Formulas (2) and (3), where p denotes the length of the vector ($p > 1$).

$$\overrightarrow{W}_L^{add} = \langle \frac{2}{p(p+1)}, \frac{4}{p(p+1)}, ..., \frac{2(p-1)}{p(p+1)}, \frac{2}{p+1} \rangle \quad (2)$$

$$\overrightarrow{W}_U^{add} = \langle \frac{2}{p+1}, \frac{2(p-1)}{p(p+1)}, ..., \frac{4}{p(p+1)}, \frac{2}{p(p+1)} \rangle \quad (3)$$

OWA-FRNN assigns a test instance y to the class C with the highest sum of $\underline{C}(y)$ and $\overline{C}(y)$:

$$\underline{C}(y) = OWA_{\overrightarrow{W}_L}\{1 - R(x,y) \mid x \in X \setminus C\} \quad (4)$$

$$\overline{C}(y) = OWA_{\overrightarrow{W}_U}\{R(x,y) \mid x \in C\} \quad (5)$$

Here, $R(x,y)$ corresponds to the similarity between vectors x and y. In our experiments, we used cosine similarity:

$$cos\_similarity(A,B) = \frac{1 + cos(A,B)}{2}. \quad (6)$$

Where $cos(A,B)$ is the cosine distance between elements A and B:

$$\cos(A,B) = \frac{A \cdot B}{||A|| \times ||B||} \quad (7)$$

For our setup, A and B denote tweet embedding vectors, $A \cdot B$ is their scalar product, and $||A||$ is the vector norm of A. We considered similarity instead of distance because Formula (6) provides values that fit our classification method: 0 for opposite vectors and 1 for identical ones.

One more parameter that we need for Formulas (4) and (5) is k - the number of nearest neighbors of test instance y. The difference between k in Formulas (4) and (5) is that for the first, it corresponds to the amount of training samples that are y's neighbors outside class C and for the second - those inside class C. The parameter k is used to limit the calculations and, just as for wkNN, there are no general rules on how to choose it. Hence, we will tune this parameter for each model separately.

We used the OWA-FRNN approach as the primary classification technique, with a Python implementation[8] provided by (Lenz et al., 2020).

2.4 Ensembles

To improve our results, we considered the usage of models' ensembles. The ensemble combines several classification models' outputs to provide the final prediction. The idea behind it is to improve the performance of a single model by combining it with other models to fuse their advantages.

We tuned the best setup for each embedding method separately (with parameters as text preprocessing and a number of neighbors k) and then united them in an ensemble. As a combination method for models outputs, or in other words, "a

[8] `https://github.com/oulenz/fuzzy-rough-learn`

voting function", we used the mean. We tuned the set of the models used in the ensemble to prevent weak models from decreasing the final score. All obtained results are presented in Section 3.

2.5 Evaluation

To compare different approaches on the train data, we used a 5-fold cross-validation technique by splitting the provided dataset into train and test data with an 80/20 ratio.

As an evaluation metric for predicted labels and actual labels, we used the F1-score (Formula (8)) calculated for the sarcastic class, as was suggested by the competition organizers.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}, \qquad (8)$$

where by $Precision$ we mean the fraction of accurately predicted sarcastic labels from all predicted test labels, and by $Recall$ we refer to the fraction of accurately predicted sarcastic labels from all actual sarcastic test labels. The highest value of the F1-score corresponds to the best model.

3 Results

This section consists of several parts. In Subsection 3.1 we provide an overview of our experiments, using cross-validation on the train data to identify the best model for irony detection and evaluate its result for test data. In Subsection 3.2, we describe error analysis and illustrate the explainability of our approach. We also performed additional experiments after the competition was finished and labels for the test data released to improve our scores and present this process in Subsection 3.3.

3.1 The best setup

Firstly, we evaluated the OWA-FRNN classification model on each embedding method separately by tuning the number of neighbors k and text preprocessing techniques. The best setup for each embedding method with the best parameters and corresponding F1-score for the sarcastic class is shown in Table 1.

Table 1 lists all methods in decreasing order by F1-score. The best result was obtained with the roBERTa-based model with a noticeable gap for the next method - DeepMoji. It can be seen that for almost all embedding techniques, the value of k is equal to 5, whereas for text preprocessing, no particular pattern was observed.

Table 1: Cross-validation best F1-scores for different embedding methods and corresponding setups.

Method	**Text**	**k**	**F1 sarcastic**
roBERTa	raw	5	**0.3722**
DeepMoji	cleaned	5	0.3157
USE	raw	5	0.2808
BERT	cleaned, no stop-words	5	0.2351
Word2Vec	cleaned, no stop-words	5	0.2050
SBERT	raw	7	0.1618

Table 2: Cross-validation F1-scores for ensembles of embedding methods.

Embeddings	**F1-score**
All six	0.0995
TOP-5	0.1866
TOP-4	0.1317
TOP-3	0.2941
TOP-2	0.1866

Secondly, we combined different embedding methods with their best setups as an ensemble. We calculated the mean for all embedding methods, then for the top-5 (excluding the weakest one - SBERT), top-4 (excluding Word2Vec and SBERT), and so on, until the single top model roBERTa is left. The results are presented in Table 2.

From Table 2 we can see that ensembles provided lower scores than single models. The standalone roBERTa performed better than in ensembles with others, which could already be expected from Table 1, where the gap between the roBERTa method and the rest is remarkable.

Hence, we can conclude that the best setup has the following components: no text preprocessing, roBERTa-based embedding technique for vectors extraction, and an OWA-FRNN classification model with a number of neighbors $k = 5$. This setup was applied to the test dataset.

We calculated labels for the test data using this setup and submitted them to the competition to obtain **F1-score = 0.4242** for the sarcastic class, leading to a **9th place** in the leader-board. Meanwhile, we received higher places for some other metrics: 7th place for averaged F-score = 0.6552 and precision = 0.6422, and 8th place for accuracy = 0.8100.

3.2 Error analysis

For the error analysis, we could rely on the fact that the OWA-FRNN classifier we experimented with has the advantage of being more explainable compared to many other black box approaches. In our case, we mean that we can trace back the test instance and see which five instances from the train data determined its class.

Initially, we traced back several correctly predicted test tweets to see if it was possible to notice any patterns. For example, for the sarcastic test tweet *"So the Scottish Government want people to get their booster shots so badly that the website doesn't even work"*, we got four sarcastic training neighbors out of five. Four neighbors were connected to the health topic and contained collocations such as *"mental health"*, *"health insurance"*, *"covid vaccine"*, and *"healthcare"*. The fifth neighbor was about emails that could be connected to *"website"* word from the test tweet. From this sample, we could conclude that having a common topic is an important feature for neighbors detection and our model deals well with it, as we also noticed from exploring other test samples.

As for wrong predictions, we also found an illustrative example for the sarcastic test tweet: *"Sometimes I lay in bed and think about how today will be the day I make my life better. Exercise, drinking water, eating healthy. Then I wake up."* It has four training neighbors about daily routine and lifestyle with mostly non-sarcastic labels, leading to the wrong prediction. For example, the closest training neighbor *"me: I'm gonna wash my hair and shave my legs! Me instead: I'm gonna dissociate in the shower for 45 minutes"* looks pretty similar to the test sample but has a non-sarcastic label. Here, we could highlight again the difficulty of sarcasm dataset labeling and how subjective it could be.

In general, we can see that topic could be a strong feature. However, the same concept could have different meanings in different topics (for example, *"temperature"* in weather or fever). Also, some neighbors have the same emojis as a test instance that can give a hint about emojis importance.

3.3 Model improvements

After the test labels were released, we experimented with more setups to improve our final F1-score.

First, we used other values of the parameter k that showed mediocre results on cross-validation to see how they perform on the test data. For example, we observed that for $k = 17$, we receive an F1-score with cross-validation equal to 0.3408, which is lower than our best setup. However, on the test data, this value of k provided us an F1-score equal to 0.5, which is more than what we got in the leader-board and would lead us to fourth place.

Secondly, we considered the usage of the weighted k Nearest Neighbors (wkNN) algorithm (Dudani, 1976). This approach is close to OWA-FRNN, and we already worked with it in our previous paper (Kaminska et al., 2021b) for the emotion detection task. The wkNN works with k closest neighbors and puts weights based not on the OWA operator but on the neighbors' distances. As a similarity function, we used cosine again.

To test the wkNN method, we applied it inside our best setup - roBERTa-based embedding vectors obtained from the raw tweets and $k = 5$. We got an F1-score for the sarcastic class of 0.3569 with cross-validation and 0.4299 for the test data. We can see a minor improvement for the test data, compared to our final scores from the leader-board.We also checked this setup for $k = 17$ and obtained a sarcastic F1-score for cross-validation equal to 0.2790 and 0.4969 for the test data. It would also top us up to the fourth place, and so in general, we can see that results for the OWA-FRNN and the wkNN methods in our setups are pretty close.

4 Conclusion & Future Work

In this paper, we presented our model for the iSarcasmEval competition. Our solution uses the instance-based classification method OWA-FRNN and a roBERTa-based model for Irony Recognition as an embedding method. We fine-tuned the best setup on the train data with cross-validation and obtained the ninth place on the test data in the competition leader-board.

Our approach is explainable in a way that we can trace back the test instance and find the training instances that determined the predicted class to explore some patterns. For example, we observe the significance of the tweet topic and even of particular keywords.

In the future, the provided solution may be improved by additional text preprocessing techniques or roBERTa-based model fine-tuning using additional datasets.

Acknowledgment

This work was supported by the Odysseus programme of the Research Foundation-Flanders (FWO).

References

Ibrahim Abu Farha, Silviu Oprea, Steve Wilson, and Walid Magdy. 2022. Semeval 2022: isarcasmeval - intended sarcasm detection in english and arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

Daniel Chandler and Rod Munday. 2011. *A dictionary of media and communication*. OUP Oxford.

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

Sahibsingh A Dudani. 1976. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, (4):325–327.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Paolo Rosso, and Véronique Moriceau. 2020. Irony detection in a multilingual context. *Advances in Information Retrieval*, 12036:141.

Richard Jensen and Chris Cornelis. 2011. Fuzzy-rough nearest neighbour classification and prediction. *Theoretical Computer Science*, 412(42):5871–5884.

Olha Kaminska, Chris Cornelis, and Veronique Hoste. 2021a. Fuzzy-rough nearest neighbour approaches for emotion detection in tweets.

Olha Kaminska, Chris Cornelis, and Veronique Hoste. 2021b. Nearest neighbour approaches for emotion detection in tweets. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 203–212.

Oliver Urs Lenz, Daniel Peralta, and Chris Cornelis. 2019. Scalable approximate frnn-owa classification. *IEEE Transactions on Fuzzy Systems*, 28(5):929–938.

Oliver Urs Lenz, Daniel Peralta, and Chris Cornelis. 2020. fuzzy-rough-learn 0.1: a Python library for machine learning with fuzzy rough sets. In *IJCRS 2020: Proceedings of the International Joint Conference on Rough Sets*, volume 12179 of *Lecture Notes in Artificial Intelligence*, pages 491–499. Springer.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

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

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Sarah Vluymans, Neil Mac Parthaláin, Chris Cornelis, and Yvan Saeys. 2019. Weight selection strategies for ordered weighted average based fuzzy rough sets. *Information Sciences*, 501:155–171.

LISACTeam at SemEval-2022 Task 6: A Transformer based Approach for Intended Sarcasm Detection in English Tweets

Abdessamad Benlahbib[1*], **Hamza Alami**[2*], **Ahmed Alami**[3]
[1] LISAC Laboratory, Faculty of Sciences Dhar EL Mehraz, USMBA, Fez, Morocco
[2] Innov-Tech Laboratory, Department of Engineering, High Technology School, Rabat
[3] Laboratory of Engineering Sciences, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco
abdessamad.benlahbib@usmba.ac.ma , hamza0alami@gmail.com
alami.alami1996@gmail.com

Abstract

In this paper, we present our system and findings for SemEval-2022 Task 6 - iSarcasmEval: Intended Sarcasm Detection in English. The main objective of this task was to identify sarcastic tweets. This task was challenging mainly due to (1) the small training dataset that contains only 3468 tweets and (2) the imbalanced class distribution (25% sarcastic and 75% non-sarcastic). Our submitted model (ranked eighth on Sub-Task A and fifth on Sub-Task C) consists of a Transformer-based approach (BERTweet model).

1 Introduction

The Cambridge Dictionary [1] defines sarcasm as *"the use of remarks that clearly mean the opposite of what they say, made in order to hurt someone's feelings or to criticize something in a humorous way."*

Due to the Web openness, sarcastic content becomes very frequent in social media and e-commerce platforms, which may cause misunderstandings. Furthermore, identifying such content is a very challenging task even for humans (Farias and Rosso, 2017). Also, it could impact some natural language processing tasks such as sentiment analysis (Farias and Rosso, 2017; Do et al., 2019; Tubishat et al., 2018; Balazs and Velásquez, 2016; Maynard and Greenwood, 2014; Ptáček et al., 2014; Bouazizi and Otsuki Ohtsuki, 2016; Ren et al., 2018).

We introduce the following example: *"The movie was enjoyable to the point that I clapped because it is finished."* For an opinion mining system, this sentence could be considered positive. However, the author expresses a negative judgment against the movie since the expression *"I clapped because it is finished"* means that it was boring.

For this reason, SemEval 2022 set up Task 6: iSarcasmEval - Intended Sarcasm Detection in English and Arabic to detect sarcastic and non-sarcastic tweets (Abu Farha et al., 2022). Our submitted system consisted of a pre-trained transformer model for English Tweets named BERTweet (Nguyen et al., 2020), secured 8th and 5th positions respectively on Sub-Task A and Sub-Task C leaderboard.

The rest of the paper is structured in the following manner: Section 2 provides the data structure and the main objective of each Sub-Task. Section 3 describes our system. Section 4 details the experiments. And finally, Section 5 concludes this paper.

2 Task Description

The organizers of this task introduced two tweet datasets for both English and Arabic languages that contain:

- a label specifying whether a tweet is sarcastic or non-sarcastic, provided by its author.
- a non-sarcastic rephrase of a sarcastic tweet provided by its author.
- a label specifying the category of ironic speech that it reflects, provided by a linguistic expert (English only).
- a label specifying the dialect (Arabic only).

This task consists mainly of three sub-tasks for the English dataset and two sub-tasks for the Arabic dataset where Sub-Task A aims at determining whether a tweet is sarcastic or non-sarcastic, Sub-Task B, which is available for English only, is a binary multi-label classification task that intends to determine which ironic speech category a sarcastic tweet belongs to if any, and finally, Sub-Task C that takes two inputs: a sarcastic tweet and its non-sarcastic rephrase, and focuses on identifying the sarcastic one between them.

[*] contributed equally
[1] https://dictionary.cambridge.org/fr/dictionnaire/anglais/sarcasm

3 System Description

In this section, we describe our proposed system that tackles Sub-Task A and Sub-Task C English.

3.1 Sub-Task A

In order to tackle Sub-Task A, we adopted a transformer-based (Vaswani et al., 2017) approach that consists of fine-tuning BERTweet [2], which is a language model pre-trained on 850M English Tweets, and it has the same architecture as BERT-base (Devlin et al., 2019), as well as it is was trained using the RoBERTa pre-training procedure (Liu et al., 2019).

Before feeding the training data to BERTweet model, we preprocessed them by removing URLs and then replacing emojis with their English textual meaning (Alami et al., 2020) using BERTweet demojizer [3]. Figure 1 depicts the Tweets preprocessing pipeline.

After the preprocessing phase, we fine-tuned BERTweet model on the training dataset that contains 3468 tweets (867 sarcastic tweets and 2601 non-sarcastic tweets).

3.2 Sub-Task C

The same model of Sub-Task A was used to handle Sub-Task C by feeding two texts to the BERTWeet model that was already fine-tuned on the training dataset. The text with the highest probability of being sarcastic is considered the sarcastic one.

4 Experimental Results

We experimented our model on the SemEval 2022 Task 6: iSarcasmEval - Intended Sarcasm Detection in English Sub-Task A and Sub-Task C datasets. All our experiments have been conducted in Google Colab environment[4], The following libraries: Transformers - Hugging Face[5] (Wolf et al., 2020), Scikit-Learn[6] (Pedregosa et al., 2011), and Keras[7] were used to train and to asses the performance of our model.

[2]https://github.com/VinAIResearch/BERTweet

[3]https://huggingface.co/transformers/v4.4.2/_modules/transformers/models/bertweet/tokenization_bertweet.html

[4]https://colab.research.google.com/

[5]https://huggingface.co/docs/transformers/index

[6]https://scikit-learn.org/stable/

[7]https://keras.io/

4.1 Datasets

Since we have participated in Sub-Task A and Sub-Task C for English, we will only describe the English dataset. The training set contains 867 sarcastic tweets and 2601 non-sarcastic tweets, the test set of Sub-Task A contains 200 sarcastic tweets and 1200 non-sarcastic tweets, and the test set of Sub-Task C contains 200 sarcastic tweets and their rephrases. Figure 2 depicts the class distribution of the English tweets in the training and test set for Sub-Task A.

4.2 Evaluation Metric

To evaluate the performance of the submitted results, the organizers adopted the F1-score for the sarcastic class as the main metric for Sub-Task A as well as the accuracy for Sub-Task C. The F1-score and accuracy are computed in the following manner where $P_{sarcastic}$ and $R_{sarcastic}$ are respectively the precision and recall of the sarcastic class, and TP, TN, FP and FN are respectively the true positive, true negative, false positive and false negative.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$P_{sarcastic} = \frac{TP_{sarcastic}}{TP_{sarcastic} + FP_{sarcastic}} \quad (2)$$

$$R_{sarcastic} = \frac{TP_{sarcastic}}{TP_{sarcastic} + FN_{sarcastic}} \quad (3)$$

$$F1_{sarcastic} = \frac{2 \times P_{sarcastic} \times R_{sarcastic}}{P_{sarcastic} + R_{sarcastic}} \quad (4)$$

4.3 Experimental Settings

During the fine-tuning of BERTWeet model, we set the hyper-parameters as follows: 10^{-5} as the learning rate, 15 epochs, 128 as the max sequence length, and 32 as batch size. The same settings were adopted for DistilBERT (Sanh et al., 2019) and BERT base uncased. Table 1 summarizes the hyperparameters settings of BERTWeet model.

For the Bidirectional Long Short-Term Memory (Bi-LSTM) (Hochreiter and Schmidhuber, 1997) and Bidirectional Gated Recurrent Unit (Bi-GRU) (Cho et al., 2014), we set 10 epochs, 128 as the max sequence length, and 16 as batch size.

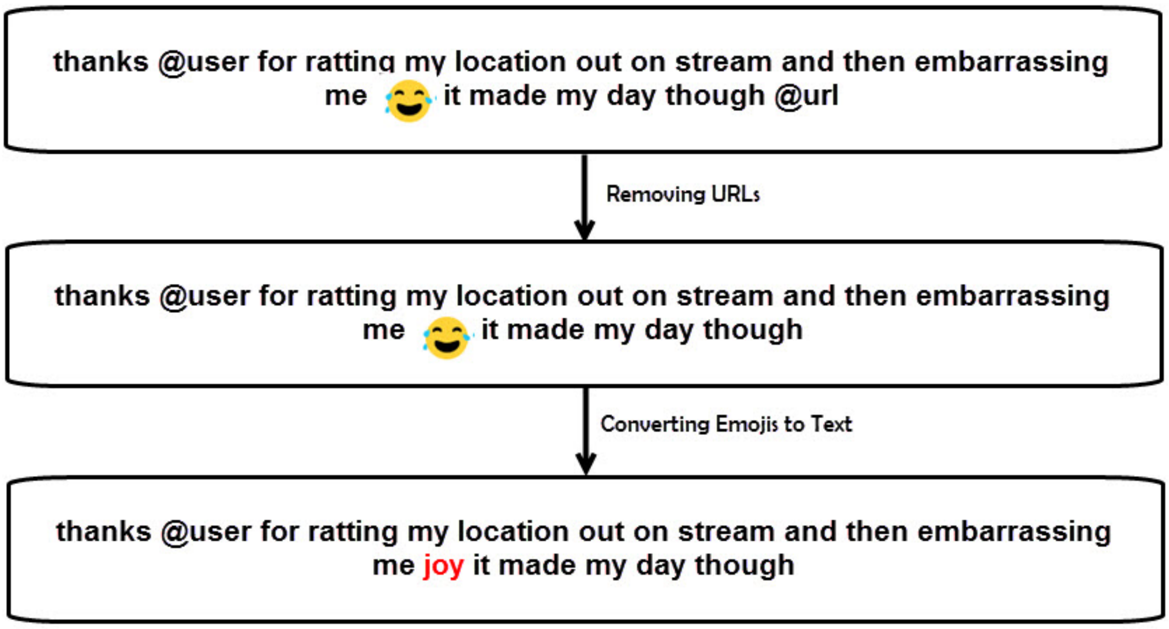

thanks @user for ratting my location out on stream and then embarrassing me 😂 it made my day though @url
Removing URLs
thanks @user for ratting my location out on stream and then embarrassing me 😂 it made my day though
Converting Emojis to Text
thanks @user for ratting my location out on stream and then embarrassing me joy it made my day though

Figure 1: Tweets preprocessing

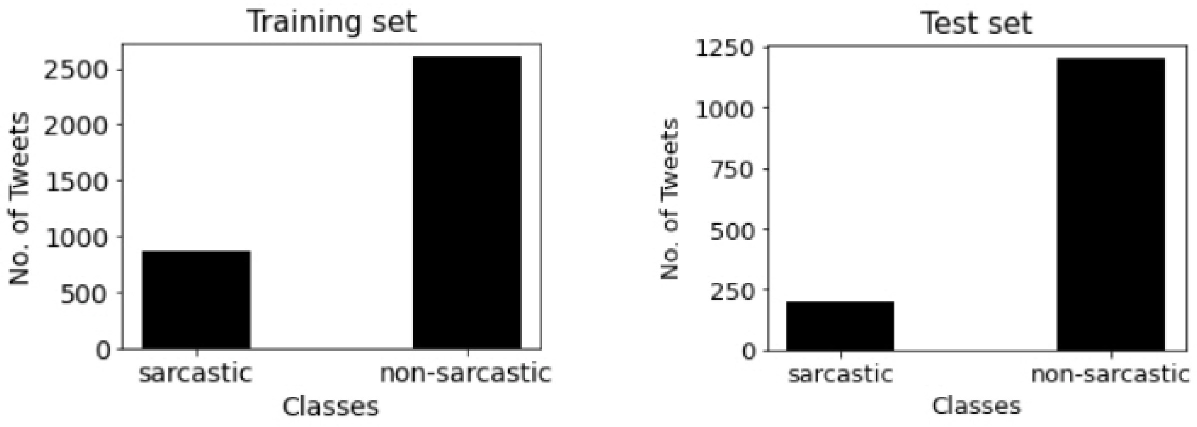


Figure 2: Class distribution of the English tweets in the training and test set for Sub-Task A

Hyperparameters	Settings
Learning rate	10^{-5}
Batch size	32
Epochs	15
Max sequence length	128
Optimizer	Adam (Kingma and Ba, 2015)
Loss	Cross-Entropy

Table 1: Hyperparameters settings for BERTWeet model in the experiments

4.4 System Performance

We evaluated various models on Sub-Task A test set including Linear Support Vector Classification (LinearSVC) (Boser et al., 1992), Logistic Regression, Multinomial Naive Bayes (MultinomialNB), Bi-LSTM, Bi-GRU, DistilBERT, BERT base uncased, RoBERTa base, and BERTweet base. We picked the combination of unigrams, bigrams, and trigrams of token counts as features for LinearSVC, Logistic Regression, and MultinomialNB since this combination delivered the best results in terms of the F-1 sarcastic metric.

For non-transformer-based models, we preprocessed the data by removing stop words and special characters. For transformer-based models, two approaches were adopted during the evaluation phase. In the first approach, we preprocessed the data as described in Figure 1. In the second one, we fine-tuned the model without applying any preprocessing to the data. Table 2 depicts the obtained results of various models on Sub-Task A - English.

We can see from Table 2 that BERTweet base model achieved the best results in detecting sarcastic tweets succeeded by RoBERTa base. Surprisingly, LinearSVC achieved better results than BERT base and DistilBERT.

We evaluated various models on Sub-Task C test set including LinearSVC, Logistic Regression, MultinomialNB, Bi-LSTM, Bi-GRU, RoBERTa base, and BERTweet base. Table 3 depicts the obtained results of various models on Sub-Task C - English. We mention that the same preprocessing approaches applied on Sub-Task A tweets were applied on Sub-Task C test set.

According to the reported results in Table 3, we can see that BERTweet base model achieved the best results succeeded by RoBERTa base. Moreover, we notice that traditional machine learning approaches such as LinearSVC, Logistic Regression, and MultinomialNB outperformed Recurrent Neural Networks: Bi-LSTM and Bi-GRU.

5 Conclusion

In this paper, we described our approach for tackling Sub-Task A and Sub-Task C of SemEval 2022 Task 6: iSarcasmEval - Intended Sarcasm Detection in English. Our submitted system consisted of a pre-trained transformer model for English Tweets named BERTweet, secured 8th and 5th positions respectively on Sub-Task A and Sub-Task C leaderboard.

Since the top-ranked system for the English Sub-Task A scored about 0.6052 F1-score for the sarcastic class, future studies and works will focus on improving the performance of sarcasm detection tasks by adopting other approaches such as data augmentation and oversampling.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Hamza Alami, Said Ouatik El Alaoui, Abdessamad Benlahbib, and Noureddine En-nahnahi. 2020. LISAC FSDM-USMBA team at SemEval-2020 task 12: Overcoming AraBERT's pretrain-finetune discrepancy for Arabic offensive language identification. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2080–2085, Barcelona (online). International Committee for Computational Linguistics.

Jorge A. Balazs and Juan D. Velásquez. 2016. Opinion mining and information fusion: A survey. *Information Fusion*, 27:95–110.

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, page 144–152, New York, NY, USA. Association for Computing Machinery.

Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

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

Hai Ha Do, PWC Prasad, Angelika Maag, and Abeer Alsadoon. 2019. Deep learning for aspect-based sentiment analysis: A comparative review. *Expert Systems with Applications*, 118:272–299.

D.I. Hernández Farias and P. Rosso. 2017. Chapter 7 - irony, sarcasm, and sentiment analysis. In Federico Alberto Pozzi, Elisabetta Fersini, Enza Messina, and Bing Liu, editors, *Sentiment Analysis in Social Networks*, pages 113–128. Morgan Kaufmann, Boston.

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

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

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

Sub-Task A - English					
Model	**F1-sarcastic**	F1-score	Precision	Recall	Accuracy
LinearSVC	0.3508	0.6055	0.6304	0.5957	0.7700
Logistic Regression	0.3076	0.6021	0.5950	0.6124	0.8200
MultinomialNB	0.2982	0.5924	0.5904	0.5948	0.8050
Bi-LSTM	0.2667	0.5575	0.5700	0.5536	0.7486
Bi-GRU	0.2358	0.5330	0.5463	0.5329	0.7221
DistilBERT (with preprocessing)	0.3070	0.5861	0.5975	0.5799	0.7743
DistilBERT (without preprocessing)	0.3267	0.5809	0.6162	0.5758	0.7350
BERT base (with preprocessing)	0.3388	0.5888	0.6258	0.5823	0.7407
BERT base (without preprocessing)	0.3087	0.5887	0.5983	0.5829	0.7793
RoBERTa base (with preprocessing)	0.3746	0.6454	0.6262	0.6823	0.8521
RoBERTa base (without preprocessing)	0.3984	0.6351	0.6642	0.6218	0.7886
BERTweet base (with preprocessing) (Official Submission)	0.4291	0.6513	0.6353	0.6896	0.7929
BERTweet base (without preprocessing)	**0.4334**	0.6547	0.6917	0.6384	0.7964

Table 2: The obtained results of various models on Sub-Task A - English test set

Sub-Task C - English				
Model	F1-score	Precision	Recall	**Accuracy**
LinearSVC	0.5392	0.5657	0.5952	0.5850
Logistic Regression	0.5996	0.6008	0.6004	0.6000
MultinomialNB	0.6033	0.6034	0.6033	0.6050
Bi-LSTM	0.4834	0.4835	0.4836	0.4850
Bi-GRU	0.5200	0.5233	0.5233	0.5200
RoBERTa base (with preprocessing)	0.7177	0.7172	0.7188	0.7200
RoBERTa base (without preprocessing)	0.7186	0.7186	0.7186	0.7200
BERTweet base (with preprocessing) (Official Submission)	0.7737	0.7735	0.7740	**0.7750**
BERTweet base (without preprocessing)	0.7585	0.7581	0.7589	0.7600

Table 3: The obtained results of various models on Sub-Task C - English test set

Diana Maynard and Mark Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

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

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on Czech and English Twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 213–223, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Yafeng Ren, Donghong Ji, and Han Ren. 2018. Context-augmented convolutional neural networks for twitter sarcasm detection. *Neurocomputing*, 308:1–7.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC^2 Workshop*.

Mohammad Tubishat, Norisma Idris, and Mohammad A.M. Abushariah. 2018. Implicit aspect extraction in sentiment analysis: Review, taxonomy, oppportunities, and open challenges. *Information Processing & Management*, 54(4):545–563.

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

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

X-PuDu at SemEval-2022 Task 6: Multilingual Learning for English and Arabic Sarcasm Detection

Yaqian Han, Yekun Chai, Shuohuan Wang, Yu Sun
Baidu, Inc.
{hanyaqian,chaiyekun,wangshuohuan,sunyu02}@baidu.com

Hongyi Huang, Guanghao Chen, Yitong Xu, Yang Yang
Shanghai Pudong Development Bank
{huanghy6,chengh13,xuyt3,yangy103}@spdb.com.cn

Abstract

Detecting sarcasm and verbal irony from people's subjective statements is crucial to understanding their intended meanings and real sentiments and positions in social scenarios. This paper describes the X-PuDu system that participated in SemEval-2022 Task 6, *iSarcasmEval - Intended Sarcasm Detection in English and Arabic*, which aims at detecting intended sarcasm in various settings of natural language understanding. Our solution finetunes pre-trained language models, such as ERNIE-M and DeBERTa, under the multilingual settings to recognize the irony from Arabic and English texts. Our system ranked second out of 43, and ninth out of 32 in Task A: one-sentence detection in English and Arabic; fifth out of 22 in Task B: binary multi-label classification in English; first out of 16, and fifth out of 13 in Task C: sentence-pair detection in English and Arabic.

1 Introduction

Sarcasm is the use of language that typically signifies the opposite to mock or convey contempt. As a narrow research field in natural language processing (NLP), sarcasm detection is a particular case in the spectrum of sentiment analysis, with important implications for a slew of NLP tasks, such as sentiment analysis, opinion mining, author profiling, and harassment detection. In the textual data, these tonal and gestural clues like heaving tonal stress and rolling of the eyes are missing, making it more difficult for machines.

The sarcastic intention of human annotators has potentially hindered the training and evaluation process in detecting the genuine emotions and positions of the natural language. Thus, this task (Abu Farha et al., 2022) adopted a novel data collection method (Oprea and Magdy, 2020), where authors themselves label the training samples. For sarcastic texts, the authors also rephrase them into non-sarcastic ones. Then, linguistic experts further checked the scathing pieces and labeled them into sub-categories of sarcasm defined by (Leggitt and Gibbs, 2000): sarcasm, irony, satire, understatement, overstatement, and rhetorical question.

This SemEval task requires the identification of sarcasm in either one sentence or sentence pairs in various language settings, which consists of three subtasks:

(1) Task A (English and Arabic): Given a text, determine whether it is sarcastic or non-sarcastic;

(2) Task B (English only): A binary multi-label classification task. Given a text, determine which ironic speech category it belongs to, if any;

(3) Task C (English and Arabic): Given a sarcastic text and its non-sarcastic rephrase, i.e. two texts that convey the same meaning, determine which is the sarcastic one.

Our method employed various multilingual or mono-lingual pre-trained language models, such as ERNIE-M (Ouyang et al., 2020) and DeBERTa (He et al., 2021) to address each component of this task, with a bunch of fine-tuning and ensemble techniques. Our system finally achieved

- 2nd out of 43 and 9th out of 32 in English and Arabic subtasks in *Task A*;
- 5th out of 22 in *Task B*;
- 1st out of 16 and 5th out of 13 in English and Arabic subtasks in *Task C*.

2 Previous Work

After detecting sarcasm in the speech was firstly proposed in (Tepperman et al., 2006), sarcasm detection has attracted extensive attention in the NLP community. Afterward, sarcasm detection in the

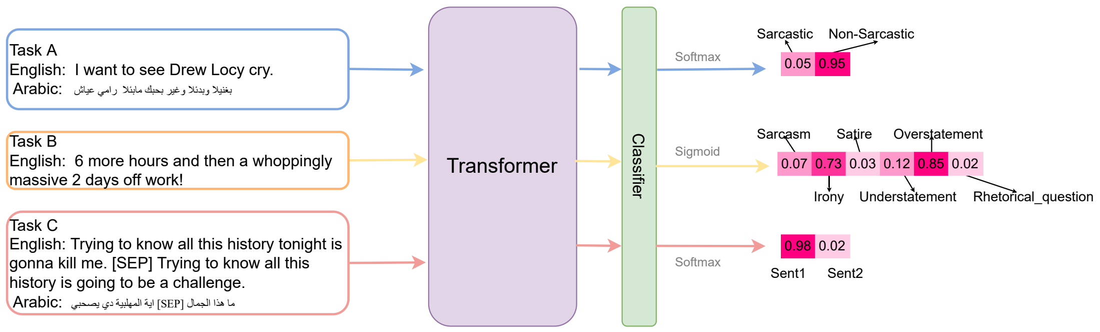

Task A
English: I want to see Drew Locy cry.
Arabic: بغنيلا وبدنلا وغير بحبك مابنلا رامي عياش
Task B
English: 6 more hours and then a whoppingly massive 2 days off work!
Task C
English: Trying to know all this history tonight is gonna kill me. [SEP] Trying to know all this history is going to be a challenge.
Arabic: ما هذا الجمال [SEP] اية المهلبية دي يصحبي
Transformer
Classifier
Softmax
Sigmoid
Softmax
Sarcastic
Non-Sarcastic
0.05
0.95
Sarcasm
Satire
Overstatement
0.07
0.73
0.03
0.12
0.85
0.02
Irony
Understatement
Rhetorical_question
0.98
0.02
Sent1
Sent2

Figure 1: Fine-tuning pre-trained models on the *iSarcasmEval* data.

text has been extended to a broad range of data forms in social media, such as tweets, comments, and TV dialogues, due to their public availability. Sarcasm detection spanned several approaches like rule-based, supervised, and semi-supervised (Joshi et al., 2016) methods, resulting in further development for automatic sarcasm detection. Rule-based methods mainly rely on linguistic information, and their classification accuracy is often not very high due to the presence of noisy data. Most previous work on sarcasm detection based on supervised machine learning tends to rely on different types of features, including sentence length, the number of capitalized words, punctuation (Davidov et al., 2010), pragmatic factors such as emoticons (González-Ibáñez et al., 2011), turn-level sentiment lexicon (Wilson et al., 2005), sarcasm markers (Ghosh and Muresan, 2018), and so on. Meanwhile, neural models have been applied to this task, relying on semantic relatedness (Amir et al., 2016) and neural intra-attention mechanism to capture the sarcasm (Tay et al., 2018) and thus reducing feature engineering efforts.

Recently, pre-trained language models such as BERT (Devlin et al., 2018), ERNIE (Sun et al., 2019), and GPT-3 (Brown et al., 2020), have set the new state-of-the-art in a wide range of NLP benchmarks, such as GLUE (Wang et al., 2018). Khatri et al. (2020) evaluated the performance of pre-trained model using feature-based and fine-tuning methods on irony detection in English tweets, finding the latter is better. Meanwhile, there is also a surge of applying pre-trained models in sarcasm detection (Dadu and Pant, 2020; Potamias et al., 2020; Javdan et al., 2020). Our system explored the multilingual and monolingual pre-trained language models to testify their fine-tuning performance on English and Arabic sarcasm detection tasks.

3 Approach

3.1 Pre-trained Language Models

We adopt *pretrain-then-finetune* paradigm for better leveraging the performance of large-scale pre-trained models. As illustrated in Figure 1, for all tasks, we utilize pre-trained models to extract the input representations, followed by a fully-connected feed-forward layer and a softmax/sigmoid activation after the `[CLS]` token for prediction. For sub-task A and B that input samples only contain one sentence, we directly fine-tune the pre-trained Transformers. For sub-tasks with two sentences, *i.e.*, sub-task C, we employ the multi-layer pre-trained Transformer blocks as the cross-encoder by concatenating sentence pairs and separating them with a `[SEP]` token.

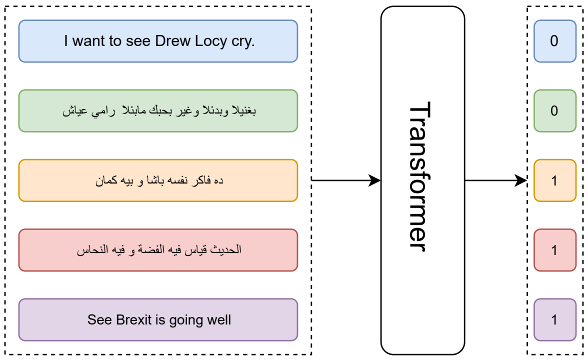

I want to see Drew Locy cry.
بغنيلا وبدنلا وغير بحبك مابنلا رامي عياش
ده فاكر نفسه باشا و بيه كمان
الحديث قياس فيه الفضة و فيه النحاس
See Brexit is going well
Transformer
0
0
1
1
1

Figure 2: Multilingual learning on Task A. “0/1” indicate the non-sarcastic and sarcastic class.

3.2 Multilingual Learning

By observing that subtasks in task A and task C, we found that both subtasks in Task A and C are for the same objective but in different languages, *i.e.*, Arabic and English. Therefore, we adopt multilingual learning method by simultaneously fine-tuning the pre-trained models on both Arabic and

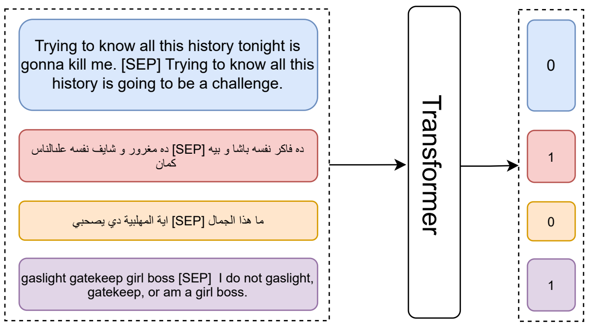

Trying to know all this history tonight is gonna kill me. [SEP] Trying to know all this history is going to be a challenge.
ده فاكر نفسه باشا و بيه [SEP] ده مغرور و شايف نفسه علىالناس كمان
ما هذا الجمال [SEP] اية المهلبية دي يصحبي
gaslight gatekeep girl boss [SEP] I do not gaslight, gatekeep, or am a girl boss.
Transformer
0
1
0
1

Figure 3: Multilingual learning on Task C. "0/1" indicate the first or second sentence belongs to sarcasm.

English training data based on multilingual pre-trained models, *i.e.*, ERNIE-M. Specifically, we combine both tasks in Task A or C as a single task, that is, training on Arabic and English sarcasm detection within the same subtask at the same time. As shown in Figure 2, we combine the one-sentence binary sarcasm detection subtasks in English and Arabic together and fine-tune the multilingual pre-trained models in one forward pass. Similarly, as illustrated in Figure 3, we conduct the identical settings for Task C. We found that this approach can achieve obvious performance gain on some specific settings and will discuss it in Secion 4.5.

3.3 Ensemble Learning

Considering the limited training data, we split the training data into k-fold with disparate random seeds, selecting one out of k data blocks for evaluation and using the rest $k-1$ for data training, as shown in Figure 4. Then, we choose the optimal model evaluated on various folds and random seeds. Finally, we apply ensemble techniques by averaging all outputs of test sets using optimal models.

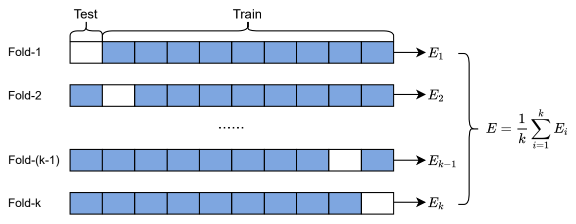


Figure 4: Illustration of ensemble learning. E_i indicates the prediction of the i-th model on the test set.

4 Experiments

4.1 Task Description

4.1.1 Task A: Binary Sarcasm Detection

The first task is binary text classification: given a tweet sample, the system needs to predict whether it is sarcastic or non-sarcastic. The following examples respectively present a sarcastic and non-sarcastic tweet.

(1) `The only thing I got from college is a caffeine addiction. (#sarcastic)`

(2) `I want to see Drew Lock cry. (#non-sarcastic)`

Example 1 is a sarcastic tweet where the author's true intention is "College is really hard, expensive, and exhausting, and I often wonder if the degree is worth the stress."

4.1.2 Task B: Multi-label Sarcasm Detection

The second task is a multi-label classification task, where the system requires to predict multiple categories out of six labels, such as `#Sarcasm`, `#Irony`, `#Satire`, `#Understatement`, `#Overstatement`, and `#Rhetorical_question`. The following examples provide examples for multiple sub-categories:

(1) `Falling asleep at your laptop is always fun. (#Sarcastic)`

(2) `Wow Bdubs can bench press 150 kilometers. (#Irony)`

(3) `Lil Pump is the Nelson Mandela of our generation. (#Satire #Sarcastic)`

(4) `Lucky for 2nd placed Brentford that there's no stand out team like Leeds this year, or they might have no chance of winning the league. (#Understatement #Sarcastic)`

(5) `6 more hours and then a whoppingly massive 2days off work! wowzers! (#Overstatement #Irony)`

(6) `wait what the fuck that solo yolo is mad?` `(#Rhetorical_question #Sarcastic)`

In the above examples, the types of sarcasm are subdivided into six categories. `#Sarcasm`, which is an ironic remark meant to mock by saying something different than what the speaker really means. For example, in example 1, the speaker hates falling asleep on his laptop. `#Irony` is when something happens that is the opposite of what was expected. As shown in example 2, the fact is that Bdubs cannot bench press 150 kilometers. `#Satire` is a type of wit that is meant to mock human vices or mistakes, often through hyperbole, understatement and sarcasm, as shown in Example 3. `#Understatement` is often a way of being critical. In example 4, because Norwich is the standout this year, Brentford cannot win the league. `#Overstatement` is an act of stating something more profound than it actually is, to make the point more serious, important, or beautiful. In example 5, a whoppingly massive two days off work means regret, and the genuine emotion ought not to require overstatement. `#Rhetorical_question` is a question that is asked even if the person doing the asking knows what the answer is. The solos in example 6 was truly expressed to be awful.

4.1.3 Task C: Binary Irony Classification on Two Sentences

The third subtask is binary classification: given a sarcastic tweet and its non-sarcastic rephrase (i.e., two tweets that convey the same meaning), the system needs to predict the sarcastic one. The following examples present a sarcastic sentence and its non-sarcastic paraphrase.

(1) `Trying to know all this history tonight is gonna kill me.` `(#Sarcastic)`

(2) `Trying to know all this history is going to be be a challenge.` `(#Rephrase)`

4.2 Evaluation Metrics

For these three sub-tasks, standard evaluation metrics including accuracy and F1 score are used to evaluate the participating system, calculated as follows:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

Task	#Instances	#Metric
`Task A`	1400	F1-score
`Task B`	1400	Macro-F1 score
`Task C`	200	accuracy

Table 1: Summary of official test set in SemEval-2022 Task6.

Class Label	#Instances
`sarcastic`	867
`non-sarcastic`	2601
`total`	3468

Table 2: Satirical and non-satirical categories in training data.

$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$recall = \frac{TP}{TP + FN} \tag{3}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4}$$

where TP, FP, TN, FN represent true positive, false positive, true negative, and false negative, respectively.

As shown in Table 1, task A, B and C use the F1-score for the sarcastic class, the Macro-F1 score over all classes, and accuracy, respectively. The Macro-F1 score implies that all class labels have equal weights in the final score.

4.3 Data

The detailed statistics of the sarcasm detection dataset are summarized in Table 2 and 3. As shown in Table 2, the training data are shown to be imbalanced, with 867 positive samples vs. 2601 negative ones. We only remove extra spaces, tabs, and line breaks for pre-processing. All emojis that contain emotional factors in training texts are kept without any change.

Multi-class Label	#Instances
`sarcasm`	713
`irony`	155
`satire`	25
`understatement`	10
`overstatement`	40
`rhetorical_question`	101

Table 3: Six satirical sub-categories in Task B.

4.4 Experiment Details

Due to the long-tailed nature of training data, we tried to use outside data[1] for data augmentation. Particularly, we merge the classes of "figurative""irony" in the exta data into a "sarcasm" class, but found it of no benefit. We conjecture that this is due to manual annotators' subjective intention can give the same samples various interpretations and therefore introduce some noise.

Still, the training data is relatively small and insufficient to achieve an unbiased performance estimate with a random train/test split. Instead, we use a k-fold cross-validation procedure ($k = 10$), a common model evaluation scheme in machine learning. The k-fold cross-validation procedure involves splitting the training dataset into k folds. In which $k - 1$ folds are used to train a model, and the rest one fold is used as the evaluation set. Finally, the final output of k models is the mean of these runs.

For English tasks, we compare ERNIE-M (Ouyang et al., 2020) and DeBERTa (He et al., 2021) as the pre-trained workhorse, while for Arabic tasks, we only consider ERNIE-M. We use the AdamW optimizer (Loshchilov and Hutter, 2017) and weight decay of 0.01. We warm up the learning rate for the first 10% of the update to a peak value of 1e-5 and 5e-6, respectively, and then linearly decay it afterward. We also use dropout (Srivastava et al., 2014) with a rate of 0.15 to prevent overfitting. We adopt a total batch size of 64 by running gradient accumulation on each GPU device with a step size of 8 and a batch size of 1, sharded across 8 NVIDIA V100 GPU chips. Our final solution is to ensemble all the model results obtained using a 10-fold cross-validation strategy with different learning rates (1e-5 and 5e-6) and training epochs (20 and 30), respectively.

4.5 Results

Table 4 compares the final performance on the official test set of task A,B,C under proposed model settings. It is obvious that DeBERTa outperforms ERNIE-M on English task since it is pre-trained only on English corpus. As to the multilingual learning in Task A and C, we observe the significant performance gain (*i.e.*, +6 absolute percentage point on F1 measure) on Task C while find it on par with monolingual fine-tuning on Task A. We guess

[1] https://www.kaggle.com/c/gse002/data?select=test.csv

Task	Lang	ERNIE-M (multilingual)	ERNIE-M (monolingual)	DeBERTa	Rank
Task A	en	36.75	38.46	**56.91**($*$)	2/43
	ar	40.36	**41.87**($*$)	-	9/32
Task B	en	N/A	-	**7.99**($*$)	5/22
Task C	en	82.50	75.00	**87.00** ($*$)	1/16
	ar	**90.50**	84.00($*$)	-	5/13

Table 4: Official test-set performance under various experimental settings. The "ERNIE-M (multilingual)" column indicates the performance of multilingual learning in Task A and C. Scores with asterisk indicate final submitted results. The official evaluation metrics for Task A,B,C are F1-score, macro F1-score, and accuracy, respectively.

this is because Task C are given two sentences for comparison, which is more straightforward than Task A (single sentence) to capture the ironic pattern for sarcasm detection. Due to the time limit, we only submit the monolingual fine-tuning results of ERNIE-M (*i.e.*, 84% acc.), which ranks 5th out of 13 in the Arabic subtask of Task C. Instead, the performance of our multilingual learning can achieve 2nd in Task C (Arabic). We contend that it would be worthwhile further exploring multilingual learning methods in various language settings in the future.

5 Conclusion

We present our system that participated in SemEval Task 6 and employ the multilingual learning method to train the English and Arabic tasks jointly. We empirically find that it confers benefits in specific scenarios and outranks the monolingual pre-trained models on Arabic tasks. However, we do not adopt other Arabic-specific pre-trained models, which is also worth comparing. In the future, it is a promising direction to explore different sarcasm detection approaches under multilingual settings.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mário J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *CoRR*, abs/1607.00976.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tanvi Dadu and Kartikey Pant. 2020. Sarcasm detection using context separators in online discourse. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 51–55.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcasm in Twitter and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 107–116, Uppsala, Sweden. Association for Computational Linguistics.

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

Debanjan Ghosh and Smaranda Muresan. 2018. "with 1 follower I must be AWESOME : P". exploring the role of irony markers in irony recognition. *CoRR*, abs/1804.05253.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in Twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586, Portland, Oregon, USA. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Soroush Javdan, Behrouz Minaei-Bidgoli, et al. 2020. Applying transformers and aspect-based sentiment analysis approaches on sarcasm detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 67–71.

Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Automatic sarcasm detection: A survey. *CoRR*, abs/1602.03426.

Akshay Khatri, Pranav P, and Anand Kumar M. 2020. Sarcasm detection in tweets with BERT and glove embeddings. *CoRR*, abs/2006.11512.

John S. Leggitt and Raymond W. Gibbs. 2000. Emotional reactions to verbal irony. *Discourse Processes*, 29(1):1–24.

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

Silviu Oprea and Walid Magdy. 2020. isarcasm: A dataset of intended sarcasm. *ArXiv*, abs/1911.03123.

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-m: enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. *arXiv preprint arXiv:2012.15674*.

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas-Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. *CoRR*, abs/1805.02856.

Joseph Tepperman, David Traum, and Shrikanth Narayanan. 2006. Yeah right: Sarcasm recognition for spoken dialogue systems.

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

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 347–354, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

DUCS at SemEval-2022 Task 6: Exploring Emojis and Sentiments for Sarcasm Detection

Vandita Grover
Department of Computer Science
University of Delhi
vgrover@cs.du.ac.in

Hema Banati
Dyal Singh College
University of Delhi
hemabanati@dsc.du.ac.in

Abstract

This paper describes the participation of team DUCS at SemEval 2022 Task 6: iSarcasmEval - Intended Sarcasm Detection in English and Arabic. Team DUCS participated in SubTask A of iSarcasmEval which was to determine if the given English text was sarcastic or not. In this work, emojis were utilized to capture how they contributed to the sarcastic nature of a text. It is observed that emojis can augment or reverse the polarity of a given statement. Thus sentiment polarities and intensities of emojis, as well as those of text, were computed to determine sarcasm. Use of capitalization, word repetition, and use of punctuation marks like '!' were factored in as sentiment intensifiers. An NLP augmenter was used to tackle the imbalanced nature of the sarcasm dataset. Several architectures comprising of various ML and DL classifiers, and transformer models like BERT and Multimodal BERT were experimented with. It was observed that Multimodal BERT outperformed other architectures tested and achieved an F1-score of 30.71%. The key takeaway of this study was that sarcastic texts are usually positive sentences. In general emojis with positive polarity are used more than those with negative polarities in sarcastic texts.

1 Introduction

According to the Collins Dictionary [1]

> "Sarcasm is speech or writing which actually means the opposite of what it seems to say".

Fox Tree et al. (2020) report that sarcasm is challenging to identify, even for humans. People often use gestures like rolling of eyes or heavy tonal stress to express sarcasm (Pandey et al., 2019), in speech or in-person communication. Intonation and stress in speech, too, are strong indicators of sarcasm (Castro et al., 2019). Also, the context in the form of shared knowledge between the speaker and the audience can be leveraged to detect sarcasm (Amir et al., 2016). But most of the time the tone or the context is missing in text data, especially tweet data. Some users attempt to use capitalization like "OH YEAH" or repetitions like "woowww" to indicate tonal intensity. But, most of the time these cues are not enough, especially in absence of context to gauge sarcasm. This makes sarcasm detection a challenging task.

In this work, team DUCS participated in *SemEval 2022 Task 6 : iSarcasmEval* Subtask A, (Abu Farha et al., 2022), to determine if the given text is sarcastic or not. The English text training dataset contains 3468 sentences. Each text is labeled as sarcastic or non-sarcastic by the text-author. Of the 3468 sentences 867 are sarcastic and 2601 are non-sarcastic. In the dataset corresponding to a sarcastic text, a rephrased non-sarcastic version has also been provided. These sarcastic texts are labeled as one of sarcasm, irony, satire, understatement, overstatement, and rhetorical question. These ironic speech categories are determined by a linguist expert. The target task is to determine whether a text is sarcastic or not on the Test dataset which contains 1400 sentences.

For sarcasm detection in the above-discussed dataset, an attempt was made to derive cues from the text itself. It is observed that in real scenarios, the absence of labels by users like #not or #sarcasm or any other contextual information makes sarcasm detection challenging (Chaudhari and Chandankhede, 2017). Thus, in this work, the authors attempted to use the information available in the text itself. Emoji occurrences in the text are captured to study their impact in discerning sarcasm. Sentiment intensity and polarity of both the emojis and the text were computed to train the classifiers

[1]https://www.collinsdictionary.com/dictionary/english/sarcasm

for sarcasm classification.
This paper is organized as follows. Section 2 discusses related work in sarcasm detection. In Section 3, System Overview, the proposed approach to incorporate polarity and intensity of emojis along with text sentiment polarity and intensity is discussed. Section 4, describes the Experimental Setup and the data augmentation approach used for the given dataset. The results of the experiments conducted are reported in Section 5. In the last section Conclusion, the key takeaways and learnings are discussed.

2 Related work

The sarcastic nature of the text can deceive classifiers as well as humans, given it conveys the opposite of what it means. Many researchers have attempted to use text sentiment and emojis for sarcasm detection.
Subramanian et al. (2019) use word and emoji embeddings simultaneously to train deep learning models with attention layer for sarcasm detection. Pamungkas and Patti (2018) utilize structural features and affective features of tweets for irony detection. Structural features include the presence of hashtags, links, emojis, quotes, etc. The counts of mentions, exclamations, upper-case, intensifiers, links, along with the counts of verbs, nouns, and adjectives are also used along with the use various affective resources to capture sentiment polarity and emotions. Lemmens et al. (2020) used an ensemble approach with LSTM for representing emojis and hashtags, CNN-LSTM for representation of cases, stop-words, punctuation, and sentiments. MLP was trained with Facebook's InferSent embeddings (Conneau et al., 2017) while SVM was trained with emotion and stylometric features. A decision tree with Adaboost served as the base estimator for the ensemble. This ensemble was used to predict sarcasm on the conversational text where context was available. Sundararajan and Palanisamy (2020) propose a rule-based classifier with an ensemble of 20 different features. They observed that the features based on sentiment can predict sarcasm better in combination with contradictory features.
In most work so far researchers have tried to capture sentiment and linguistic features to identify sarcasm. A few researchers have attempted to include emojis in sarcasm classification. But, how the usage of emojis with text impacts or conveys sarcasm is yet to be explored in depth. It needs to be studied how frequent is sentiment incongruence of emojis in the text they are used with and can the polarity of emojis and text help deduce sarcasm. Hence in this work, the authors focus on these features - emoji sentiment and text sentiment as well as their polarities to understand how this combination impacts sarcasm. This approach is discussed in the next section.

3 System overview

In the given dataset, situational context is not available. Although linguistic markers are provided, they may not be available in real scenarios, eg when the users post on social media. In the absence of context, it is important to take into account other factors that may point to sarcasm. For instance, the surface sentiment of a text is often used in many sarcasm classifiers (Joshi et al., 2016). Sulis et al. (2016), report that most often sarcasm is used with an apparently positive statement to produce a negative impact. It is noted that the users may use intensity in the form of capitalization, repeated letters, etc. to express sarcasm (Chaudhari and Chandankhede, 2017). In this work, to identify sarcasm, the sentiment polarity of the text and intensity in the form of sentiment score was incorporated for the sarcasm classification task. Intensity as a feature is used as it may help capture intonation or stress a user may want to express.
It has been reported that features like emojis can augment as well as alter the sentiment polarity (Grover, 2021).
Pamungkas and Patti (2018) observed that emoji incongruity with the text may help improve the irony classification. Thus, this work investigates whether in sarcastic texts users use in-congruent emojis with the text to express sarcasm. And whether the use of emojis in the text helps in detecting sarcasm?

Data preparation The text dataset was cleaned to remove URLs as they do not contribute to the text sentiment. Most text preprocessing approaches in sentiment classification convert characters to lower case, reduce repeating letters, and perform lemmatization. To compute the sentiment score and polarities, special care was taken to not remove repetition of letters, capitalization or punctuation as they may serve as intensifiers. These special usages can later be removed during the experimentation while training the classifiers.

Emoji extraction In the next step, the emojis are separated from text. The training dataset for this task contained 672 text sentences with emojis of which 195 were sarcastic. The *demoji 1.1.0* Python package [2], is used for emoji extraction.

Text sentiment polarity and intensity To evaluate intensity or the sentiment score of the text and text polarity the VADER sentiment analyzer (Hutto and Gilbert, 2014), available in Python's natural language toolkit is employed. VADER is observed to work better in the language used by the users on social media (Illia et al., 2021), (Bonta and Janardhan, 2019). Users tend to communicate informally, using slang, abbreviations, capitalization and punctuation marks, etc. on social media. VADER can capture these components while computing sentiment polarity and sentiment scores. The compound scores provided by VADER are used as the sentiment score. If the compound score is $\leq$ -0.5 the sentiment polarity is considered negative, between (-0.5, 0.5), the polarity is considered as neutral, while the score $\geq$ = 0.5 is considered to be positive.

Emoji sentiment polarity As discussed above, since emojis are known to contribute to text sentiment and in many cases, they may even reverse the sentiment polarity, the emojis extracted for each text are stored separately in the list. A total of 672 texts have emojis in the provided dataset.
For extracting emoji sentiment, Emoji Sentiment Ranking (Novak et al., 2015) is employed. Emoji Sentiment Ranking gives the sentiment score of the emojis.
A text may contain numerous emojis which is an indicator of both the context as well as associated emotion with the text. For example, a single emoji used several times with a text may be considered as a sentiment intensifier. Thus, all the emojis in the text were considered while computing emoji polarity and emoji sentiment score.
Emoji sentiment scores were captured for all emojis. If an emoji did not appear in the Emoji Sentiment Ranking, it was demojized using the *emoji 1.6.3* Python package[3]. Demojizing here means replacing the emojis with their Unicode Consortium description in English, (Unicode, 2022). These emoji descriptions are passed to the VADER sentiment analyzer to compute the sentiment score of the emojis. Again the compound score provided by VADER is used to compute the final score. Scores of all the emojis are then added to compute the total emoji score for all the emoji scores associated with the text. These emojis are then normalized as done in VADER and corresponding emoji polarities are computed.

$$normalizedScore = \frac{score}{\sqrt{score^2 + \alpha}} \quad (1)$$

where default value of α is 15.
Each text with emojis is labeled with the computed emoji score and emoji polarity. For the text with no emojis, the emoji sentiment score is assigned to zero, and polarity is assigned as neutral.
For example if the text is *"@AsdaServiceTeam imagine your delivery being 2 hours late, and imagine calling up your service team only for them to hang up at 10pm, coincidentally the same time the office closes. But it's okay, my £3 delivery fee is being refunded though 👊"*. The VADER text score is computed as 0.2263 with "+" text polarity. The VADER emoji sentiment score is computed as 0.0150371002692231 with "+" emoji polarity. The prepared dataset with the derived features of text sentiment scores, text polarity, emoji sentiment score, and emoji polarity is now used to train the different classifiers discussed in the next section.

4 Experimental setup

For sarcasm detection, this work employs various machine and deep learning classifiers. The proposed set up is also tested with transformer models like BERT and Multimodal BERT.
First, the text is cleaned to remove stop words, newlines, and spaces. The cleaned text is then converted to lower case and then tokenized to prepare for loading the word embeddings. Pre-trained GloVe embeddings, (Pennington et al., 2014) of 100 dimensions are used to represent the tokenized text.
80% of the dataset was used for training while the remaining 20% was used for validation.

Data augmentation This dataset was highly imbalanced with 867 sarcastic and 2601 non-sarcastic texts, i.e. (3:1 ratio for non-sarcastic: sarcastic texts). So the dataset was augmented to improve this ratio of sarcastic and non-sarcastic texts. The sarcastic texts were oversampled with *nlpaug* python library [4] for text augmentation.

[2]https://pypi.org/project/demoji/
[3]https://pypi.org/project/emoji/
[4]https://nlpaug.readthedocs.io/en/latest/index.html

The contextual BERT embeddings were used to synthesize new texts from existing texts in the dataset. Two different augmented datasets were created. In the first augmented dataset (Aug1 Training Set), each sarcastic text was synthesized once to create one more text. In this set, the ratio of non-sarcastic: sarcastic texts was 2:1. In the next augmented dataset (Aug2 Training Set), each sarcastic text was synthesized to create two more texts, thus resulting in the ratio of 1:1 for non-sarcastic to sarcastic texts.
For example, if the tweet was, "See Brexit is going well". Then in the first augmented text (Aug1 Training Set) the original tweet (mentioned above) was retained and a synthesized tweet in the form "can see brexit negotiations is going extremely well" was added to the dataset. This was done for every sarcastic tweet, making the ratio of non-sarcastic:sarcastic texts as 2:1.
In the Aug2 Training Set) the number of synthesized texts was increased to two. So two synthesized tweets apart from the original tweet were appended to the dataset making the ratio of non-sarcastic:sarcastic as 1:1.
The computed sentiment polarity, text sentiment score, emoji sentiment polarity, and emoji sentiment score were replicated from the original texts in the synthesized texts.

Model architecture The different architectures experimented with are as follows:

- Gaussian Naive Bayes (Perez et al., 2006)
- Support Vector Machine with Linear Kernel (Wang, 2005)
- Logistic Regression (Wright, 1995)
- Sequential model with 1 dense fully connected layer (Sutskever et al., 2014)
- LSTM with Adadelta optimiser, 128 batch size, run over 10 epochs (Sundermeyer et al., 2012)
- Bi-LSTM (Graves and Schmidhuber, 2005) with Adadelta optimiser, 128 batch size, run over 10 epochs
- BERT (Devlin et al., 2018) with the learning rate of 1e-5, batch size 32 run over 4 epochs with the base uncased classifier
- Multimodal BERT with the batch size 32 run over 6 epochs with the base uncased classifier

5 Results

This section reports the results of various classifiers on which the proposed approach was tested with.
The original training set (Original Training Set) provided for the competition is an imbalanced dataset contains 2601 non-sarcastic and 867 sarcastic texts. The training set augmented with 1X sarcastic tweets (Aug1 Training Set) contains 1734 sarcastic and 2601 non-sarcastic texts. 867 sarcastic texts were synthesized to create 867 more sarcastic texts, resulting in a total of 1734 sarcastic texts. The training set augmented with 2X sarcastic tweets (Aug2 Training Set) contains 2601 sarcastic and 2601 non-sarcastic texts. In this set, each of the 867 sarcastic texts were synthesized twice. Hence the total number of sarcastic texts were 867(original) and 1734(each original tweet synthesized twice).
Validation-F1 reports F1 score (%) that were obtained during the validation. Results obtained on the test dataset, (Abu Farha et al., 2022), provided during the competition are reported in Test-F1.
The results of models trained without sentiment scores and polarities on the original training set in Table 1, training set augmented with 1X sarcastic tweets in Table 2, and training set augmented with 2X sarcastic tweets in Table 3 are reported.

Classifier	**Validation-F1**	**Test-F1**
Gaussian Naive Bayes	25.87	14.52
Logistic Regression	7.03	5.45
SVM	7.11	4.65
Sequential	44.09	24.53
LSTM	44	25
Bi-LSTM	44	25
BERT	77.56	22.46

Table 1: Original Training Set without Sentiment Scores and Polarities

The results of models trained with text and emoji sentiment scores and their respective polarities on Original, Aug1, and Aug2 are reported in 4, 5, 6 respectively.

The Multimodal BERT trained with 1736 sarcastic and 2601 non-sarcastic sentences was eventually used to determine sarcastic text in the test data. With this work the authors achieved an overall F1-Score of 30.71% on the test dataset and was ranked 24 out of the 43 participating teams.

Classifier	Validation-F1	Test-F1
Gaussian Naive Bayes	50.73	19.39
Logistic Regression	19.13	16.18
SVM	19.5	16.42
Sequential	56.48	24.53
LSTM	56.48	25
Bi-LSTM	56.48	24.53
BERT	64	23.94

Table 2: Aug1 Training Set without Sentiment Scores and Polarities

Classifier	Validation-F1	Test-F1
Gaussian Naive Bayes	59.04	16.58
Logistic Regression	61.4	22.5
SVM	61.57	22.58
Sequential	66.71	25
LSTM	66.71	25
Bi-LSTM	66.71	25
BERT	64	23.94

Table 3: Aug2 Training Set without Sentiment Scores and Polarities

It was observed that in general, models trained with sentiment scores of text and emojis alongwith their polarities performed better than those trained only with text
Figure 1 shows the distribution of emojis across sarcastic text in the train and test dataset. From Figure 1 of iSarcasmEval dataset

- It is observed that sarcastic texts mostly have emojis with positive polarity.
- Sarcastic texts generally demonstrate positive sentiment.

Some other observations noted during the experiments are reported as follows.

- Sarcastic text when augmented twice results in overfitting and results are not satisfactory on the test set. The best F1-score on the test dataset is achieved when each sarcastic text is augmented once.

Classifier	Validation-F1	Test-F1
Gaussian Naive Bayes	21.3	15.72
Logistic Regression	6.41	2.84
SVM	6.52	2.87
Sequential	37.57	24.53
LSTM	47.03	28.47
Bi-LSTM	47.03	28.48
BERT	35.29	23.28
MultiModal BERT	50.00	25.12

Table 4: Original Training Set with Text and Emoji Sentiment Scores and Polarities

Classifier	Validation-F1	Test-F1
Gaussian Naive Bayes	32.0	18.5
Logistic Regression	17.56	11.11
SVM	17.53	10.89
Sequential	56.75	24.52
LSTM	40.73	28.49
Bi-LSTM	40.51	28.49
BERT	73.184	29.27
MultiModal BERT	75.22	**30.71**

Table 5: Aug1 Training Set with Text and Emoji Sentiment Scores and Polarities

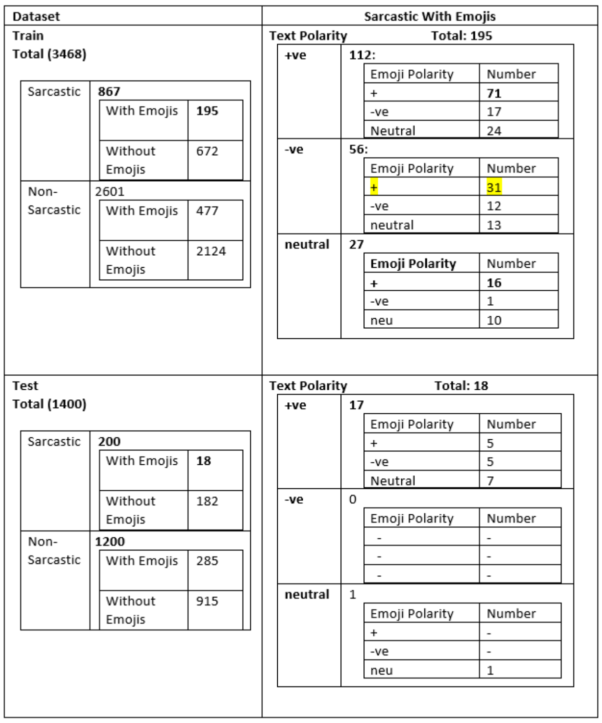
Classifier	Validation-F1	Test-F1
Gaussian Naive Bayes	57.51	20.93
Logistic Regression	54.3	20.62
SVM	54.77	20.75
Sequential	64.26	24.52
LSTM	49.82	28.50
Bi-LSTM	49.82	28.50
BERT	78.36	24.31
MultiModal BERT	81.22	24.69

Table 6: Aug2 Training Set with Text and Emoji Sentiment Scores and Polarities

- Overall the Multi-modal BERT, which can handle tabular data, performs the best.
- Models trained with sentiment scores and polarities of text and emojis performed slightly better than those trained with only text.
- In the test dataset only 18 sarcastic sentences had emojis.
- F1-score on test dataset for text with emojis was found to be 21.05% while that of text without emojis was 31.02%. This performance may be improved if more sentences with emojis are available.

6 Conclusion

In this work, the authors took their first steps in understanding how the sentiment of emojis alongwith the text sentiment helps in detecting sarcasm. Since sarcasm is difficult to detect without context, authors attempted to uncover information implicit in the text. For this the intensity and polarity of both the available emojis and the text itself were used to identify sarcasm.
It was observed that sarcastic texts generally have positive polarity. Sarcastic texts employed emojis with positive polarities more than the negative emojis. This information can be used further to improve results of sarcasm classification. The given dataset



Dataset				
Train **Total (3468)**	Sarcastic	**867**	With Emojis	**195**
			Without Emojis	672
	Non-Sarcastic	2601	With Emojis	477
			Without Emojis	2124
Test **Total (1400)**	Sarcastic	**200**	With Emojis	**18**
			Without Emojis	182
	Non-Sarcastic	**1200**	With Emojis	285
			Without Emojis	915

Sarcastic With Emojis			
Text Polarity	**Total: 195**		
+ve	**112:**	Emoji Polarity	Number
		+	**71**
		-ve	17
		Neutral	24
-ve	**56:**	Emoji Polarity	Number
		+	31
		-ve	12
		neutral	13
neutral	**27**	**Emoji Polarity**	Number
		+	**16**
		-ve	1
		neu	10
Text Polarity	**Total: 18**		
+ve	**17**	Emoji Polarity	Number
		+	5
		-ve	5
		Neutral	7
-ve	0	Emoji Polarity	Number
		-	-
		-	-
		-	-
neutral	1	Emoji Polarity	Number
		+	-
		-ve	-
		neu	1

Figure 1: Distribiution of Emojis in Sarcastic Text in Train and Test Datasets of iSarcasmEval

had relatively fewer texts with emojis, thus, more work is required to fully capitalize on the proposed approach. This motivates the authors to explore more sarcasm datasets with emojis to exhaustively study the impact of emojis in identifying sarcasm.

7 Acknowledgement

The authors thank the organizers of SemEval-22 organizers and iSarcasmEval for the conduct of this competition. We thank fellow competitors and peer reviewers for contributing to our learning.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976*.

Venkateswarlu Bonta and Nandhini Kumaresh2and N Janardhan. 2019. A comprehensive study on lexicon based approaches for sentiment analysis. *Asian Journal of Computer Science and Technology*, 8(S2):1–6.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _obviously_ perfect paper). *arXiv preprint arXiv:1906.01815*.

Pranali Chaudhari and Chaitali Chandankhede. 2017. Literature survey of sarcasm detection. In *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 2041–2046.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jean E Fox Tree, J Trevor D'Arcey, Alicia A Hammond, and Alina S Larson. 2020. The sarchasm: Sarcasm production and identification in spontaneous conversation. *Discourse Processes*, 57(5-6):507–533.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

Vandita Grover. 2021. Exploiting emojis in sentiment analysis: A survey. *Journal of The Institution of Engineers (India): Series B*, pages 1–14.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225.

Fathonah Illia, Migunani Puspita Eugenia, and Sita Aliya Rutba. 2021. Sentiment analysis on pedulilindungi application using textblob and vader library. In *Proceedings of The International Conference on Data Science and Official Statistics*, volume 2021, pages 278–288.

Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Automatic sarcasm detection: A survey. *CoRR*, abs/1602.03426.

Jens Lemmens, Ben Burtenshaw, Ehsan Lotfi, Ilia Markov, and Walter Daelemans. 2020. Sarcasm detection using an ensemble approach. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 264–269.

P Kralj Novak, J Smailovic, B Sluban, and I Mozetic. 2015. Emoji sentiment ranking. *PLoS ONE*, 10(12):e0144296.

Endang Wahyu Pamungkas and Viviana Patti. 2018. # nondicevosulserio at semeval-2018 task 3: Exploiting emojis and affective content for irony detection in english tweets. In *International Workshop on Semantic Evaluation*, pages 649–654. Association for Computational Linguistics.

Avinash Chandra Pandey, Saksham Raj Seth, and Mahima Varshney. 2019. Sarcasm detection of amazon alexa sample set. In *Advances in Signal Processing and Communication*, pages 559–564. Springer.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Aritz Perez, Pedro Larranaga, and Inaki Inza. 2006. Supervised classification with conditional gaussian networks: Increasing the structure complexity from naive bayes. *International Journal of Approximate Reasoning*, 43(1):1–25.

Jayashree Subramanian, Varun Sridharan, Kai Shu, and Huan Liu. 2019. Exploiting emojis for sarcasm detection. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 70–80. Springer.

Emilio Sulis, Delia Irazú Hernández Farías, Paolo Rosso, Viviana Patti, and Giancarlo Ruffo. 2016. Figurative messages and affect in twitter: Differences between# irony,# sarcasm and# not. *Knowledge-Based Systems*, 108:132–143.

Karthik Sundararajan and Anandhakumar Palanisamy. 2020. Multi-rule based ensemble feature selection model for sarcasm type detection in twitter. *Computational intelligence and neuroscience*, 2020.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

The Unicode. 2022. The Unicode Consortium. http://www.unicode.org/emoji/charts/full-emoji-list.html. [Online; accessed 15-February-2022].

Lipo Wang. 2005. *Support vector machines: theory and applications*, volume 177. Springer Science & Business Media.

Raymond E Wright. 1995. Logistic regression.

UMUTeam at SemEval-2022 Task 6: Evaluating Transformers for detecting Sarcasm in English and Arabic

José Antonio García-Díaz and **Camilo Caparrós-Laiz** and **Rafael Valencia-García**[*]
Facultad de Informática, Universidad de Murcia, Campus de Espinardo, 30100, Spain
{joseantonio.garcia8,camilo.caparrosl,valencia}@um.es

Abstract

In this manuscript we detail the participation of the UMUTeam in the iSarcasm shared task (SemEval-2022). This shared task is related to the identification of sarcasm in English and Arabic documents. Our team achieve in the first challenge, a binary classification task, a F1 score of the sarcastic class of 17.97 for English and 31.75 for Arabic. For the second challenge, a multi-label classification, our results are not recorded due to an unknown problem. Therefore, we report the results of each sarcastic mechanism with the validation split. For our proposal, several neural networks that combine language-independent linguistic features with pre-trained embeddings are trained. The embeddings are based on different schemes, such as word and sentence embeddings, and contextual and non-contextual embeddings. Besides, we evaluate different techniques for the integration of the feature sets, such as ensemble learning and knowledge integration. In general, our best results are achieved using the knowledge integration strategy.

1 Introduction

Sarcasm is a form of rhetorical device based on biting humour to *disarm* an opponent during a dialog (Wilson, 2006). On the Web and, specially in social networks and opinion forums, sarcasm is very popular because it is funny to read and helps to stimulate the viral phenomenon of social media content (Peng et al., 2019). As sarcasm usually relies on figurative language and wordplay, in which words diverts from their conventional meaning, sarcastic statements hinder the ability of automatic classification tasks to perform sentiment analysis, hate-speech detection, or author analysis among other tasks.

From a Natural Language Processing (NLP) perspective, sarcasm, among other forms of figurative speech, such as irony or satire, has been explored in (del Pilar Salas-Zárate et al., 2020). In this work, the authors explored what stands out the most discriminant features for satire and irony detection. The authors identified a total of 25 feature sets. Apart from lexicon-based features and word-n-grams features, such as unigrams or bigrams, the authors identified style features, sentiment and emotional features, pragmatic features, and punctuation features, to name a few. Our proposal for solving sarcasm detection includes language-independent feature sets extracted with a custom tool, UMUTextStats (García-Díaz et al., 2021; García-Díaz and Valencia-García, 2022; García-Díaz et al., 2022a,b).

In this work, we describe the participation of the UMUTeam at SemEval 2022 task 6, concerning sarcasm identification in Arabic and English (Abu Farha et al., 2022). In this edition, three challenges are proposed. The first one is a binary classification problem to determine whether a document is sarcastic or not. This challenge is for English and Arabic. The second challenge is available only in English, and consists in a multi-label classification task discerning among different types of ironic speech. Finally, the last challenge consists of the identification of a sarcastic document between itself and a non-sarcastic rephrase, but with the same meaning. Our team attemted to participate in the first and second challenge. We achieve a F1 score of the sarcastic class of 17.97% for English, and 31.75% for Arabic in the first challenge. However, our results are not considered in the official leader board for the second challenge due to an unknown error. Therefore, we report the results for the second challenge with the validation split of dataset.

2 Dataset

The dataset proposed at iSarcasm 2022 was annotated by the authors themselves. Besides, each author was asked to rephrase their sarcastic texts without the usage of sarcasm. Finally, some linguistic

[*]Corresponding author

Split	English	Arabic
training	2774	2800
val	694	700
test	1400	1400
total	4868	4900

Table 1: Corpus statistics by split and language for the first challenge in English and Arabic

Trait	Training	Validation
irony	130	25
overstatement	35	5
rhetorical question	81	20
sarcasm	565	148
satire	19	6
understatement	9	1

Table 2: Corpus statistics per sarcastic mechanism (subtask 2, English)

experts were asked to perform the multi-label annotations based on the following ironic speech labels: sarcasm, irony, satire, understatement, overstatement, and rhetorical question (Leggitt and Gibbs, 2000).

The statistics of the dataset concerning the first challenge are shown in Table 1. There is a significant imbalance between the labels. The relationship between the sarcasm and non-sarcasm texts is a 1:3 for English and 1:6 for Arabic. The statistics concerning the second challenge are shown in Table 2. There are 713 documents annotated as sarcasm, 155 as irony, 101 as rhetorical questions, 40 as overstatement, 25 as satire, and 10 as understatement.

From the training set, we select a 20% of instances to build the validation set using stratified sampling, in order to keep the balance.

3 System architecture

Figure 1 depicts the architecture of our proposal. Basically, we build two systems: one for English and one for Arabic, so we could apply different preprocessing techniques and apply language-specific pretrained embeddings. In a nutshell, this pipeline can be described as follows. First, is the preprocessing step module. For both languages, we ensure that the dataset does not contains hyperlinks, hashtags, quotations or emojis. Plus, for the English dataset we expand acronyms. Second, is the datasplitter, to divide the iSarcasm dataset into training and validation. As there was a strong imbalance in the dataset, we keep this imbalance in both splits. Third, is the feature extraction module, for extracting the language-independent linguistic features and the sentence embeddings. Forth, is the training of several neural networks using hyperparameter selection. Finally, is the feature integration module, in which we evaluate ensemble learning and knowledge integration in order to combine the results of each neural network.

Next, the feature extraction module is described. The first feature set is a subset of language-independent linguistic features (LF) from UMUTextStats. This feature sets includes Part-of-Speech (PoS) features and stylometric features concerning several linguistic metrics such as Type Token Ratio (TTR), punctuation symbols and corpus length. The second and third feature sets are, respectively, non-contextual word and sentence embeddings from FastText (Mikolov et al., 2018). For this, we use the Arabic and English pretrained models. The word embeddings allow to evaluate convolutional and recurrent neural network architectures, apart from multi-layer perceptrons that suitable for feature sets of fixed size. The forth feature set is contextual sentence embeddings. We use BERT (Devlin et al., 2018) for English, and Arabic BERT (Safaya et al., 2020) for Arabic. To extract these embeddings, we applied a similar method as described at (Reimers and Gurevych, 2019). Before extracting the sentence embeddings, we used RayTune (Bergstra et al., 2013) to fine tune BERT and Arabic BERT. For this, we used Tree of Parzen Estimators (TPE) to select the best hyperparameters from a total of 10 trials. The hyperparameters evaluated are: (1) the weight decay (between 0 and 0.3); (2) two training batch sizes: 8 and 16 (we were limited to the GPU); (3) four warm-up steps: 0, 250, 500, 1000; (4) the number of training epochs, between 1 and 5; and (5) a learning rate between 1e–5 and 5e–5.

Once the feature sets are extracted, the next step in the pipeline is the training of the neural networks. We train a neural network per feature set and a neural network combining all the feature sets, using a knowledge integration strategy. Each training is performed with an hyperparameter optimisation stage. This evaluation includes 20 shallow neural networks, in which one or two hidden layers are stacked and that contains the same number of neurons. For the shallow neural networks we evaluate the following activation functions: (`linear`,

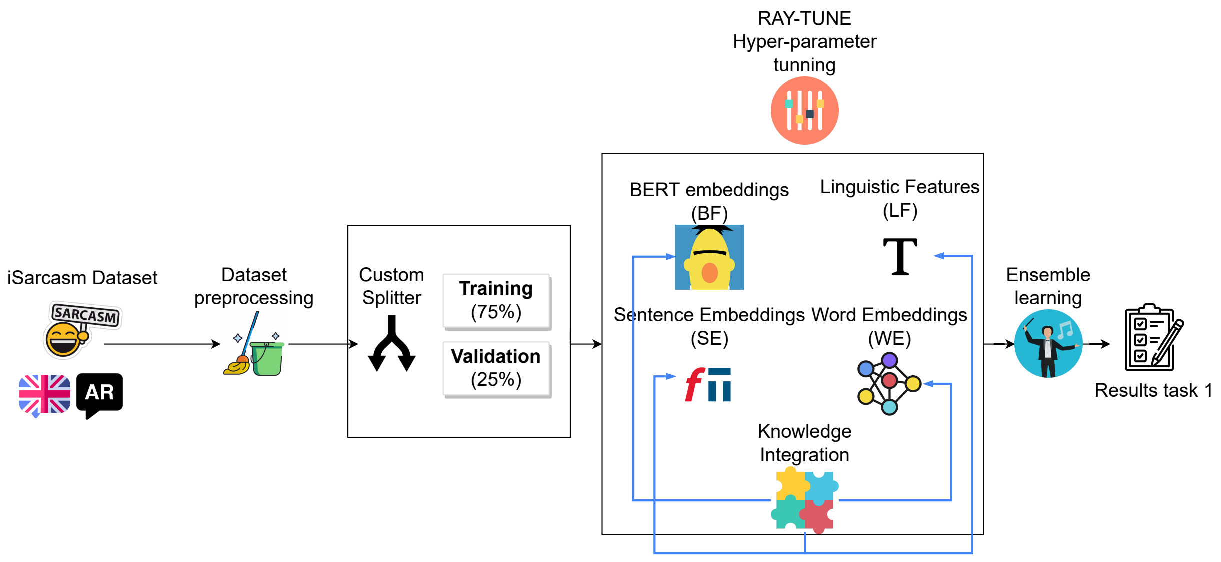


Figure 1: System architecture for solving the iSarcasm 2022 shared tasks

`ReLU`, `sigmoid`, and `tanh`). We also evaluate 5 deep-learning networks, composed from 3 to 8 hidden layers, in which the number of neurons per layer are arranged different shapes (brick, triangle, diamond, rhombus, and funnel). The activation functions evaluated for the deep-learning networks are the `sigmoid`, `tanh`, `SELU` and `ELU`. We also adjust the learning rate to evaluate `10e-03` or `10e-04`. As commented above, the word embeddings from fastText allow us to evaluate 10 convolutional and 10 bidirectional recurrent neural networks. As the dataset was heavily imbalanced, we evaluate large batch sizes, so in all the experiments, we evaluate large batch sizes: `128`, `256`, and `512`. We apply these large batch sizes for ensuring all batches contains sufficient number of instances of both classes. In addition, we apply regularisation by using a dropout mechanism (`[False, .1, .2, .3]`).

Besides, we evaluate two ensemble learning strategies to combine all the features. One method consisting of hard voting the labels predicted by each model (mode) and another method consisting of averaging the probabilities of the label of each model (mean).

It is worth mentioning that one of the tasks of English language is a multi-label problem. To solve it, we trained several binary classification models, one per ironic speech label.

4 Results and validation

During the development stage, we evaluate our models with a custom validation split. The results for the first task are depicted in Table 3, both for English and Arabic. We can observe that there is a strong difference between the results of the English and the Arabic datasets.

First, in case of English, we can observe that the results for identifying sarcasm (F1-pos) are limited, reaching a F1-score of 45.82% with the knowledge integration strategy. We focus on this metric because the validation split is unbalanced, containing 173 sarcastic documents and 521 non-sarcastic documents. The results of the LF are limited in case of English. LF are based mostly on stylometric and PoS features, which are not enough for the correct identification of sarcasm in English. Moreover, the results concerning the sarcasm label achieved with the non-contextual word embeddings (WE) are similar to the ones achieved with the LF (40.10% vs 40.08%), both slightly worse than non-contextual sentence embeddings (SE, 42.27%). The results achieved with transformers (BF) are the best results achieved with the feature sets evaluated in isolation (F1-pos of 45.10%). When combining the results, we observe that we achieve slightly superior results by the knowledgte integration approach, improving to a F1-pos of 45.82%. However, the results are more limited with the the ensemble strategies, achieving the lower results of 38.06% (mode) and 37.96% (mean). The identification of the non-sarcasm label (F1-neg) is more similar (and even slightly superior) compared with the rest of the strategies.

Second, in case of Arabic, we observe astonishing results for all feature sets. We reviewed the

	English			Arabic		
Strategy	**F1-neg**	**F1-pos**	**F1-score**	**F1-neg**	**F1-pos**	**F1-score**
LF	67.92	40.08	54.00	98.14	90.00	94.07
SE	73.21	42.27	57.74	97.99	88.35	93.17
WE	76.68	40.10	58.39	98.07	89.00	93.53
BF	82.40	45.10	63.75	99.08	94.69	96.88
LF-SE-WE-BF (K.I.)	83.57	**45.82**	**64.69**	99.08	94.69	96.88
LF-SE-WE-BF (mode)	**83.71**	38.06	60.89	**99.17**	**95.05**	**97.11**
LF-SE-WE-BF (mean)	78.84	37.96	58.40	96.71	81.86	89.28

Table 3: Results for the custom validation split in the first challenge for English and Arabic. We show F1-neg for non-sarcasm, F1-pos for sarcasm, and the macro F1-score (F1-score) for the neural networks evaluated with a feature set, and the combinations of features by using knowledge integration and two ensemble learning strategies.

dataset in order to find duplicates but we could not identify a relevant number of them. The identification of a large number of duplicates could indicate that some of the instances of the training split were present in our custom dataset. In this case and due to our lack of understanding of Arabic, we could not identify in the validation split the high performance achieved. After the competition, the ground truth labels of the test set were released for the development of the working notes. We observed that the results with the test split are more limited. For example, the F1-score falls from 94.07% with LF with the custom validation split to 33.82% with the official test split. The drop in the results cannot be explained by class imbalance, as we performed a stratified split in order to build the validation split. In fact, our validation split has 102 sarcasm documents whereas the rest (598) are non-sarcasm. The official test contains 1400 documents, 200 labelled as sarcasm whereas the rest were labelled as non-sarcasm. The results achieved with the rest of the feature sets SE, WE, and BF are in the same line, achieving the best result with a macro F1-score of 44.85% with the knowledge integration strategy.

Next, we report the results achieved for each trait separately in Table 4. This table reports the overall macro F1-score. The limited results achieved are caused by the strong imbalance among the validation split. From a total of 694 samples of the validation split of the English dataset, there are 25 samples of irony, 5 of overstatement, 20 rhetorical questions, 148 based on pure sarcasm, 6 based on satire, and 1 for understatement. These results indicate that one of the main drawbacks of our proposal is related to handle class imbalance, as the majority of the neural networks developed does not behave better than a random classifier. Moreover, some surprising results are achieved for the rhetorical questions and understatements, with the ensemble strategy of averaging the probabilities (mean).

Concerning the official results, we achieved very limited results for the binary classification challenge in English, achieving position 41 with a F1-score of the sarcastic class of 17.97% (see Table 5). The best result was achieved by user *stce* with a F1-score over the sarcastic label of 60.52%, which indicates that our system is far from the best results. The average F1-score of the sarcastic label of the rest of the participants is 32% with a standard deviation of 11.24%.

The results achieved with the Arabic dataset for the binary classification were better, with a F1-score of the sarcastic label of 31.75%, reaching position 22 of a total of 32 participants. (see Table 6). In this case, the best result was achieved by user *Abdelkader* with an F1-score over the sarcastic label of 56.32%. The average F1-score of the sarcastic label of the rest of the participants is 35% with a standard deviation of 10.04%.

5 Conclusions and further research lines

In this working notes we have described the participation of the UMUTeam in the iSarcasm shared task of SemEval 2022. In this shared task, the participants were required to solve a binary and a multi-label classification task regarding sarcasm identification in English and Arabic. We achieved a F1 score of the sarcastic class of 17.97% for English, and 31.75% for Arabic in the first challenge.

After sending the official results, we received the annotated test set. Although we are happy with our participation in this shared task, as we have evaluated some of our methods, such as a subset of language-independent feature sets in Arabic, we

Strategy	F1-trait-1	F1-trait-2	F1-trait-3	F1-trait-4	F1-trait-5	F1-trait-6
LF	52.39	66.52	68.18	56.30	55.46	49.96
SE	52.85	64.10	60.34	58.23	64.10	49.96
WE	**57.50**	52.79	63.96	56.41	60.86	49.96
BF	56.96	**69.78**	67.25	**62.84**	56.71	49.96
LF-SE-WE-BF (K.I.)	56.47	60.86	**71.10**	62.60	**62.28**	49.96
LF-SE-WE-BF (mode)	57.50	66.52	64.12	61.96	49.78	**49.96**
LF-SE-WE-BF (mean)	51.35	50.12	6.50	54.01	42.15	0.14

Table 4: Macro F1-score of the custom validation split in the second challenge. The traits are, number from 1 to 6, irony, overstatement, rhetorical question, sarcasm, satire, and understatement

Rank	User/Team	F1-sarcastic
1	stce	60.52
2	emma	52.95
3	saroyehun	52.95
41	**UMUTeam**	17.97
42	Matan	16.84
43	abhayshukla9	15.53

Table 5: Results for the first challenge (English)

Rank	User/Team	F1-sarcastic
1	Abdelkader	56.32
2	Aya	50.76
3	rematchka	47.67
22	**UMUTeam**	31.75
23	Pat275	30.13
43	Matan	29.51

Table 6: Results for the first challenge (Arabic)

are aware that our results are limited. Our preliminary experiments with the official annotated test suggests that our major weakness is the class imbalance. As we already include some techniques to address this problem, as weighting the classes and evaluating larger batch sizes, we will explore methods for performing data-augmentation and try to increase the performance of our models.

Acknowledgements

This work is part of the research project LaTe4PSP (PID2019-107652RB-I00) funded by MCIN/AEI/10.13039/501100011033. This work is also part of the research project PDC2021-121112-I00 funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. In addition, José Antonio García-Díaz is supported by Banco Santander and the University of Murcia through the Doctorado Industrial programme.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

James Bergstra, Daniel Yamins, and David Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning*, pages 115–123. PMLR.

María del Pilar Salas-Zárate, Giner Alor-Hernández, José Luis Sánchez-Cervantes, Mario Andrés Paredes-Valverde, Jorge Luis García-Alcaraz, and Rafael Valencia-García. 2020. Review of english literature on figurative language applied to social networks. *Knowledge and Information Systems*, 62(6):2105–2137.

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

José Antonio García-Díaz, Mar Cánovas-García, Ricardo Colomo-Palacios, and Rafael Valencia-García. 2021. Detecting misogyny in spanish tweets. an approach based on linguistics features and word embeddings. *Future Generation Computer Systems*, 114:506–518.

José Antonio García-Díaz, Ricardo Colomo-Palacios, and Rafael Valencia-García. 2022a. Psychographic traits identification based on political ideology: An author analysis study on spanish politicians' tweets posted in 2020. *Future Generation Computer Systems*, 130:59–74.

José Antonio García-Díaz, Salud María Jiménez-Zafra, Miguel Ángel García-Cumbreras, and Rafael Valencia-García. 2022b. Evaluating feature combination strategies for hate-speech detection in spanish using linguistic features and transformers. *Complex & Intelligent Systems*.

José Antonio García-Díaz and Rafael Valencia-García. 2022. Compilation and evaluation of the spanish saticorpus 2021 for satire identification using linguistic features and transformers. *Complex & Intelligent Systems*, pages 1–14.

John S Leggitt and Raymond W Gibbs. 2000. Emotional reactions to verbal irony. *Discourse processes*, 29(1):1–24.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Wei Peng, Achini Adikari, Damminda Alahakoon, and John Gero. 2019. Discovering the influence of sarcasm in social media responses. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(6):e1331.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. 2020. KUISAIL at SemEval-2020 task 12: BERT-CNN for offensive speech identification in social media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2054–2059, Barcelona (online). International Committee for Computational Linguistics.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743.

R2D2 at SemEval-2022 Task 6: Are language models sarcastic enough? Finetuning pre-trained language models to identify sarcasm

Mayukh Sharma, Ilanthenral Kandasamy and **W.B. Vasantha**
School of Computer Science and Engineering
Vellore Institute of Technology
Vellore - 632014, Tamil Nadu, India
04mayukh@gmail.com, ilanthenral.k@vit.ac.in,
vasantha.wb@vit.ac.in

Abstract

This paper describes our system used for SemEval 2022 Task 6: iSarcasmEval: Intended Sarcasm Detection in English and Arabic. We participated in all subtasks based on only English datasets. Pre-trained Language Models (PLMs) have become a de-facto approach for most natural language processing tasks. In our work, we evaluate the performance of these models for identifying sarcasm. For Subtask A and Subtask B, we used simple finetuning on PLMs. For Subtask C, we propose a Siamese network architecture trained using a combination of cross-entropy and distance-maximisation loss. Our model was ranked 7^{th} in Subtask B, 8^{th} in Subtask C (English), and performed well in Subtask A (English). In our work, we also present the comparative performance of different PLMs for each Subtask.

1 Introduction

Sarcasm, a form of figurative speech, allows people to express contempt or mock using irony. The irony is used to communicate the opposite of the intended meaning to express humour or mock something. Sarcasm plays an essential role in people's daily conversation and finds its use across social media to express thoughts. Recently, social media has drawn in millions of users around the world. Owing to its figurative nature, sarcasm poses a significant challenge to systems performing sentiment analysis on these social media platforms. Therefore, it is essential to design and develop systems that efficiently identify sarcasm. The most challenging aspect of sarcasm is the different socio-cultural backgrounds of people who drive it. Therefore, sarcasm intended by the author may not always be perceived by the audience from different backgrounds. Most datasets on sarcasm detection are collected using predefined criteria or human annotators (Oprea and Magdy, 2020). This is a sub-optimal approach that may not always capture the sarcasm intended by the author leading to noise in the models trained on it. SemEval 2022 Task 6: iSarcasmEval: Intended Sarcasm Detection in English and Arabic (Abu Farha et al., 2022) draws attention to the problem of identifying sarcasm using a dataset of intended sarcasm.

Our proposed approach for Subtask A and Subtask B approach uses a classification objective to finetune PLMs. PLMs learn semantic and syntactic features via training on large amounts of a text corpus. This information is used in downstream tasks by simply finetuning task-specific datasets. PLMs have shown remarkable performance on such downstream tasks using the simple finetuning approach (Sharma et al., 2021b). We extend the same idea to identify sarcasm (Subtask A) and identify the type of irony (Subtask B). Subtask C aimed at identifying sarcastic text from its non-sarcastic counterpart. We propose a Siamese network-based architecture using PLMs trained on a combination of cross-entropy and distance-maximisation loss. The classification objective(cross-entropy) identifies the sarcastic/non-sarcastic text, while the distance-maximisation loss maximises the distance between sarcastic/non-sarcastic features learnt by the model during training. We experimented with different PLMs, namely BERT, RoBERTa, MPNet, DeBERTa, to present a comparative study of their performance for the task of sarcasm detection.

Our final submissions for Subtask A and Subtask B used finetuning on MPNet, while we used an ELECTRA-based model for Subtask C. Our proposed system performed well in Subtask B and Subtask C(English), attaining a 7^{th} and 8^{th} rank, respectively on the official leaderboard and performed well in Subtask A. Our experiments show that all PLMs had almost similar performance with slight variation in results. The overall scores were low for Subtask A and Subtask B, indicating that sarcasm detection and identifying the type of irony is a difficult task for PLMs. However, Subtask C results show that the models can efficiently differentiate between a sarcastic text and its non-sarcastic rephrase. Our code available at GitHub[1] for method replicability.

2 Background

Identifying sarcasm is an essential task in Natural Language Processing (NLP). Owing to its figurative nature, it affects the performance of sentiment analysis systems whose performance have significantly improved over the years (Rosenthal et al., 2017) (Sharma et al., 2021a). In social media, sarcasm is mainly used for humour but can hide hateful content, making the identification of sarcasm a vital topic. Methods to deal with sarcasm detection can be separated into two categories, i.e. content-based models and context-based models (Hazarika et al., 2018). Text-based models model the problem as a classification task using pragmatic and lexical features to identify sarcasm. (Riloff et al., 2013) shows sarcasm is expressed as a combination of positive sentiment words and negative situations. Work done in(Joshi et al., 2015) uses the concept of context incongruity for sarcasm detection. Contextual methods use information about the text and the context in which the text is used. (Khattri et al., 2015) uses the sentiment of the tweet as well as the history of the author's previous tweets on similar topics to identify sarcasm. Work done in (Wallace et al., 2015) uses nouns and sentiments presented in a forum towards irony/sarcasm detection. (Hazarika et al., 2018) worked on using both content as well as contextual information for identifying sarcasm. (Castro et al., 2019) presents work done to identify sarcasm from TV shows in a multimodal setting. (Sharma et al., 2020) used a multimodal feature fusion model using attention (Bahdanau et al., 2016) for identifying sarcasm in internet memes. (Felbo et al., 2017) use models trained on emoji using distant supervision to identify sentiment, sarcasm, and emotion. Another challenge in sarcasm detection is the availability of data. Due to the highly subjective nature of sarcasm, it is challenging to collect high-quality data. (Oprea and Magdy, 2020) tries to solve this problem by introducing a dataset of intended sarcasm where the sarcastic data is labelled by the authors removing any noise or ambiguity in labels. The subjective and figurative nature of sarcasm makes it a formidable task, and it poses a challenge for affective systems performing sentiment analysis (Satapathy et al., 2017). Therefore, the task of sarcasm detection is essential to advance state-of-the-art sentiment analysis systems. Moreover, most datasets for sarcasm detection contain much noise and are sub-optimal in capturing the sarcasm intended by the author of the text. SemEval 2022 Task 6: iSarcasmEval: Intended Sarcasm Detection In English and Arabic (Abu Farha et al., 2022) aims to use a dataset of intended sarcasm for identifying sarcasm in text. The task has three subtasks which we define as:

Subtask A (English and Arabic): Given a labelled dataset D of texts, the task aims to learn a classification function that can identify sarcastic/non-sarcastic texts.

Subtask B (English only): For a given labelled dataset D of texts, the objective of the task is to learn a multilabel classification function that can predict the type of irony I where $I \in \{$ Sarcasm, Irony, Satire, Understatement, Overstatement, Rhetorical $\}$.

Subtask C (English and Arabic): Given a dataset D of sarcastic texts and their non-sarcastic rephrase, i.e. both texts convey the same meaning, the objective of the task is to learn a classification function that can identify the sarcastic text from its non-sarcastic rephrase.

Our team participated in Subtask A(English), Subtask B, and Subtask C(English).

Dataset statistics: Table 1 and Table 2 contain the dataset statistics for all Subtasks (English). Dataset statistics for Subtask A and Subtask B show a clear data imbalance problem. To overcome the class imbalance, we used sklearn to compute class weights which are defined as: Let X be the vector containing counts of each class X_i where $i \in X$ and N be the total number of samples. Then the weights for each class were given as: $weight_i = N/(length(X) * X_i)$ where length function computes the number of classes in vector X. There was no imbalance for Subtask C for each sarcastic sample, the corresponding non-sarcastic rephrase was given.

3 System overview

3.1 *Pre-trained language models (PLMs):*

NLP, a diverse field, contains an array of tasks, but most datasets for these tasks contain only a few hundred or thousand human labelled samples. This makes training large models for these tasks

[1]https://github.com/04mayukh/R2D2-at-SemEval-2022-Task-6-iSarcasmEval

Type	Sarcasm	Irony	Satire	Understatement	Overstatement	Rhetorical	Total
Train	677	147	24	9	38	94	823
Validation	147	8	1	1	2	7	44
Test	180	20	49	1	10	11	1400

Table 1: Dataset statistics for SubTask B.

	Subtask A (English)			Subtask C (English)
Type	Sarcastic	Not Sarcastic	Total	Sarcastic/Rephrase
Train	794	2327	3121	780
Validation	73	274	347	87
Test	200	1200	1400	1400

Table 2: Dataset statistics for Subtask A and Subtask C.

a challenging task. Transfer learning using GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2017) is one of the popular choices for solving this problem. Most recently, researchers came up with a method called pre-training (Qiu et al., 2020), which involves training general-purpose models from unannotated text data. This allows models to learn syntactic and semantic features in the text in an unsupervised setting. Transformer architecture proposed in (Vaswani et al., 2017) is the most common choice for training PLMs. These models can be finetuned on various downstream tasks using task-specific datasets. Finetuning allows models to adapt to small task-specific datasets easily and shows promising results (Sharma et al., 2021b). Next, we provide a summary of PLMs used in our approach.

3.2 *Brief overview of used PLMs:*

BERT: It is a bidirectional language model developed by Google that uses transformers. BERT (Devlin et al., 2019) stands for Bidirectional Encoder Representation using Transformer. It uses the auto-encoding modelling technique. It uses Masked Language Modelling (MLM) and the Next Sentence Prediction (NSP) objective for pre-training the model.

RoBERTa: A Robustly Optimized BERT Pre-training Approach was proposed by Facebook in (Liu et al., 2019) and used the BERT architecture with slight modifications to improve its performance. They replaced MLM with dynamic masking and removed the NSP objective during pre-training. They also found that BERT was undertrained, so they trained the model for longer durations with more data and bigger batch size. RoBERTa outperformed BERT on several downstream tasks.

ELECTRA: It was inspired by generative adversarial networks and introduced a new pre-training objective called Replaced Token Detection (RTD) (Clark et al., 2020). Unlike MLM, which introduces <MASK> tokens, ELECTRA replaces specific tokens with plausible fakes. The pre-training objective is to identify if the given token is replaced or the original one. Unlike BERT, where only prediction for the masked token is done, replaced token detection objective is applied to all tokens in ELECTRA, making RTD more efficient than MLM.

MPNet: It was proposed by Microsoft in (Song et al., 2020) and uses a combination of auto-regressive and auto-encoding strategies for pre-training. It solves the problem of MLM in BERT and permuted language modelling in XLNet (Yang et al., 2019) and achieves better performance. It models dependency between tokens using permuted language modelling (vs MLM in BERT) and uses the auxiliary position information to allow the model to see complete sentence reducing position discrepancy (vs permuted language modelling in XLNet). Thus, it uses a combination of masked language modelling and permuted language modelling to jointly model the dependency among predicted tokens and use positional information of complete sentences.

3.3 *Finetuning (Subtask A and Subtask B):*

For Subtask A and Subtask B, we finetuned the pre-trained models defined above. Subtask A was a binary classification task. We added a simple classification head on top of PLMs for Subtask A. It consisted of a 64-neuron dense layer followed by a batch normalisation layer and a final 1-neuron

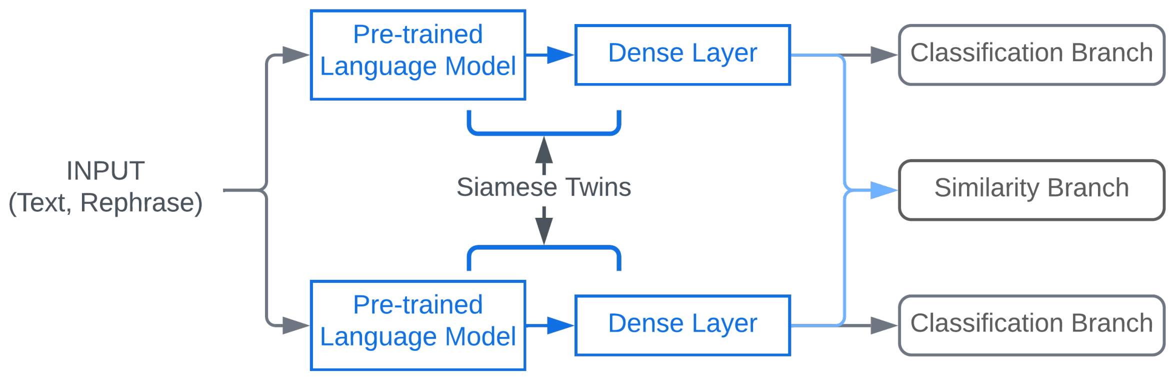


Figure 1: Overview of our Siamese network architecture for Subtask C.

layer with sigmoid activation. Subtask B was a multilabel task aimed at identifying the type of irony. We used a multi-branch model using features from the PLMs, with each branch trying to identify one of the given categories of irony. We used the same classification head for each branch defined in Subtask A.

3.4 *Siamese Network (Subtask C):*

The goal of Subtask C was to identify a sarcastic text from its non-sarcastic rephrase. We use a model based on Siamese network trained using a combination of cross-entropy and distance maximisation loss. Figure 1 shows our Siamese network architecture. A Siamese neural network comprises twin networks that can accept distinct inputs. These twin networks share the same weights, which is known as weight tying (Koch, 2015). Weight tying ensures that the features generated for distinct input are in the same feature space because each network calculates the same function. We use the PLMs to define the twin network of our Siamese architecture. We use the twin network to generate features corresponding to sarcastic text and its non-sarcastic rephrase. Next, these features are separated into two different branches, which we define as:

Distance Maximisation (Similarity branch): Since we want to separate the sarcastic text from its non-sarcastic rephrase, we want to maximise the distance between their features learned by the twin network. We merge the features generated from the twin network using Euclidean distance. It is then passed through a single neuron layer with sigmoid activation, which helps to normalise the distance within a known range of 0-1. We maximise this distance using the distance maximisation loss function, which we define as: Let d_i be the output from the final layer(1-neuron with sigmoid activation), then the loss function L is defined as:

$$L_i = (max[1 - d_i, 0])^2$$

and $L = \sum L_i$ for i belongs to N (Total samples). This ensures that sarcastic/non-sarcastic texts with similar meanings learn different sets of features.

Classification Branch: The primary role of this branch is to classify the generated features as sarcastic/non-sarcastic. It contains two classification heads, each using features generated from the twin network. The classification head comprises a 64-neuron dense layer followed by a batch normalisation layer and a final 1-neuron classification layer with sigmoid activation. Each head independently classifies the features from the respective outputs of the twin network. The classification head with the best performance on the development set was used for making predictions on the test set.

4 Experimental Setup

Text pre-processing: The text was first passed through a pre-processing pipeline to remove noise and normalize into standard features. We removed any website names in the text as they add noise to the data. We also found certain chat words like LOL (laugh out loud) present in the data and converted them into their respective full forms. Emojis are converted to their actual meanings. We used the emoji[2] library for emoji conversion. Lastly, we used ekphrasis (Baziotis et al., 2017) to normalize date, numbers to a standard format and perform spelling correction. PLMs require the text to be tokenized as part of pre-processing step. We use

[2]https://github.com/carpedm20/emoji/

Model	Sarcasm	Irony	Satire	Understatement	Overstatement	Rhetorical	Macro-F1
MPNet*	**.248**	**.032**	**.139**	**.003**	**.0**	**.034**	**.076**
BERT	.219	.042	.126	.0	.031	.022	.074
RoBERTa	.256	.060	.086	.0	.014	.118	.089
ELECTRA	.232	.037	.080	.0	.015	.023	.064

Table 3: Test set results(F1 score) for different PLMs using simple finetuning on Subtask B. (MPNet* was used for the official submission and has been highlighted in bold). The underlined score respresent the best performing models for each sarcasm category.

	Subtask A			Subtask C	
Model	F1 Sarcastic	F1 Macro	Accuracy	F1 Macro	Accuracy
ELECTRA*	.330	.541	.637	**.741**	**.750**
BERT	.323	.522	.605	.763	.765
RoBERTa	.324	.523	.606	.716	.720
MPNet∼	**.3276**	**.5265**	**.610**	.728	.735

Table 4: Test set results for Subtask A and Subtask C. Our final submission for Subtask A was done using MPNet∼ and for Subtask C using ELECTRA*. We have highlighted the official submissions in bold and underlined the individual best metrics across different PLMs.

hugging face's implementation of Fast tokenizers[3] for each pre-trained model. Sequence length was fixed to 70 tokens. Samples greater/smaller than the defined length were truncated or padded.

Data preparation for Siamese network (Subtask C): The dataset for Subtask C consisted of sarcastic texts and their rephrase. We rearranged this data to make sure the input to the Siamese network contains samples in the form of (sarcastic, rephrase) and (rephrase, sarcastic). This is important because the classification layers on top of the Siamese network are used to make predictions on the inputs of the twin network independently using two classification heads. If we do not rearrange the data, each of the two heads will learn to simply predict the output as always sarcastic and non-sarcastic, thereby not learning from training data.

Finetuning: Our approach for all subtasks involves finetuning PLMs and using their features. We used features of [CLS] token for BERT, ELECTRA and start token (<s>) features for MPNet and RoBERTa. These features are then passed to further layers of models as per the architecture we defined above.

Hyperparameters and Training: We developed our models on Keras[4] (Chollet et al., 2015) and used Hugging Face's[5] implementation of transformer[6](Wolf et al., 2020) models. Finetuning was performed on Colab using TPUs. For finetuning we used Adam (Kingma and Ba, 2015) optimiser. We experimented with learning rates ranging from 2e-5 to 5e-5. For Subtask A and Subtask B, we used a binary cross-entropy loss. For Subtask C, we used a binary cross-entropy loss on the classification branch and distance maximization objective on the similarity branch. We finetuned the models for ten epochs and used the weights with the best performance on the development set to make predictions on the test set.

Evaluation metric: Subtask A uses the F1 score of the sarcastic class as an evaluation metric. For Subtask B, the macro averaged F1 score is the official metric, while for Subtask C, accuracy is used as the evaluation metric.

5 Results

Table 3 and Table 4 describe the results of our experiments using different pre-trained models. Our official submission for the task used MPNet for Subtask A, Subtask B and ELECTRA for Subtask C. Our system was ranked 7th in Subtask B, attaining a macro F1 score of 0.076 with the highest F1 score for satire and second highest score for sarcasm and understatement category. Our Siamese architecture-based system also performed well, attaining accuracy of 75% and ranked 8th on the leaderboard for Subtask C.

We performed experiments using BERT, RoBERTa, MPNet, and ELECTRA during the

[3]Hugging Face's Fast Tokenizers
[4]https://keras.io
[5]https://huggingface.co
[6]https://huggingface.co/transformers

evaluation phase. We evaluated these models using the validation set, and the model with the best performance was used to make a submission on the test data. Table 3 and Tabel 4 summarise results using different pre-trained models. We have highlighted the official test submissions in bold while underlined the best performing metric across different models. Subtask A results show that all PLMs have similar performance for sarcasm detection. They perform well, but there is considerable scope for improvement. PLMs are trained on general text corpus making it difficult for them to understand figurative content like sarcasm. For Subtask B, RoBERTa performs better than other PLMs. Results for Subtask B show that it is comparatively easy to identify sarcasm and satire compared to other types of irony, which have very low performance on evaluation metrics. Another reason for this could be the high imbalance in the dataset, making it difficult for models to identify different types of irony. For Subtask C, all models have similar performance with slight variations. The models perform significantly better on evaluation metrics when compared to Subtask A and Subtask B, indicating that models can distinguish between sarcastic and non-sarcastic content having similar meanings.

6 Conclusion

This paper describes our proposed model used for SemEval 2022 Task 6: iSarcasmEval: Intended Sarcasm Detection In English and Arabic. Different PLMs were used to do a comparative analysis of their performance for the sarcasm detection task. Our fine-tuning approach worked well for Subtask B, with the best score for satire and second-best performance for the sarcasm and understatement category. For Subtask C, we proposed a novel Siamese network architecture to identify sarcastic content from it's non-sarcastic rephrase. It performed well, attaining 8^{th} rank on the leaderboard. Our comparative analysis shows that sarcasm detection is a difficult task for the PLMs, and there is scope for further improvements, which we will take up in future works.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

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

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _Obviously_ perfect paper). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4619–4629, Florence, Italy. Association for Computational Linguistics.

François Chollet et al. 2015. Keras. `https://keras.io`.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

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

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. CASCADE: Contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1837–1848, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

Anupam Khattri, Aditya Joshi, Pushpak Bhattacharyya, and Mark Carman. 2015. Your sentiment precedes you: Using an author's historical tweets to predict sarcasm. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 25–30, Lisboa, Portugal. Association for Computational Linguistics.

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

Gregory R. Koch. 2015. Siamese neural networks for one-shot image recognition.

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

Silviu Oprea and Walid Magdy. 2020. iSarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1279–1289, Online. Association for Computational Linguistics.

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

XiPeng Qiu, TianXiang Sun, YiGe Xu, YunFan Shao, Ning Dai, and XuanJing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 704–714, Seattle, Washington, USA. Association for Computational Linguistics.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Ranjan Satapathy, Claudia Guerreiro, Iti Chaturvedi, and Erik Cambria. 2017. Phonetic-based microtext normalization for twitter sentiment analysis. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 407–413.

Mayukh Sharma, Ilanthenral Kandasamy, and W.b. Vasantha. 2020. Memebusters at SemEval-2020 task 8: Feature fusion model for sentiment analysis on memes using transfer learning. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1163–1171, Barcelona (online). International Committee for Computational Linguistics.

Mayukh Sharma, Ilanthenral Kandasamy, and W.B. Vasantha. 2021a. Comparison of neutrosophic approach to various deep learning models for sentiment analysis. *Knowledge-Based Systems*, 223:107058.

Mayukh Sharma, Ilanthenral Kandasamy, and W.b. Vasantha. 2021b. YoungSheldon at SemEval-2021 task 7: Fine-tuning is all you need. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1146–1152, Online. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pretraining for language understanding. In *NeurIPS 2020*. ACM.

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

Byron C. Wallace, Do Kook Choe, and Eugene Charniak. 2015. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1035–1044, Beijing, China. Association for Computational Linguistics.

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

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

SarcasmDet at SemEval-2022 Task 6: Detecting Sarcasm using Pre-trained Transformers in English and Arabic Languages

Malak Abdullah and **Dalya Faraj** and **Safa Swedat**

Jumana Khrais and **Mahmoud Al-Ayyoub**
Computer Science, Jordan University of Science and Technology, Irbid, Jordan
mabdullah@just.edu.jo
dfalnore18, saswedat20, jdikhries162@cit.just.edu.jo
maalshbool@just.edu.jo

Abstract

This paper presents solution systems for task 6 at SemEval2022, iSarcasmEval: Intended Sarcasm Detection In English and Arabic. The shared task 6 consists of three sub-task. We participated in subtask A for both languages, Arabic and English. The goal of subtask A is to predict if a tweet would be considered sarcastic or not. The proposed solution SarcasmDet has been developed using the state-of-the-art Arabic and English pre-trained models AraBERT, MARBERT, BERT, and RoBERTa with ensemble techniques. The paper describes the SarcasmDet architecture with the fine-tuning of the best hyperparameter that led to this superior system. Our model ranked seventh out of 32 teams in subtask A- Arabic with an f1-sarcastic of 0.4305 and Seventeen out of 42 teams with f1-sarcastic 0.3561. However, we built another model to score f-1 sarcastic with 0.43 in English after the deadline. Both Models (Arabic and English scored 0.43 as f-1 sarcastic with ranking seventh).

1 Introduction

In recent years, sarcasm has received remarkable research attention due to its importance and extraordinary impact on society, especially with the significant increase in its use in social media (Ghosh et al., 2020; Van Hee et al., 2018; Faraj and Abdullah, 2021). However, sarcasm is saying something, and what is meant is the opposite. Adding to the difficulty of detecting sarcasm is that all the words in the sentence are positive, but the intended meaning is just the opposite (Channon et al., 2005). Therefore, the detection of sarcasm depends mainly on the general meaning of the sentence, which makes us move away from traditional methods and use artificial intelligence, especially natural language processing.

Natural Language Processing (NLP) is a branch of artificial intelligence that gives machines the ability to read and understand the meanings of humans (Nadkarni et al., 2011). As a result, machines became able to understand the underlying meanings of words and not just rely on keywords. Moreover, it helps reveal forms of speech, such as emotion analysis, humor, ridicule, abuse, etc (Abujaber et al., 2021; Qarqaz et al., 2021).

Task 6 at SemEval-2022, "iSarcasmEval: Intended Sarcasm Detection In English and Arabic" (Abu Farha et al., 2022) suggested three main subtasks in both Arabic and English languages: SubTask A is a binary classification problem that determines whether a tweet is sarcastic or not, SubTask B is a multi-label classification problem which determines the category of tweets sarcastic if it is found in English only, SubTask C is two sentences, one sarcastic and the other paraphrased to be non-sarcastic, define which one is sarcastic.

We only participated in subtask A. Our solution combines four state-of-the-art pre-trained NLP models: AraBERT and MARBERT for Arabic and BERT and RoBERTa for English. SarcasmDet placed seventh out of 32 teams in the subtask A-Arabic and seventh (after the competition deadline) out of 42 teams. We have experimented with the pre-trained language models with different hyperparameters using the simple transformers library. It is worth mentioning that using the ensemble technique has increased our score remarkably.

The paper is constructed as follows: Section 2 provides the related works. Section 3 describes the shared task and the provided dataset. Section 4 describes our system solution. Section 5 shows our experiments. Section 6 provides the results, and finally, the conclusion is in Section 7.

2 Related Work

Sarcastic texts over social media are among the most researched issues of importance for sentiment analysis. Authors of (Rajadesingan et al., 2015) proposed Sarcasm Classification Using a Behavioral modeling Approach. They investigate users' past tweets and use psychological studies to better sarcasm detection of tweets. They used three models to perform their evaluation (Logistic Regression (LR) Decision Tree (DT) and support vector machine (SVM)). The evaluation is performed on a dataset obtained from Twitter and provided on Kaggle with 9104 sarcastic tweets. They used different class distributions (1:1, 10:90, and 20:80) and 10-fold cross-validation techniques. The best performance was obtained from the LR model and class distribution (1:1) with 83.46% accuracy. In 2016, researchers of (Bouazizi and Ohtsuki, 2016) provided pattern-based features approach to detect sarcasm expressions posted on Twitter which extracted with the hashtag sarcasm. They applied four models (Random Forest (RF), SVM, K-Nearest Neighbour (KNN), and Max-Entropy (Max.Ent.)) on three different balanced datasets with different sizes (1000, 2256, and 6000 instances). RF model outperformed the rest with 81.3% in terms of F1-score. In this paper (Pawar and Bhingarkar, 2020), researchers almost did the same work; they used the Pattern-based approach and the same classifiers (RF, SVM, and KNN). The used dataset consists of 9104 sarcastic tweets. RF outperforms the rest in accuracy and F1-score with 81% and 79%, respectively.

The focus on detecting sarcasm in Arabic has emerged in recent years, and researchers have become increasingly interested in this field. In (Karoui et al., 2017), the authors were among the beginning people to supply a dataset to detect sarcasm in Arabic. First, they collected the dataset from Twitter of the Arabic tweets with different Arabic dialects, such as the Egyptian, Syrian, and Saudi dialects. Then, they cleaned the dataset, 5,479 tweets, including 1733 irony, and added four features for each tweet: surface, sentiment, shifter, and internal context features. Finally, they applied several machine learning algorithms, and experiments showed that the Random Forest classifier achieved a high accuracy of 72.76% for detecting sarcasm in Arabic tweets.

On the other hand, another dataset is provided for sarcasm in Arabic by (Farha and Magdy, 2020). This dataset contains three topics: sarcasm, sentiment, and dialect labels include 10,547 rows of tweets where 16% are sarcastic tweets. The researchers applied a Bidirectional LSTM deep learning approach (biLSTM) to achieve an F1-score of 0.46, which indicates the difficulty of detecting sarcasm in the Arabic language.

The researchers in (Faraj and Abdullah, 2021) participated in the WANLP 2021 Shared Task for subtask 1 (Sarcasm Detection) competition. First, they used the AraSarcasm-v2 dataset and then cleaned data by using NLTK library (Bird, 2006), such as normalized and removed emojis, links, and Html tags. Next, they implemented several pre-trained models such as AraBERt, multilingual BERT cased and uncased, and XLM-Roberta. Their solution, called SarcasmDet, is based on fine-tuning of the large AraBERT and base AraBERT, and they got first place with an accuracy of 0.7830 and fourth place with f1-sarcastic 0.5989.

3 Task and Data Description

In Shared Task on SemEval 2022 - Task 6 (iSarcasmEval): Intended Sarcasm Detection In English and Arabic (Abu Farha et al., 2022), has three subtasks in two languages: Arabic and English, and each task solves different requirements. We participated in SubTask A: a binary classification problem that determined whether a tweet contains sarcasm or not.

The dataset consists of nine columns: the tweet, the rephrase, sarcastic, which defines whether the tweet is sarcastic or non-sarcastic, irony, stair, understatement, overstatement, and the rhetorical question. The last five columns are the type of sarcasm contained in the tweets. Moreover, the dataset contains 3467 non-null rows. Regarding our task, a binary sarcasm classification of the tweets, the tweet, and the sarcasm are the only required columns to solve the task. Table 1 shows a sample of the utilized dataset in English subtask A.

Table 1: Sample of English data

Tweet	**Sarcastic**
The only thing I got from college is a caffeine addiction	1
I love it when professors draw a big questionmark next to my answer on an exam becauseI'm always like yeah I don't either	1
The population spike in Chicago in 9 months is about to be ridiculous	0
You'd think in the second to last English classof the year my prof would stop calling meSean	0

The data is highly imbalanced as there are 2600 tweets classified as non-sarcastic and 867 as sarcastic.

id	tweet	Sarcastic	rephrase	dialect
3	اية المهلبية دي يصحي	1	ما هذا الجمال	nile
2805	أينما وجد العدل فثم شرع الله	0	NaN	msa

Table 2: sample from the training Arabic dataset

The Arabic training dataset contains 3102 tweets and five columns: id, tweet, sarcastic, rephrase, and dialect. They have categorized the text into sarcastic or non-sarcastic based on the author himself. Table 2 shows an example of training the Arabic dataset for subtask A- Arabic. The Arabic dataset in subtask A- Arabic is imbalanced due to class sarcasm and a clear difference between sarcastic and non-sarcasm tweets. The number of sarcastic tweets was 745, and the number of non-sarcastic tweets was 2357.

4 SarcasmDet Description

Texts are sequential, so they must be trained by models supporting data in which the order of its features is an important factor. Transformers are deep learning techniques that utilize the idea of self-attention mechanism (Potamias et al., 2020). In this work, two transformer-based pre-trained models are fine-tuned to achieve the requirements of the sarcasm detection task for each language. The two models are the BERT and the RoBERTa for English, where we combined two pre-trained models, AraBERT and MARBERT, for Arabic.

Subtask A - English Figure 1 shows the architecture of the proposed model for English. The ensemble technique is adopted by performing a weighted sum for the predictions of BERT and RoBERTa. One BERT machine with weight one and four RoBERTa machines, each with weight one, is deployed except for one machine with a weight of 0.5.

Regarding the BERT model, the dataset is tokenized by a pre-defined tokenizer of the model. The BERT model originally consists of 12 layers, the first ten layers are chosen to be untrainable and the last two layers to be trainable. Moreover, the model is trained using a batch size equal to 6 and the number of epochs equal to 3. As to the RoBERTa model, the dataset is tokenized using its pre-defined tokenizer. The layers are fine-tuned. The first nine layers are untrainable, the last three layers are trainable, and the model is trained using a batch size equal to 12 and the number of epochs equal to 3.

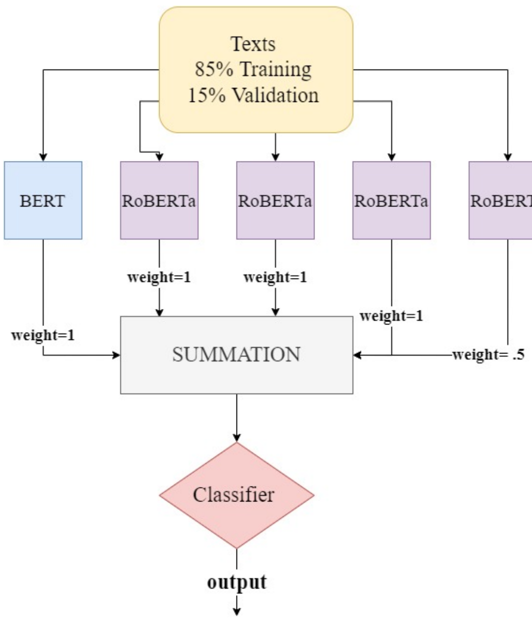


Figure 1: Architecture of SarcasmDet-English

Subtask A - Arabic The ensemble is a technique of combining several different models in the prediction process. In addition, we used hard voting that depends on the highest vote from all the model predictions. We combined two pre-trained models, AraBERT and MARBERT. The Arabic pre-trained model called AraBERT (Antoun et al.). AraBert is a pre-trained model that focuses directly on the Arabic language, and it is based on the BERT architecture (Devlin et al., 2018). There are two versions of AraBERT(v01 and v02). The first version, AraBERT-v01, was trained on 77M sentences, with a size of 23GB and 2.7B of words. The second version, AraBERT-v02, was trained on 200M sentences with 77GB and 8.6B words. MARBERT (Abdul-Mageed et al., 2020) is also based on the BERT architecture without the next sentence prediction and focuses on Dialectal Arabic and MSA.

In SarcasmDet, tweets are fed to the AraBERT and MARBERT. Next, we added the final layer, which is fine-tuning with the best hyperparameters as shown in **table 3** to classify the Arabic tweet into a sarcastic tweet or not. Then we applied the

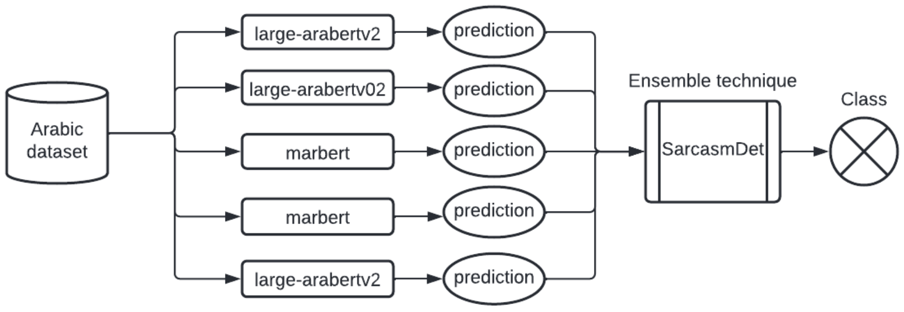


Figure 2: Architecture of SarcasmDet-Arabic

hard voting technique to select the final label in all tweets, as shown in Figure 2.

Model	LR	Epoch	Batch Size
large-arabertv2	2e-5	5	8
marbert	1e-5	5	8

Table 3: Best the hyperparameter that we used in our models

5 Experiments

5.1 English

Unfortunately, the number of sarcastic sentences is much smaller than the number of the non-sarcastic ones, so using the accuracy to evaluate the model performance is not a good choice. Thus, the metric used to describe the version of the model is the F1-Score. F1-score is the harmonic mean of the precision and the recall. It is the average of the model's ability to find all sarcastic sentences and how accurate it is when classifying sentences as sarcastic.

The transformer-based models, BERT and RoBERTa, are trained using different hyperparameters. Table 4 and Table 5 show the results of the models with different fine-tuning.

Table 4: Result of Bert Model

BERT Parameters	Results F1-sarcastic
Batch size: 6 Number of epochs: 3 Number of trainable layers:4	0.38
Batch size: 6 Number of epochs: 3 Number of trainable layers:1	0.35
Batch size: 6 Number of epochs: 3 Number of trainable layers:2	0.39

Table 5: Result of RoBert Model

RoBERTa Parameters	Results f1-sarcastic
Batch size: 6 Number of epochs: 3 Number of trainable layers:3	0.36
Batch size: 4 Number of epochs: 2 Number of trainable layers:4	0.29
Batch size: 7 Number of epochs: 3 Number of trainable layers:4	0.37

5.2 Arabic

We have experimented with fine-tuning for two pre-trained models: AraBERT and MARBERT with different hyperparameters. Also, we attempted to increase the dataset using the augmentation technique with the ArSarcasm-v2 dataset to improve the results. We used the data set after the increment and fed the tweet to AraBERT and MARBERT, then fine-tune with hyperparameters to classify the Arabic tweet into a sarcastic tweet or not. **table 6** shows the hyper-parameters we have used in our experiments for the tested models.All of the models have been implemented using the HuggingFace library and SimpleTransforner pre-trained package.

6 Results

6.1 English

The model achieved a recall value equal to 0.37, a precision value equal to .51, and an F1-Score equal to 0.43. Table 7 shows the parameters and the results of the BERT model, RoBERTa model, and the ensemble model. It is noteworthy that the higher F1-Score in the SemEval competition in subtask A of task 6 is 0.6052, and the seventh rank is 0.4342. So the proposed architecture results, which is 0.4322, are equivalent to the eighth rank

Experiment	Model	LR	Epoch	Batch Size
1	large-arabertv2	1e-5	5	8
2	large-arabertv2	2e-5	5	8
3	large-arabertv02	2e-5	5	8
4	marbert	2e-5	5	8
5	marbert	1e-5	5	8
6	large-arabertv2	2e-5	5	8
7	marbert	1e-5	5	8

Table 6: all the hyper-parameter that we used in our experiments

in the competition.

Table 7: Best Parameters of BERT, RoBERTa and Ensembling.

Model	Parameters	Results (F1-Score)
BERT	Batch size: 6 Number of epochs: 3 Number of trainable layers: 2	.39
RoBERTa	Batch size: 12 Number of epochs: 3 Number of trainable layers: 3	.41
Ensemble	BERT model with weight :13 RoBERTa models with weight:11 RoBERTa with weight : .5	.43

6.2 Arabic

First, we applied AraBERT and MARBERT with different finetuning. AraBERT outperformed on MARBERT with a score of an f1-sarcastic 0.4255. Then, we augmentation the dataset and applied AraBERT and MARBERT, in this case, MARBERT outperformed AraBERT with a score of an f1-sarcastic 0.4065, although the data size increased, this did not lead to an improvement in the results, the best result was for ARABERT using Arabic data for the task. But SarcasmDet using the ensemble technique specifically hard vote significantly outperformed both AraBERT and MARBERT with an f1-sarcastic score of 0.4304, Table **table 8** shows the organizers' final result and table **table 9** shows all experiments that we have implemented.

7 Conclusion

Sarcasm is an influential issue in human life, whether written or spoken. However, sarcasm detection in texts is a challenging task. This paper presented our model SarcasmDet for solving subtask A- English and Arabic in task6 at SemEval 2021 - iSarcasmEval: Intended Sarcasm Detection in English and Arabic. SarcasmDet is based on the fine-tuning of two pre-trained NLP models for each language and then applied ensemble technique to improve the model. The models trained on a dataset obtained from a competition SemEval 2022 subtask A of task 6. For English subtask A, the dataset consists of 3467 records, 866 of them are sarcastic, and the rest are non-sarcastic. The results showed that the RoBERTa model outperformed the BERT model, where the Ensembling technique outperformed both in the f1-sarcastic score, which was 0.43. On the other hand, the Arabic dataset consists of 8K tweets divided into training and testing sets. Our solution SarcasmDet is ranked 7th out of 32 teams with an f1-sarcastic score of 0.4305.

8 Acknowledgement

We gratefully acknowledge the Deanship of Research at the Jordan University of Science and Technology (JUST) for supporting this work via Grant #20210200.

References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2020. Arbert & marbert: deep bidirectional transformers for arabic. *arXiv preprint arXiv:2101.01785*.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Dia Abujaber, Ahmed Qarqaz, and Malak A Abdullah. 2021. Lecun at semeval-2021 task 6: Detecting persuasion techniques in text using ensembled pretrained transformers and data augmentation. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1068–1074.

Wissam Antoun, Fady Baly, and Hazem Hajj. Arabert: Transformer-based model for arabic language under-

Rank	F1-sarcastic	F-score	Precision	Recall	Accuracy
7th	0.4305	0.6114	0.6240	0.7412	0.6957

Table 8: Official results on subtask 1 (sarcasm Detection) test set.

Experiment	Model	F1-sarcastic
1	large-arabertv2	0.3847
2	large-arabertv2	0.4255
3	large-arabertv02	0.4166
4	marbert	0.3650
5	marbert	0.4015
6	large-arabertv2	0.3276
7	marbert	0.4065
8	SarcasmDet	0.4305

Table 9: Results in all experiments, SarcasmDet is an ensemble for experiments 2,3,4,5 and 7

standing. In *LREC 2020 Workshop Language Resources and Evaluation Conference 11–16 May 2020*, page 9.

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72.

Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488.

Shelley Channon, Asa Pellijeff, and Andrea Rule. 2005. Social cognition after head injury: Sarcasm and theory of mind. *Brain and Language*, 93:123–134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dalya Faraj and Malak Abdullah. 2021. Sarcasmdet at sarcasm detection task 2021 in arabic using arabert pretrained model. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 345–350.

Ibrahim Abu Farha and Walid Magdy. 2020. From arabic sentiment analysis to sarcasm detection: The arsarcasm dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 32–39.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. *arXiv preprint arXiv:2005.05814*.

Jihen Karoui, Farah Banamara Zitoune, and Veronique Moriceau. 2017. Soukhria: Towards an irony detection system for arabic in social media. *Procedia Computer Science*, 117:161–168.

Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. 2011. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551.

Neha Pawar and Sukhada Bhingarkar. 2020. Machine learning based sarcasm detection on twitter data. In *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, pages 957–961. IEEE.

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas-Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Ahmed Qarqaz, Dia Abujaber, and Malak Abdullah. 2021. R00 at nlp4if-2021 fighting covid-19 infodemic with transformers and more transformers. In *Proceedings of the Fourth Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 104–109.

Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 97–106.

Cynthia Van Hee, Gilles Jacobs, Chris Emmery, Bart Desmet, Els Lefever, Ben Verhoeven, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2018. Automatic detection of cyberbullying in social media text. *PloS one*, 13(10):e0203794.

JCT at SemEval-2022 Task 6-A: Sarcasm Detection in Tweets Written in English and Arabic using Preprocessing Methods and Word N-grams

Yaakov HaCohen-Kerner, Matan Fchima, Ilan Meyrowitsch

Department of Computer Science, Jerusalem College of Technology, Lev Academic Center
21 Havaad Haleumi St., P.O.B. 16031, 9116001 Jerusalem, Israel
kerner@jct.ac.il, matanf1992@gmail.com, meyrowitsch@gmail.com

Abstract

In this paper, we describe our submissions to SemEval-2022 contest. We tackled subtask 6-A - "iSarcasmEval: Intended Sarcasm Detection In English and Arabic – Binary Classification". We developed different models for two languages: English and Arabic. We applied 4 supervised machine learning methods, 6 preprocessing methods for English and 3 for Arabic, and 3 oversampling methods. Our best submitted model for the English test dataset was an SVC model that balanced the dataset using SMOTE and removed stop words. For the Arabic test dataset, our best submitted model was an SVC model that preprocessed removed longation.

1 Introduction

The rapid development of various types of social networks allows the development of increasingly offensive language in general and sarcasm in particular. Sarcasm is the use of words that mean the opposite of what you want to say especially to insult someone, show irritation, or be funny[1].

Sarcastic language can harm individuals or groups of people and may cause harmful effects on society. Thus, it is important to develop high-quality systems capable of detecting sarcastic expressions automatically.

Sarcasm detection is a difficult task not only for a computer but even for a human being. High-quality performance of this task requires understanding the context of the situation, the relevant culture, and in some cases the specific issue or people involved in this situation (Maynard and Greenwood, 2014).

Moreover, the noisy nature of social media texts especially Twitter messages make the detection task even harder. Therefore, it is an interesting and challenging task to detect sarcasm using supervised machine learning (ML) methods and natural language processing (NLP) tools.

Furthermore, Rosenthal et al. (2014) show a significant drop in sentiment polarity classification performance when processing sarcastic tweets, compared to non-sarcastic ones.

In this paper, we describe our research and participation in subtask 6-A for sarcasm detection in tweets written in two languages: English and Arabic. The full description of task 6 in general and 6-A in particular including the datasets and the participating teams is given in Abu Farha et al. (2022).

The structure of the rest of the paper is as follows. Section 2 introduces a background concerning sarcasm detection, text preprocessing, and text classification with imbalanced classes. Section 3 describes subtask 6-A and its datasets. In Section 4, we present the submitted models and their experimental results. Section 5 summarizes and suggests ideas for future research.

2 Related Research

Most prior textual sarcasm detection datasets have been annotated by using either manual labeling or a weak supervision method. In the first approach, sarcasm labels are provided by human annotators (e.g., Filatova, 2012; Riloff et al., 2013; Abercrombie and Hovy, 2016). However, such labels represent annotator perception, which

[1] www.merriam-webster.com. Retrieved 9-Feb-2022.

may differ from the author intention, as further pointed out by Oprea and Magdy (2020).

In the second approach, texts are labeled as sarcastic if they meet predefined criteria, e.g., including specific tags (e.g. #irony, #sarcastic, #sarcasm) (Ptáček et al., 2014; Khodak et al., 2018). However, this method can lead to noisy labels because of various reasons (Oprea and Magdy, 2020).

To overcome these disadvantages, Oprea and Magdy (2019) introduced another method. In their method, the sarcasm labels for texts are provided by the authors themselves.

Most of the sarcasm detection studies are in the English language (Campbell and Katz, 2012; Riloff et al., 2013; Bamman and Smith, 2015; Rajadesingan et al., 2015; Wallace et al., 2015; Amir et al., 2016; Joshi et al., 2016; Hazarika et al., 2018; Oprea and Magdy, 2019).

There are also some studies in Arabic (Karoui et al., 2017; Ghanem et al.,2019; Abbes et al., 2020; Abu-Farha and Magdy, 2020).

For more information about various issues concerning sarcasm detection please refer to survey papers such as Joshi et al. (2017), Sarsam et al. (2020), Verma et al. (2021), and Moores and Mago (2022).

2.1 Text preprocessing

Text preprocessing is an important step in many NLP domains such as ML, sentiment analysis, text mining, and text classification (TC). In text documents in general and in social text documents in particular, it is common to find various types of noise, e.g., typos, emojis, slang, HTML tags, spelling mistakes, and repetitive letters. Analysis of text that has not been carefully cleaned or preprocessed might lead to misleading results.

Not all of the preprocessing types are considered effective for TC tasks. For instance, HaCohen-Kerner et al. (2008) demonstrated that the use of word unigrams including stop words leads to improved results compared to the results obtained using word unigrams excluding stop words.

HaCohen-Kerner et al. (2019) investigated the impact of all possible combinations of six preprocessing methods (spelling correction, HTML tag removal, converting uppercase letters into lowercase letters, punctuation mark removal, reduction of repeated characters, and stopword removal) on TC in three benchmark mental disorder datasets. In another study, HaCohen-Kerner et al. (2020) explored the influence of various combinations of the same six basic preprocessing methods mentioned in the previous paragraph on TC in four general benchmark text corpora using a bag-of-words representation. The general conclusion was that it is always advisable to perform an extensive and systematic variety of preprocessing methods, combined with TC experiments because this contributes to improving TC accuracy.

2.2 Text classification with imbalanced classes

The problem of TC with imbalanced classes is that there are too few examples of the minority class to effectively learn a good predictive TC model. There are various methods to cope with this problem. The main idea is to change the dataset until a more balanced distribution is reached. Two well-known sampling methods that enable such a change are oversampling and undersampling. Random oversampling means randomly duplicating examples in the minority class. Random undersampling means randomly deleting examples in the majority class.

An additional frequent method is to generate synthetic samples which means randomly sampling the attributes from instances in the minority class. There are several algorithms that support the generation of synthetic samples. The most popular is called the Synthetic Minority Oversampling Technique (SMOTE) (Chawla, 2002). This method is an oversampling method that creates synthetic samples from the minor class instead of creating copies. This method selects two or more similar instances and perturbs an instance one attribute at a time by a random amount within the difference to the similar instances. We used also two other oversampling methods BorderlineSMOTE and ADASYN that work similarly.

Readers interested in expanding and deepening the topic of solutions to TC with imbalanced classes are referred to the following articles (Chawla et al., 2002; He and Ma, 2013; Krawczyk, 2016; Brownlee, 2020, Shaikh et al., 2021).

3 Task and Datasets Description

Tables 1-4 present various statistical details about the training and test sets for English and Arabic**.**

The analysis of the details presented in Tables 1 and 2 show that the training and test sets for English are highly imbalanced, with a ratio of about 25:75 and 14:86 (non-sarcastic:sarcastic), respectively. We tried to balance the dataset in our experiments using the oversampling methods that we have mentioned above.

	Sarcastic	Not sarcastic	Total
Documents	867	2,600	3,467
% Docs	25.008%	74.992%	100%
words	15,863	49,432	65,295
characters	86,932	275,172	362,104
avg word per doc	18.296	19.012	18.8333
avg chars per doc	100.267	105.835	104.443
words std	10.235	11.595	11.275
chars std	57.545	65.504	63.653

Table 1: details of the training set for English.

	Sarcastic	Not sarcastic	Total
Documents	200	1,200	1,400
% Docs	14.286%	85.714%	100%
words	4,455	18,505	22,960
characters	23,828	99,024	122,852
avg word per doc	22.275	15.42	16.4
avg chars per doc	119.14	82.52	87.751
words std	12.687	8.861	9.800
chars std	66.224	48.322	52.841

Table 2: details of the test set for English.

	Sarcastic	Not sarcastic	Total
Documents	1,490	2,357	3,847
% Docs	38.732%	61.268%	100%
words	6,370	36,119	42,489
characters	32,858	205,286	238,144
avg word per doc	8.55	15.324	13.697
avg chars per doc	44.104	87.096	76.771
words std	3.924	6.428	6.593
chars std	21.166	36.797	38.389

Table 3: details of the training set for Arabic.

	Sarcastic	Not sarcastic	Total
Documents	200	1,200	1,400
% Docs	14.286%	85.714%	100%
words	1,454	7,545	8,999
characters	7,430	37,772	45,202
avg word per doc	7.27	6.287	6.427
avg chars per doc	37.15	31.476	32.287
words std	4.190	3.574	3.685
chars std	22.094	19.64	20.107

Table 4: details of the test set for Arabic.

The analysis of the details presented in Tables 3 and 4 show a similar picture. The training and test sets for Arabic are highly imbalanced, with a ratio of about 39:61 and 14:86 (non-sarcastic:sarcastic), respectively. Also, for Arabic, we tried to balance the dataset in our experiments using the oversampling methods that we have mentioned above. Both for English and Arabic, the test datasets are even more imbalanced than the compatible training datasets.

4 The Submitted Models and Experimental Results

We applied 4 supervised ML methods on the training datasets: Random Forest (RF), Support Vector Classifier (SVC), Logistic regression (LR), and Decision Tree (DT).

RF is an ensemble learning method for classification and regression (Breiman, 2001). Ensemble methods use multiple learning algorithms to obtain improved predictive performance compared to what can be obtained from any of the constituent learning algorithms. RF operates by constructing a multitude of decision trees at training time and outputting classification for the case at hand. RF combines Breiman's "bagging" (Bootstrap aggregating) idea in Breiman (1996) and a random selection of features introduced by Ho (1995) to construct a forest of decision trees.

SVC is a variant of the support vector machine (SVM) ML method (Cortes and Vapnik, 1995) implemented in SciKit-Learn. SVC uses LibSVM (Chang & Lin, 2011), which is a fast implementation of the SVM method. SVM classifies vectors in a feature space into one of two sets, given training data. It operates by constructing the optimal hyperplane dividing the two sets, either in the original feature space or in higher dimensional kernel space.

LR (Cox, 1958; Hosmer et al., 2013) is a linear classification model. It is known also as maximum entropy regression (MaxEnt), logit regression, and the log-linear classifier. In this model, the probabilities describing the possible outcome of a single trial are modeled using a logistic function.

DT (Song and Ying, 2015) is a flowchart-like structure method in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes).

These ML methods were applied using the following tools and information sources:

- The Python 3.7.3 programming language[2].
- Scikit-learn – a Python library for ML methods[3].
- Numpy – a Python library that provides fast algebraic calculous processing, especially for multidimensional objects[4].

We applied six preprocessing methods for English:

1. change to lower-case
2. stop words removal
3. numbers removal
4. emojis removal
5. HTML tags removal
6. punctuations removal

For Arabic we applied three preprocessing methods:

1. punctuations removal
2. Tashkeel removal[5]
3. longation removal[6,7]

We applied three oversampling methods to balance the data:

4. SMOTE
5. BorderlineSMOTE
6. ADASYN

Tables 5 and 6 present the F1-scores over the PCL class of our models for English and Arabic, respectively. The analysis of the details presented in Tables 5 and 6 show that there is a big impact on the longation in the Arabic language, and removing it improved the models. In the English language, we do not see a specific preprocess method that is the most effective method (mark in bold – the three highest results in each language for different number of word unigrams).

Tf-idf	**balance method**	**model**	**Train test**	**preprocessing methods**	**F1-score**
500	SMOTE	SVC	70-30	SW, LC	**0.413**
	SMOTE	SVC	80-20	SW, LC, Only words	0.413
	ADASYN	LR	70-30	SW	0.412
1000	BorderlineSMOTE	LR	80-20	SW, LC, Only words	**0.433**
	ADASYN	SVC	70-30	NONE	0.425
	BorderlineSMOTE	SVC	80-20	SW, LC, Only words	0.424
2000	ADASYN	SVC	80-20	Only words	**0.429**
	SMOTE	SVC	80-20	SW, LC, Only words	0.410
	BorderlineSMOTE	LR	80-20	SW	0.407
3000	BorderlineSMOTE	LR	80-20	SW, LC	0.407
	BorderlineSMOTE	SVC	80-20	SW, LC	0.403
	ADASYN	LR	80-20	SW, LC	0.402
4000	ADASYN	LR	70-30	SW, LC	0.410
	SMOTE	SVC	80-20	SW, Only words	0.408
	BorderlineSMOTE	SVC	80-20	SW, LC, Only words	0.401
5000	SMOTE	LR	80-20	NONE	0.401
	BorderlineSMOTE	LR	70-30	LC	0.4
	SMOTE	SVC	80-20	SW	0.39

Table 5: F1-scores over the PCL class of our models for English.

[2] https://www.python.org/downloads/release/python-373/
[3] https://scikit-learn.org/stable/index.html
[4] https://numpy.org
[5] https://github.com/motazsaad/process-arabic-text
[6] https://github.com/bakrianoo/aravec
[7] https://medium.com/analytics-vidhya/sentiment-analysis-of-arabic-text-data-tweets-4e96c8da892b

Tf-idf	balance method	model	Train test	preprocessing methods	F1-score
500	BorderlineSMOTE	LR	80-20	RL	**0.690**
	ADASYN	SVC	80-20	RL	0.682
	NONE	SVC	80-20	RL	0.679
1000	ADASYN	LR	80-20	RT, RL	**0.679**
	NONE	SVC	80-20	RL	0.679
	NONE	SVC	80-20	RP, RL	0.672
2000	ADASYN	LR	80-20	RL	**0.698**
	ADASYN	SVC	80-20	RL	0.685
	NONE	SVC	80-20	RL	0.679
3000	NONE	SVC	80-20	RL	0.679
	NONE	SVC	80-20	RP, RL	0.672
	NONE	SVC	80-20	RT, RL	0.670
4000	SMOTE	LR	80-20	RT, RL	0.679
	NONE	SVC	80-20	RL	0.679
	ADASYN	SVC	80-20	RP, RT	0.676
5000	NONE	SVC	80-20	RL	0.679
	NONE	SVC	80-20	RP, RL	0.672
	NONE	SVC	80-20	RT, RL	0.670

Table 6: F1-scores over the PCL class of our models for Arabic.

Table 7 presents the F1-Scores over the PCL class of our best models for English and Arabic that were submitted to the competition. The F1-scorer over the PCL class on the training dataset of our best model for English and Arabic were 0.4183 and 0.6791 respectively while the F1-scores over the PCL class on the test dataset of our best model were only 0.215 and 0.29515 respectively.

Possible explanations might be: (1) The training dataset is different in its balance rate than the balance rate of the competition test datasets and (2) the content of a relatively high number of Tweeter messages in the competition test dataset are fundamentally different from the content of the Twitter messages in the training dataset.

5 Summary and Future Research

In this paper, we describe our models and submissions to subtask 6-A of SemEval-2022: sarcasm detection in Twitter messages written in English and Arabic using preprocessing methods and word n-grams.

Language	Model name and split mode	ML Method	Applied text preprocessing and oversampling methods	F1-score over the PCL class on the training dataset	F1-score over the PCL class on the test dataset	Place in the competition
English	Ilan_SVC 80-20	SVC	Balance with SMOTE, removed stop words	0.418	0.215	33-35
Arabic	Matan_SVC 80-20	SVC	No balance method, removed longation	0.679	0.295	24

Table 7: F1-scores over the PCL class of our best models for English and Arabic that were submitted to the competition.

We applied 4 supervised ML methods, 6 preprocessing methods for English and 3 for Arabic, and 3 oversampling methods.

Our best submitted model for the English test dataset was an SVC model that balanced the dataset using SMOTE and removed stop words. For the Arabic test dataset, our best submitted model was an SVC model that preprocessed longation removal.

There are various ideas for future research that are connected to the nature of Twitter messages as follows: (1) the use of skip character n-grams because they serve as generalized n-grams that allow us to overcome problems such as noise and sparse data (HaCohen-Kerner et al., 2017), which are common to Twitter messages and (2) Many Twitter messages contain acronyms. Acronym disambiguation might enable better classification (HaCohen-Kerner et al., 2010A).

Another idea that may lead to better classification is to use additional feature sets such as stylistic feature sets (HaCohen-Kerner et al., 2010B).

References

Ines Abbes, Wajdi Zaghouani, Omaima El-Hardlo, and Faten Ashour. 2020. DAICT: A dialectal Arabic irony corpus extracted from Twitter. In Proceedings of the 12th Language Resources and Evaluation Conference, pages 6265–6271, Marseille, France. European Language Resources Association.

Gavin Abercrombie and Dirk Hovy. 2016. Putting Sarcasm Detection into Context: The Effects of Class Imbalance and Manual Labeling on Supervised Machine Classification of Twitter Conversations. In Proceedings of the ACL 2016 Student Research Workshop, pages 107–113. ACL.

Ibrahim Abu Farha, Silviu Oprea, Steve Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In the 16th International Workshop on Semantic Evaluation (SemEval-2022), Association for Computational Linguistics.

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mario J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. In CoNLL, pages 167–177. ACL.

David Bamman and Noah A. Smith. 2015a. Contextualized sarcasm detection on twitter. In ICWSM, pages 574–577. AAAI Press.

Leo Breiman. 1996. Bagging predictors. Machine learning 24(2), 123-140.

Leo Breiman. 2001. Random forests. Machine learning 45(1), 5-32.

Jason Brownlee. 2020. Imbalanced classification with python: Better metrics, balance skewed classes, cost-sensitive learning. Machine Learning Mastery.

Chih-Chung Chang, and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. ACM transactions on intelligent systems and technology (TIST) 2(3), 1-27.

Nitesh V. Chawla, Kevin W Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority oversampling technique. Journal of artificial intelligence research, 16, 321-357.

John D. Campbell and Albert N. Katz. 2012. Are there necessary conditions for inducing a sense of sarcastic irony? Discourse Processes, 49(6), 459–480.

Corinna Cortes and Vladimir Vapnik. 1995.Support-vector networks. Machine learning 20.3 : 273-297.

David R. Cox. 1958. The regression analysis of binary sequences, Journal of the Royal Statistical Society: Series B (Methodological), 20, 215–232.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In LREC. ELRA.

Bilal Ghanem, Jihen Karoui, F. Benamara, Veronique ´ Moriceau, and P. Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in Arabic tweets. Proceedings of the 11th Forum for Information Retrieval Evaluation.

Yaakov HaCohen-Kerner, Ariel Kass, and Ariel Peretz. 2008. Combined one sense disambiguation of abbreviations. In Proceedings of ACL-08: HLT, Short Papers, Association for Computational Linguistics, pages 61-64, Columbus, Ohio, Association for Computational Linguistics. URL: https://aclanthology.org/P08-2.

Yaakov HaCohen-Kerner, Ariel Kass, and Ariel Peretz. 2010A. HAADS: A Hebrew Aramaic abbreviation disambiguation system. Journal of the American Society for Information Science and Technology, 61(9), 1923-1932.

Yaakov HaCohen-Kerner, Hananya Beck, Elchai Yehudai, and Dror Mughaz. 2010B. Stylistic feature sets as classifiers of documents according to their historical period and ethnic origin. Applied Artificial Intelligence, 24(9), 847-862.

Yaakov HaCohen-Kerner, Ziv Ido, and Ronen Ya'akobov. 2017. Stance classification of tweets using skip char Ngrams. In Joint European

Conference on Machine Learning and Knowledge Discovery in Databases (pp. 266-278). Springer, Cham.

Yaakov HaCohen-Kerner, Yair Yigal, and Daniel Miller. 2019. The impact of Preprocessing on Classification of Mental Disorders, in Proc. of the 19th Industrial Conference on Data Mining, (ICDM 2019), New York.

Yaakov HaCohen-Kerner, Daniel Miller, and Yair Yigal. 2020. The influence of preprocessing on text classification using a bag-of-words representation, PloS one, vol. 15, p. e0232525.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. In COLING, pages 1837–1848. ACL.

Haibo He and Yunqian Ma (Eds.). 2013. Imbalanced learning: foundations, algorithms, and applications.

David W. Hosmer, Stanley Lemeshow, and Rodney X. Sturdivant. 2013. Applied logistic regression, Vol. 398, John Wiley & Sons.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In IJCNLP, pages 757–762. ACL.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J. Carman. 2017. Automatic sarcasm detection: A survey. ACM Computing Surveys (CSUR), 50(5), 1-22.

Jihen Karoui, Farah Banamara Zitoune, and Veronique ´ Moriceau. 2017. SOUKHRIA: Towards an irony detection system for Arabic in social media. Procedia Computer Science, 117:161–168. Arabic Computational Linguistics

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018. A large self-annotated corpus for sarcasm. In LREC. ELRA.

Bartosz Krawczyk. 2016. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, *5*(4), 221-232.

Diana G. Maynard and Mark A. Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In Lrec 2014 proceedings. ELRA.

Bleau Moores and Vijay Mago. 2022. A Survey on Automated Sarcasm Detection on Twitter. arXiv preprint arXiv:2202.02516.

Sarang Shaikh, Sher Muhammad Daudpota, Ali Shariq Imran, and Zenun Kastrati. 2021. Towards improved classification accuracy on highly imbalanced text dataset using deep neural language models. Applied Sciences, 11(2), 869.

Samer Muthana Sarsam, Hosam Al-Samarraie, Ahmed Ibrahim Alzahrani, and Bianca Wright. 2020. Sarcasm detection using machine learning algorithms in Twitter: A systematic review. International Journal of Market Research, 62(5), 578-598.

Silviu Oprea and Walid Magdy, 2019. isarcasm: A dataset of intended sarcasm. arXiv preprint arXiv:1911.03123.

Silviu Oprea and Walid Magdy. 2020. The effect of sociocultural variables on sarcasm communication online. Proceedings of The 23rd ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW).

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. ˇ Sarcasm detection on czech and english twitter. In COLING, pages 213–223. ACL.

Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In Proceedings of the eighth ACM international conference on web search and data mining WSDM, pages 97–106. ACM.

Byron C. Wallace, Choe, Do Kook Choe, and Eugene Charniak. 2015. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1035–1044. ACL.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcas as contrast between a positive sentiment and negative situation. In EMNLP, pages 704–714. ACL.

Sara Rosenthal, Preslav Nakov, Alan Ritter, and Veselin Stoyanova. 2014. Semeval-2014 task 9: Sentiment analysis in twitter. In Proceedings of the International Workshop on Semantic Evaluation (SemEval-2014), Dublin, Ireland, August.

Yan-yan Song and Ying Lu. 2015. Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 27(2), 130.

Tin Kam Ho. 1995. Random decision forests. Proceedings of 3rd international conference on document analysis and recognition. Vol. 1. IEEE.

Palak Verma, Neha Shukla, and A. P. Shukla. 2021. Techniques of Sarcasm Detection: A Review.

In 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE) (pp. 968-972). IEEE.

SemEval-2022 Task 7: Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts

Michael Roth **Talita Anthonio** **Anna Sauer**
University of Stuttgart
Institute for Natural Language Processing
{michael.roth,talita.anthonio,anna.sauer}@ims.uni-stuttgart.de

Abstract

We describe SemEval-2022 Task 7, a shared task on rating the plausibility of clarifications in English-language instructional texts. The dataset for this task consists of manually clarified how-to guides for which we generated alternative clarifications and collected human plausibility judgements.[1] The task of participating systems was to automatically determine the plausibility of a clarification in the respective context. In total, 21 participants took part in this task, with the best system achieving an accuracy of 68.9%. This report summarizes the results and findings from 8 teams and their system descriptions. Finally, we show in an additional evaluation that predictions by the top participating team make it possible to identify contexts with multiple plausible clarifications with an accuracy of 75.2%.

1 Introduction

Understanding texts in natural language requires that both explicit text components as well as implicit references and relationships are interpreted correctly. This applies in particular to instructional texts, which demand a clear understanding of individual instruction steps in order to reach the desired goal. Possible uncertainties should therefore already be clarified in the text. In principle, such clarifications can also be generated automatically. In that case, however, it will be necessary to investigate the circumstances under which a clarification is plausible and unambiguous.

As a first step towards such an investigation, this shared task evaluates the ability of NLP systems to distinguish between plausible and implausible clarifications of an instruction. Inspired by the success of previous cloze-based evaluations (see Section 2), we set up our task as a cloze task, in which clarifications are presented as fillers and systems have to identify which fillers plausibly fit in a given context (see Table 1). Our focus in this task is on different types of referring expressions that are either underspecified or not realized explicitly at all, and we consider possible clarifications in the form of additional specification or explicitation.

Choose a Hair Salon			
(1)	Check ratings of different salons.		
(2)	Visit the salon's website.		
(3)	Call __$\emptyset$__ and ask questions.		
✓	the salon	✗	a friend
✓	the number	✗	your stylist
✓	the owner		

Table 1: Simplified example from a pilot study: the top shows a sentence (3) and shortened version of its discourse context (1–2). In the clarified version of this sentence, the phrase *the salon* was inserted. Other phrases shown in the bottom part are automatically generated fillers, annotated as plausible (✓) or implausible (✗).

Research in linguistics and psychology has shown that individuals use language differently (Pennebaker and King, 1999; Heylighen and Dewaele, 2002). In particular when it comes to implicit and underspecified language, individual differences can also lead to different interpretations (Scholman and Demberg, 2017; Poesio et al., 2019). As a result, worst case scenarios include medical instructions being followed incorrectly or news being passed on inaccurately. In view of the fact that language is inherently ambiguous, however, it is neither sensible nor expedient to produce clarifications for all occurrences of underspecification. Avoiding worst-case scenarios therefore goes beyond ranking individual clarifications by plausibility and must take into account whether multiple (incompatible) clarifications are perceived as plausible, thus reflecting possible misunderstandings.

We discuss our task and data in more detail in

[1]The task data is available at https://github.com/acidAnn/claire.

Sections 3 and 4, respectively. The participating teams are summarized in Section 5 and their results on the task and additional evaluations in Section 6.

2 Related Work

Cloze tasks have become a standard framework for evaluating various discourse-level phenomena in NLP. Some prominent examples include the narrative cloze test (Chambers and Jurafsky, 2008), the story cloze test (Mostafazadeh et al., 2016), and the LAMBADA word prediction task (Paperno et al., 2016). In these tasks, NLP systems are required to make a prediction about the filler of a cloze that is most likely to continue the discourse. However, it is not always clear whether exactly one likely filler exists. Evaluations typically circumvent this issue by requiring systems only to distinguish between a correct and an incorrect filler, or by evaluating predictions only with a relative measure. Both of these options ignore the more general challenge that multiple fillers can be plausible. Our shared task addresses this challenge explicitly, by requiring systems to classify different clarifying fillers as either plausible or implausible. This is a natural extension of previous cloze tasks to discourse contexts in which multiple interpretations are plausible. This extension makes it possible to evaluate in how far NLP systems can reflect cases of underspecification and uncertainty as well as possible sources of misunderstanding.

Our task is based on manual text revisions that can be traced through revision histories and by which possible needs for clarification can be identified. Thus, the task follows a number of existing research contributions that deal with text revisions. A number of previous works examine reasons for and types of revisions in, for example, Wikipedia (Bronner and Monz, 2012; Daxenberger and Gurevych, 2012; Yang et al., 2017; Faruqui et al., 2018) and the essay-based corpus ArgRewrite (Zhang and Litman, 2015, 2016; Afrin and Litman, 2019; Kashefi et al., 2022). In this work, we use revision histories of instructional texts because clarifications seem particularly relevant in this domain.

The starting point of our task is the data set wikiHowToImprove (Anthonio et al., 2020), which comprises revision histories of more than 250,000 how-to guides from the online platform wikiHow.[2] In our own previous work, we investigated the extent to which these histories are useful for examining textual improvements (Anthonio and Roth, 2020), predicting revision requirements (Bhat et al., 2020), modeling cases of lexical vagueness (Debnath and Roth, 2021), and resolving implicit references (Anthonio and Roth, 2021). The last two studies in particular have shown that wikiHowToImprove is well suited as a resource for studying clarifications of semantic phenomena. We describe one of these studies and how we build upon it in more detail in Section 3.

[2]www.wikihow.org

3 Task and Background

The general idea of the present shared task is to use revisions of English-language instructional texts as a basis to identify potential clarifications and to rate them regarding their plausibility. We assume that at least certain cases of clarifying revisions follow patterns that can be recognized automatically, by comparing the text before/after revision. An example of such a pattern is the insertion of a nominal phrase mentioned in context that makes an implicit reference explicit (see Table 1). We consider additional patterns as part of this shared task (see Section 4), but only consider cases of insertion for simplicity.

The focus on insertions allows us to consider clarifications as solutions to a cloze test, since the revision always fills in a text segment that previously was not present. Compared to previous cloze tasks, we do not assume that the revision observed is always unique and plausible. Instead, we also consider alternative clarification options and obtain plausibility judgments for all options.

As background, we first summarize findings from a pilot study that we conducted before setting up the present task (§3.1). Based on this, we then describe the settings of the shared task (§3.2).

3.1 Pilot Study

In our pilot study (Anthonio and Roth, 2021), we constructed a dataset of implicit references and potential clarifications in three steps: (1) heuristically identifying insertions of nominal phrases mentioned in the previous context, (2) automatically generating alternative clarifications using generative language modeling (GPT; Radford et al., 2018), and (3) collecting human plausibility judgements for each clarification option.

The first step of our pilot showed that it is possible to extract about 6,000 relevant clarifications from the revisions in wikiHowToImprove. We further found that most noisy instances can be filtered by the application of linguistic constraints and that remaining cases can be identified during the manual verification in the final step. In the second step, we found GPT to produce completions for many sentences that seem sensible on the surface level. Using our best strategy, namely re-ranking based on paragraph-level perplexity, the best sequence generated by the model was identical to the human-inserted clarification in over 56% of cases, and the clarification appeared in the top-10 generated sequences in 78% of cases.

A crucial finding in the third step was that the annotator indicated a preference for the human-inserted clarification in most cases (68%), but different, model-generated clarifications were judged as equally good in many cases (24%). In some cases, the annotator actually preferred a generated clarification over the human-edited insertion (8%).

The framework for the shared task is strongly motivated by the finding that alternative clarifications, generated by a computational model, can be as good or even better than human-produced clarifications. In some cases, we simply found different verbalizations of the same proposition. In other cases, like examples (a) and (b) below, we found plausible alternatives that are not fully compatible semantically.

(a) Call <u>the salon</u> and ask questions.

(b) Call <u>each salon</u> and ask questions.

When multiple incompatible readings exist, there is a risk that instructions will be misunderstood and not lead to the desired goal. To identify potential occurrences of such cases, we consider different fillers in the shared task and rate the plausibility of each filler independently.

3.2 Shared Task Settings

The SemEval shared task is set up as follows: Systems are provided with a cloze sentence, surrounding sentences and a potential clarifying filler as input, and are required to make a prediction regarding the plausibility of the filler in the given context. For evaluation, predicted labels are compared against the manually collected plausibility judgements described in Section 4. We define two subtasks with different labels and evaluation measures.

Task 1: Classification. In the classification task, systems need to distinguish between three labels (IMPLAUSIBLE, NEUTRAL and PLAUSIBLE). We use *accuracy* as the main evaluation measure, calculated as the proportion of correct predictions among all predictions of a system.

Task 2: Ranking. In the ranking task, systems need to predict a continuous plausibility score. We evaluate the predictions based on their *correlation* with human judgements, calculated as Spearman's rank correlation coefficient between all predictions and all judgements.

We describe the selection of data and collection of human judgements in the next section. In Section 6, we discuss additional evaluations performed to assess system performance with regard to the presence of multiple plausible clarifications.

4 Data

We closely follow the three steps of our pilot study, described in Section 3.1, to construct the data for this shared task. Our starting point is the dataset wikiHowToImprove (Anthonio et al., 2020), a resource of sentence-level revisions and their contexts based on wikiHow. In the first two steps, we create relevant data from this resource automatically; in the final step, we collect manual annotations to form a gold standard. In step 1, we apply a pattern-based approach to identify revisions that involve insertions that serve specific clarifying functions (§4.1). In step 2, we use transformer-based language models to produce sets of alternate clarifications that may or may not be compatible with an observed insertion (§4.2). In step 3, we collect human plausibility judgements on each clarification independently (§4.3).

4.1 Data Extraction

We collect relevant revisions by identifying cases in which a single *contiguous* insertion and no other change was made within a sentence. We compute differences and extract cases automatically based on the Python library `difflib`[3] and the following preprocessing tools: `spaCy`[4] for sentence splitting and tokenization, the Berkeley Neural Parser (Kitaev and Klein, 2018) for constituency parsing and `Stanza` (Qi et al., 2020) for POS tagging, dependency parsing and co-reference resolution. For the shared task, we focus on four types of phenomena, which are summarized in Table 2.

[3]https://docs.python.org/3/library/difflib.html

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

Phenomenon	Clarification pattern	Example	Potential filler
Implicit Reference	$\emptyset \rightarrow$ [DET] NOUN	Lift your toes up while keeping your leg straight. Hold __ $\emptyset$ __ for a few seconds, then release. Incorporate calf stretches into your yoga routine.	✓ your pose ✓ the stretch ? a leg ? the chair ✗ your head
Fused head	DET/JJ $\emptyset \rightarrow$ DET/JJ NOUN	Traditionally, the groom waits for the bride at the altar, the bride tosses the bouquet and (...). Since this is your wedding, feel free to change these __ $\emptyset$ __.	✓ ideas ✓ plans ? symbols ? characters ✗ changes
Noun compound	$\emptyset$ NOUN $\rightarrow$ NOUN NOUN	Heating for cold water tanks isn't quite of an issue as for tropicals. In fact, you can keep a __ $\emptyset$ __ tank without a heater.	✓ goldfish ✓ freshwater ✓ water ? fishing ✗ soup
Metonymy	NP $\emptyset \rightarrow$ NP's NP $\emptyset$ NP $\rightarrow$ NP of NP	Look at the __ $\emptyset$ __ of the teeth. If you're unsure of your dog's age, or want to determine if they are already entering into the senior territory, try the teeth.	✓ condition ✓ color ✓ thickness ? layout ? points

Table 2: Phenomena, extraction patterns and example clarifications (✓ plausible, ? neutral, ✗ implausible).

Implicit references. Instances with a non-verbalized reference in the original sentence which was clarified in the revised sentence through insertion. We select the cases from Anthonio and Roth (2021) with insertions containing a single noun or a determiner followed by a noun.

Fused heads. Instances of noun phrases for which the head noun was implicit in the original sentence and clarified in the revised sentence through insertion. We search for noun phrases with a determiner or adjective head in the original sentence and select those instances where a single noun was inserted in the revision.

Noun compounds. Instances of underspecified noun phrases, which were clarified in the revised sentence through the insertion of a dependent noun to form a more specific compound. We select instances of single noun insertions in which the inserted noun is a `compound` dependent of another noun that has already been present in the original sentence.

Metonymy. Instances in which a revision adds a noun y to a noun x to make explicit to which component or aspect of x the text refers. For the genitive pattern x*'(s)* y, we select insertions including an apostrophe and a noun y that is in a dependency relation `nmod:poss` with a noun x. For the y *of* x pattern, we select insertions that consist of a noun y and the token *of* added right in front of a noun x, allowing for intervening determiners and adjectives.

4.2 Constructing Clarifications

We produce a set of possible clarifications for each instance as follows: First, we generate the top-100 fillers in place of an observed insertion using language modeling. Second, we select a subset of potentially suitable clarifications by filtering and clustering the top-100.

Filler generation. For the implicit references, we take the top-100 generated clarifications from Anthonio and Roth (2021). For the other phenomena, we generate alternative clarifications automatically using the same approach as Anthonio and Roth (2021). That is, we feed the original sentence s with the surrounding sentences from the same paragraph to a language model. We then compute the top-100 completions for the token position(s) where an insertion was added in the revised sentence. We use BERT (Devlin et al., 2019) instead of GPT (Radford et al., 2018) to generate the clarifications, as the required insertions consist of only one token and BERT makes it possible to also consider follow-up context directly. The BERT checkpoint `bert-base-uncasedbert-base-uncased`[5] in `Transformers` (Wolf et al., 2020) was used without additional pre-training.

Filler selection. From the top-100 clarifications provided by the language model, we select four fillers with the goal of producing a semantically diverse set of clarifications. First, we remove unsuitable fillers from the top-100, including cases that only consist of digits or non-alphanumerical characters and fillers that do not have the right part of speech based on `Stanza` (retaining only *NOUN* for fused heads and metonymy and *NN* for noun compounds to exclude plural nouns).

For all instances with ≥ 4 candidate fillers, we select the observed insertion from the revised sentence as one filler. To select semantically different fillers as alternate candidates, we apply k-means clustering with $k = 4$ to the remaining candidates, using the algorithm by Elkan (2003) as implemented in `sklearn` (Pedregosa et al., 2011). We obtain vector representations for clustering from BERT (`bert-base-uncased`) by averaging over the last hidden state for all tokens in a filler. After clustering, we select the fillers closest to the four cluster centroids based on cosine similarity.

4.3 Plausibility Annotation

Task. After selecting fillers for each sentence, we collect plausibility judgements on Amazon Mechanical Turk for our train set (19,975 instances, i.e. 3995 sentences with 1 human and 4 generated fillers each[6]), development and test sets (2,500 instances each, i.e., 125 sentences per phenomenon with 5 fillers per sentence). Each clarification in the training set is annotated by 2 crowdworkers. For the development and test set, we collected annotations from 4 crowdworkers to ensure a consistently high quality. In each annotation task, we ask participants to indicate on a scale from 1 to 5 whether the clarification made sense in the given how-to-guide. A screenshot of the interface for our Human Intelligence Task (HIT) is provided in Appendix A.

[5]We also tried `bert-base-cased` in preliminary experiments but observed no improvements.

[6]1000 each for noun compounds and metonymy, 996 for implicit references and 999 for fused heads.

	Train	Dev	Test
IMPLAUSIBLE	5,474 (27%)	982 (39%)	858 (34%)
NEUTRAL	7,162 (36%)	602 (24%)	672 (27%)
PLAUSIBLE	7,339 (37%)	916 (37%)	970 (39%)
Total	19,975	2,500	2,500

Table 3: Distribution of class labels in our training, development and test sets.

Qualifications. We use several qualifications to increase the annotation quality. First, we require participants to be located in the United States or in the United Kingdom, to increase the chance that the participants are native speakers of English. Secondly, participants need to have a HIT approval rate $\geq 95\%$ and their number of approved HITS has to be ≥ 1000. Finally, annotators are required to pass a qualification test in which they are asked to judge a list of clearly plausible and implausible cases that were pre-selected unanimously by the authors.

Class labels. For Task 1 (classification), we average over the real-valued judgements collected for a clarification and map this plausibility score to one of the three classes labels. Specifically, we label clarifications with an average score ≤ 2.5 as IMPLAUSIBLE, clarifications with a score ≥ 4.0 as PLAUSIBLE, and all clarifications between these thresholds as NEUTRAL. The thresholds have been selected based on manual inspection of the data and mathematical considerations: in particular, the threshold for PLAUSIBLE requires scores to be substantially above average (in case of two judgements, $\geq$3&5 or $\geq$4&4), whereas the IMPLAUSIBLE threshold allows for a slightly wider range of judgements. The NEUTRAL label covers cases that received inconclusive individual scores as well as cases of disagreement (e.g. 3&3 as well as 2&5).

Statistics. We show the frequency distribution of the labels in the train, development and test set in Table 3. It is noteworthy that development and test set proportionally includes fewer NEUTRAL and more IMPLAUSIBLE clarifications than the training set. Presumably, this is because we increased the number of qualification questions from 4 to 6 after collecting the training data to ensure the quality of the evaluation data.

Since we are particularly interested in cases with

Team	Model type	Pre-trained model components	Additional comments
X-PuDu	ensemble	DeBERTa, ERNIE, XLM-R	pattern-aware, multi-loss
HW-TSC	ensemble	DeBERTa, RoBERTa, S-BERT	incl. unsupervised model
PALI	ensemble	DeBERTa, RoBERTa, XLM-R	pattern-aware, multi-loss
Nowruz	Transformer	T5	ordinal regression, multi-loss
JBNU-CCLab	ensemble	DeBERTa	–
DuluthNLP	Transformer	ELECTRA	class weighting
Stanford MLab	Transformer	ELECTRA	–
niksss	Transformer	BERT	–

Table 4: Summary of the best models on the test set according to the submitted system descriptions.

multiple plausible clarifications, we also compute the average number of PLAUSIBLE clarifications per sentence s, which we found to be 1.84, 1.87 and 1.84 in the training, development and test set, respectively. This means that, on average, each annotated sentence in the dataset has between 1 and 2 clarifications that the annotators rated as plausible.

5 Participants

A total of 21 users participated in the CodaLab competition set up for the shared task and 8 teams submitted system description papers. An overview of the best model by each team is shown in Table 4.[7] We observe that all systems are based on Transformer architectures, using one or more of the following pre-trained models: BERT (Devlin et al., 2019), DeBERTa (He et al., 2020), ELECTRA (Clark et al., 2019), ERNIE (Sun et al., 2019), RoBERTa (Liu et al., 2019), S-BERT (Reimers and Gurevych, 2019), T5 (Raffel et al., 2020), XLM-R (Conneau et al., 2020).

In addition to fine-tuning a single or multiple Transformer models in an ensemble, some teams have taken additional steps to adapt their system to the task. We summarize some of these steps below.

Consideration of phenomena. At least two teams took into account that the data set consists of four phenomena that were identified using different patterns (*pattern-aware*): PALI used the phenomenon description that applies to a classification instance as additional model input; X-PuDu developed an ensemble architecture that consists of different individual models and hyperparamters for each phenomenon.

[7]A table with the official results of the CodaLab competition, including participants who did not submit system descriptions, is shown in Appendix B.

Adapted loss functions. Several teams adapted the loss functions of their models to better account for various properties of the task. This includes the use of classification and regression based loss functions in a multi-task learning set-up (*multi-loss*) as well as the use of specific loss functions that consider the ordinal nature of labels (*ordinal regression*) or differences in label distributions (*class weighting*) in the classification task.

Unsupervised components. Given the similarity of our task to general cloze tasks, several teams experimented with models that were merely self-supervised and not fine-tuned on task-specific training data. In case of one team, HW-TSC, such an unsupervised component is also part of the ensemble model that produced the best results.

6 Results and Discussion

The results for Task 1 and 2 are shown in Table 5 and 6, respectively. We focus our discussion on Task 1: Classification, as the participants of Task 2 form only a subset of the Task 1 participants and the system results rank, with exception of the last two teams, in the same order. In addition to showing results by participants, we also provide a human upper bound as well as results by our own BERT-based baseline model. The upper bound was computed as the accuracy over all individual annotations when compared against the (averaged) class label of each test instance.

The human upper bound has an accuracy of 79.4%, indicating that the task is challenging and potentially involves a number of disagreements. The winning team of the competition, X-PuDu, achieves an accuracy of 68.9%, only 10.5 percentage points below the human upper bound. The results of all teams lies substantially above a naive majority class baseline of 39%. All teams but one

Rank	Team	Accuracy
–	Human (upper bound)	79.4%
1	X-PuDu	68.9%
2	HW-TSC	66.1%
3	PALI	65.4%
4	Nowruz	62.4%
5	JBNU-CCLab	61.4%
6	DuluthNLP	53.3%
7	Stanford MLab	46.6%
8	niksss	44.2%
–	BERT (baseline)	45.7%

Table 5: Results for Task 1 (classification).

Rank	Team	Spearman's ρ
1	X-PuDu	0.807
2	PALI	0.785
3	HW-TSC	0.774
4	Nowruz	0.707
5	niksss	0.252
6	Stanford MLab	0.194

Table 6: Results for Task 2 (ranking).

Rank	Team	F1 (all)	F1 (w/o N)
1	X-PuDu	0.689	0.773
2	HW-TSC	0.661	0.749
3	PALI	0.654	0.749
4	Nowruz	0.624	0.714
5	JBNU-CCLab	0.551	0.627
6	DuluthNLP	0.533	0.608
7	Stanford MLab	0.466	0.514
8	niksss	0.442	0.494

Table 7: Classification results with/without NEUTRAL.

Rank	Team	Accuracy (#P$\geq$2)
1	X-PuDu	75.2%
2	HW-TSC	73.2%
3	PALI	72.6%
4	Nowruz	71.6%
5	JBNU-CCLab	63.6%
6	DuluthNLP	62.6%
7	Stanford MLab	54.8%
8	niksss	60.0%

Table 8: Results for identifying contexts with multiple plausible fillers, based on individual model predictions.

also outperform our BERT-based baseline, which is a linear classification model based on the checkpoint provided by the Transformer library (Wolf et al., 2020) and fine-tuned on our training data.

6.1 Findings by Participants

In the following, we briefly summarize a couple of findings by task participants. More details can be found in the individual task description papers.

Different phenomena. The winning team, X-PuDu, found that different hyperparameters worked best depending on the phenomenon/extraction pattern. Based on this finding, different individual models were trained and combined in an ensemble.

Label distribution. Some teams, including DuluthNLP, noticed performance issues related to the distribution of labels in the development data. As a dedicated solution, DuluthNLP uses a decreased weight for the NEUTRAL label in the loss function.

NEUTRAL label. Team JBNU-CCLab reported that the NEUTRAL label is generally difficult to distinguish from other labels by different models. An underlying problem could be that the label represents instances that are seen as somewhat plausible by multiple annotators as well as instances that are seen as plausible by some annotators and implausible by others (see Section 4).

Noisy data. Team HW-TSC found that isolated training instances have the label NEUTRAL rather than PLAUSIBLE, even though the respective filler represents a human insertion (i.e., the filler can be found in the final version of the text in wikiHow). As the results of our human upper bound in Table 5 show, this is partly because the right label is sometimes not clear cut even for humans. We discuss this aspect in more detail in the next section.

6.2 Additional Evaluations

We perform two additional evaluations to assess the impact of the NEUTRAL label on system performance and to investigate the possibility of identifying whether multiple plausible clarifications exist by aggregating the predictions regarding individual clarifications.

Excluding NEUTRAL. For the evaluation without the NEUTRAL label, we calculate micro-averaged precision, recall and F_1-scores for the two labels PLAUSIBLE and IMPLAUSIBLE. The results in

Correct	#P≥2	Text	Fillers
8 (all)	✓	Galette des rois—or "King Cake" in English—is traditionally made to celebrate the __∅__ of Epiphany. Especially popular in France during the Christmas season, it is enjoyed elsewhere too.	✓ holidays ✓ Feast ? hours ? celebration ? proclamation
	✗	Let the __∅__ of shoes air dry. You can put them in front of a dehumidifier, a fan, or an open window, but avoid putting them in front of any type of heat source.	✓ pair ? pile ✗ shoes ✗ color ✗ end
0 (none)	✓	If you want a smoother surface, try a __∅__ of paper with a higher amount of grains, if you want a faster job but a rougher surface try a paper with a lower amount of grains.	✓ thickness ✓ piece ? fabric ? product ✗ pile
	✗	Your cucumber plant will also grow thin, light green shoots that help the plant grasp onto a surface and grow vertically. These __∅__ grow immediately next to the suckers.	✓ shoots ? fibers ? tendrils ? foliage ✗ bushes

Table 9: Examples of difficult and easy instances, selected based on how many systems classified them correctly.

terms of F_1-score are shown in Table 7. The results indicate that all systems perform substantially better in the evaluation setting that ignores NEUTRAL labels. The ranking is identical to the ranking in the evaluation including all labels. Considering only the PLAUSIBLE and IMPLAUSIBLE, Team X-PuDu achieves the highest micro-averaged F_1-score of 0.773. In the cases where their system predicts a non-NEUTRAL label, it is correct in 72.7% of cases (precision), and 82.5% of all non-NEUTRAL instances in the data received the correct prediction (recall).

Multiple clarifications. In our final evaluation, we examine whether system predictions can also be used to determine whether multiple plausible clarifications for a given context exist. For this, we consider the labels of each individual clarification and compare system outputs and annotations in terms of whether two or more clarifications for a cloze and its context received the label PLAUSIBLE. We show the result of this evaluation in terms of accuracy for each team in Table 8. Apart from the last two places, the teams rank in the same order as in the other evaluations. The best performing team, X-PuDu, correctly predicts whether two or more plausible clarifications exist for 75.2% of all cases. Table 9 shows examples that were correctly classified by all or none of the systems.

7 Conclusion

In this paper, we presented the task, data, participating systems, and results of the shared task on clarifying implicit and underspecified phrases in instructional texts. Our motivation for this task was to explore the possibility of testing different clarifications for plausibility. In particular, we were concerned with the question of whether two or more clarifications can be plausible and whether such cases can be detected automatically. To create a suitable dataset, we worked with and identified a set of revisions with manual clarifications, automatically generated possible alternatives, and then collected human plausibility ratings.

In total, 21 users participated in our shared task. We summarized the systems and results of 8 teams that submitted descriptions of their systems. The best systems from each group have in common that they are based on Transformer architectures or combine them in an ensemble. The best system achieved 68.9% accuracy, only 10.5 percentage points below a human upper bound. In additional evaluations, we have shown that an accuracy of up to 75.2% is achieved with respect to the detection of multiple plausible clarifications.

The results show that the presented task is a difficult one, but that many cases can already be modeled well by current state-of-the-art methods. There is further room for improvement with respect to both the data set and models: with respect to the data, it should be noted that the training set with less than 20k instances is relatively small and that there are many instances with a underspecified NEUTRAL label (36%). On the model side, we found that the participating teams make complementary contributions that may allow for additional improvements in combination.

One shortcoming of the task as presented and performed is that we only considered four forms of clarifications related to referring expressions. In addition, clarifications were assessed individually and judgements by different annotators were aggregated. In the long term, we believe that more forms of clarifications as well as individual differences regarding their plausibility need to be considered. Finally, future work will have to investigate under which circumstances multiple different clarifications are actually incompatible and can thus reveal potential sources of misunderstanding.

Acknowledgements

The research presented in this paper was funded by the DFG Emmy Noether program (RO 4848/2-1).

References

Tazin Afrin and Diane Litman. 2019. Identifying editor roles in argumentative writing from student revision histories. In *International Conference on Artificial Intelligence in Education*, pages 9–13. Springer.

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020. wikiHowToImprove: A resource and analyses on edits in instructional texts. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5721–5729, Marseille, France. European Language Resources Association.

Talita Anthonio and Michael Roth. 2020. What can we learn from noun substitutions in revision histories? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1359–1370, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Talita Anthonio and Michael Roth. 2021. Resolving implicit references in instructional texts. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 58–71, Punta Cana, Dominican Republic and Online. Association for Computational Linguistics.

Irshad Bhat, Talita Anthonio, and Michael Roth. 2020. Towards modeling revision requirements in wikiHow instructions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8407–8414, Online. Association for Computational Linguistics.

Amit Bronner and Christof Monz. 2012. User edits classification using document revision histories. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 356–366, Avignon, France. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Johannes Daxenberger and Iryna Gurevych. 2012. A corpus-based study of edit categories in featured and non-featured Wikipedia articles. In *Proceedings of COLING 2012*, pages 711–726, Mumbai, India. The COLING 2012 Organizing Committee.

Alok Debnath and Michael Roth. 2021. A computational analysis of vagueness in revisions of instructional texts. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 30–35, Online. Association for Computational Linguistics.

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

Charles Elkan. 2003. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th international conference on Machine Learning (ICML-03)*, pages 147–153.

Manaal Faruqui, Ellie Pavlick, Ian Tenney, and Dipanjan Das. 2018. WikiAtomicEdits: A multilingual corpus of Wikipedia edits for modeling language and discourse. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 305–315, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Francis Heylighen and Jean-Marc Dewaele. 2002. Variation in the contextuality of language: An empirical measure. *Foundations of science*, 7(3):293–340.

Omid Kashefi, Tazin Afrin, Meghan Dale, Christopher Olshefski, Amanda Godley, Diane Litman, and Rebecca Hwa. 2022. Argrewrite v. 2: an annotated argumentative revisions corpus. *Language Resources and Evaluation*, pages 1–35.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

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

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

James W. Pennebaker and Laura A. King. 1999. Linguistic styles: language use as an individual difference. *Journal of personality and social psychology*, 77(6):1296–1312.

Massimo Poesio, Jon Chamberlain, Silviu Paun, Juntao Yu, Alexandra Uma, and Udo Kruschwitz. 2019. A crowdsourced corpus of multiple judgments and disagreement on anaphoric interpretation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1778–1789, Minneapolis, Minnesota. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. OpenAI.

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

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

Merel Scholman and Vera Demberg. 2017. Crowdsourcing discourse interpretations: On the influence of context and the reliability of a connective insertion task. In *Proceedings of the 11th Linguistic Annotation Workshop*, pages 24–33, Valencia, Spain. Association for Computational Linguistics.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

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

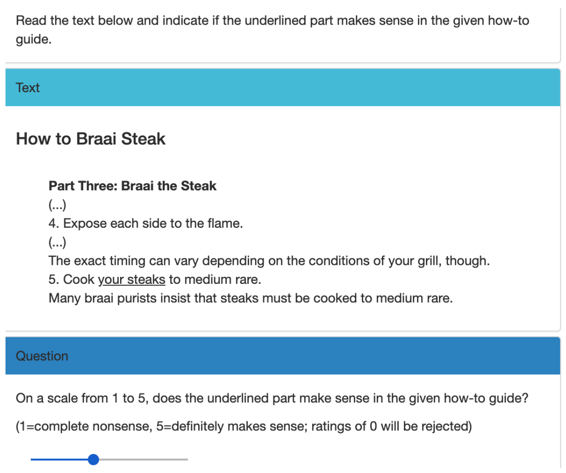
Diyi Yang, Aaron Halfaker, Robert Kraut, and Eduard Hovy. 2017. Identifying semantic edit intentions from revisions in Wikipedia. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2000–2010, Copenhagen, Denmark. Association for Computational Linguistics.

Fan Zhang and Diane Litman. 2015. Annotation and classification of argumentative writing revisions. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 133–143, Denver, Colorado. Association for Computational Linguistics.

Fan Zhang and Diane Litman. 2016. Using context to predict the purpose of argumentative writing revisions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1424–1430, San Diego, California. Association for Computational Linguistics.

A Annotation Interface

The annotation interface for our crowdsourcing task is depicted in Figure 1. Annotators see and have to rate a single underlined clarification in its context.


Read the text below and indicate if the underlined part makes sense in the given how-to guide.

Text

How to Braai Steak

Part Three: Braai the Steak
(...)
4. Expose each side to the flame.
(...)
The exact timing can vary depending on the conditions of your grill, though.
5. Cook your steaks to medium rare.
Many braai purists insist that steaks must be cooked to medium rare.

Question

On a scale from 1 to 5, does the underlined part make sense in the given how-to guide?
(1=complete nonsense, 5=definitely makes sense; ratings of 0 will be rejected)

Figure 1: Interface for collecting annotations.

B CodaLab Leaderboard

In the main part of the paper, we only list results of participants who provided a description of their system(s) for the shared task. Table 10 shows a complete set of user names and results of the participants in the CodaLab competition, including users who did not submit a system description.

Team name	User name	acc.	ρ
X-PuDu	tt123	0.689	0.807
HW-TSC	Yinglu_Li	0.661	0.774
–	tiantaijian	0.661	0.763
–	fanxiaoxing	0.656	–
PALI	stce	0.654	0.785
–	hudou	0.641	–
–	huangwkk	0.631	0.774
Nowruz	mohammadmahdinoori	0.624	0.707
JBNU-CCLab	OrangeAvocado	0.614	–
–	CitizenTano	0.595	–
–	huawei_zhangmin	0.589	0.640
–	parkwonjae	0.554	–
–	lith	0.537	0.600
–	ywzhang_cr	0.537	0.600
DuluthNLP	Sakrah	0.533	–
Stanford MLab	patrickliu2011	0.466	0.194
–	Autism_PAFC	0.461	–
–	SelinaIW	0.456	–
niksss	niksss	0.442	0.252
–	andrei.manea	0.418	-0.109
–	tanigaki	0.395	0.415

Table 10: Oveview of results, including user submissions without a shared task system description.

JBNU-CCLab at SemEval-2022 Task 7: DeBERTa for Identifying Plausible Clarifications in Instructional Texts

Daewook Kang, Sung-Min Lee, Eunhwan Park, Seung-Hoon Na
Computer Science and Engineering, Jeonbuk National University, South Korea
{dwkng, cap1232, judepark, nash}@jbnu.ac.kr

Abstract

In this study, we examine the ability of *contextualized representations* of pretrained language model to distinguish whether sequences from instructional articles are plausible or implausible. Towards this end, we compare the BERT, RoBERTa, and DeBERTa models using simple classifiers based on the sentence representations of the [CLS] tokens and perform a detailed analysis by visualizing the representations of the [CLS] tokens of the models. In the experimental results of Subtask A: *Multi-Class Classification*, DeBERTa exhibits the best performance and produces a more distinguishable representation across different labels. Submitting an ensemble of 10 DeBERTa-based models, our final system achieves an accuracy of 61.4% and is ranked fifth out of models submitted by eight teams. Further in-depth results suggest that the abilities of pretrained language models for the plausibility detection task are more strongly affected by their model structures or attention designs than by their model sizes.

1 Introduction

WikiHow[1] is the largest how-to website with more than 300,000 articles and over 2.5M registered users that help user improve their knowledge of specific areas. However, these instructional articles have grammatical errors and ambiguous content that cause misunderstandings. To enhance the clarity of instructional texts, *clarification* is required as a revision that makes implicit elements explicit, resolves ambiguities, or replaces underspecified phrases with a clearer and more precise expressions.

SemEval-2022 Task 7 (Roth et al., 2022) evaluates the ability of an NLP system to distinguish between plausible and implausible clarifications of an instruction. The task is formulated as a CLOZE task in which clarification is presented as a *filler* in a blanked sentence. Given a context with a filler option (i.e., a sentence X and filler option O), the system should determine the plausibility of a sequence, that is whether a sequence is "plausible," "neutral," and "implausible."

Our work is motivated by the recent successes of pretrained language models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and DeBERTa (He et al., 2021a,b), which have effectively induced *contextualized representations*, achieving remarkable fine-tuning performance on downstream tasks.

To explore the plausibility detection of clarifications in SemEval-2022 Task 7, we compare the plausibility detection abilities of three pretrained language models (BERT, RoBERTa, and DeBERTa) in Subtask A, *Multi-Class Classification*. Our main observations of the development set are highlighted as follows.

- Among the three models, DeBERTa exhibits the best performance in the plausibility classification task[2].
- Visualization analysis of the representations of the [CLS] tokens of BERT, RoBERTa, and DeBERTa, reveals that the best distinguishable representations among different classes are achieved with DeBERTa.
- A comparison of base and large models confirms that large models are always better than base models for all the types: BERT, RoBERTa and DeBERTa.
- Given the comparative results among various models, we hypothesize that the abilities of pretrained language models for the plausibility detection task are more strongly affected

[1] https://www.wikihow.com/Main-Page

[2] Note that we used BERT-{base, large}, RoBERTa-{base, large}, and DeBERTa-{base, large}. DeBERTa V3 model of (He et al., 2021a) was used for DeBERTa models.

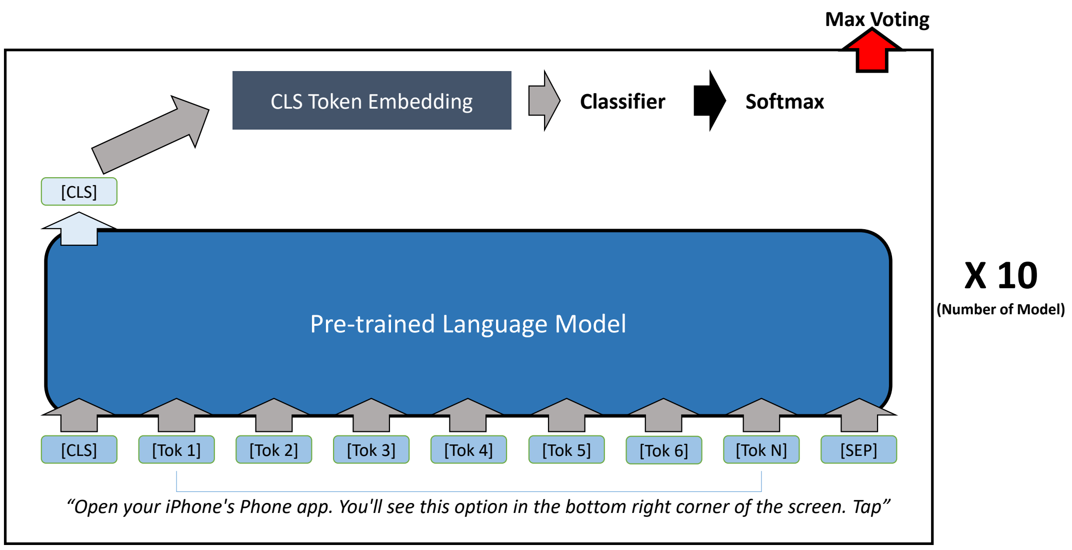


Figure 1: Architecture of the proposed system for plausibility classification using pretrained language models

by their model structures or attention designs rather than their parameter sizes.

By ensembling 10 different DeBERTa-based models, our final submitted system achieves an accuracy of 61.4% on the test data and is ranked the 8th place among 21 systems[3], that is, 5thth place among the 8 teams who submitted their papers on Subtask A.

The remainder of this paper is organized as follows: Section 2 presents related work. Section 3 describes the system architecture in detail. Section 4 describes the experimental settings, results, and analyses. Our concluding remarks and a description of future work are presented in Section 5.

2 Related work

Since the success of BERT (Devlin et al., 2019), it has been used for numerous natural language processing (NLP) tasks and has inspired the emergence of many other pretrained models. In RoBERTa (Liu et al., 2019), dynamic masking in the masked language modeling (MLM) objective dynamically while revisiting the next sentence prediction owing to its uncertain effectiveness has achieved promising results on GLUE (Wang et al., 2018), RACE (Lai et al., 2017), and SQuAD (Rajpurkar et al., 2016). DeBERTa (He et al., 2021b) further advanced BERT based on two major extensions, *disentangled attention* and *enhanced mask decoder*, by combining both the relative and absolute positions of words. DeBERTa V3 (He et al., 2021a) replaces the MLM objective in DeBERTa with the replaced token detection (RTD) objective proposed by ELECTRA (Clark et al., 2020) and further proposes *gradient-disentangled embedding sharing* to alleviate the *tug-of-war* problem between the generator and discriminator [4] as an improvement of the embedding sharing method used in ELECTRA.

3 System description

Figure 1 presents the architecture of our system, which uses pretrained language models equipped with ensemble inference.

3.1 Methods with pretrained language models

Let X^p, X^c and X^n be the previous, main, and follow-up context, respectively. As in Section 4.1, we concatenate these sentences for the i-th training example as follows: $X_i = X^p \oplus X^c \oplus X^n$, where X_c is unmasked by filling each possible filler option O_{ij}. We use $Y_i \in \{0, 1, 2\}$ to refer to the ground truth of the i-th example, where Y_0, Y_1, and Y_1 refer to *implausible*, *neutral*, and *plausible* sequences, respectively. Finally, we denote a training set as $\mathcal{D} = \{X_i, Y_i\}_{i=1}^{N}$, where N is the total number of training examples obtained by unmasking all the main sentences with their possible filler options.

We feed X_i into a pretrained language model denoted as LM to encode contextualized represen-

[3]https://competitions.codalab.org/competitions/35210

[4]Because the generator and discriminator have different objectives, they tend to pull shared word embeddings in different directions, resulting in degradation of the training speed.

tations $\mathbf{M}_i \in \mathbb{R}^{|X_i| \times d}$ as follows:

$$\mathbf{M}_i = \mathsf{LM}(X_i)$$

where $|X_i|$ is the length of the concatenated sentence X_i and d is the dimensionality of the hidden representation of LM. As mentioned, we use BERT, RoBERTa, and DeBERTa for LM.

Let $\mathbf{M}_{i,[CLS]}$ be the representation of the [CLS] token of X_i. To perform a plausibility prediction, we feed $\mathbf{M}_{i,[CLS]}$ to a linear layer as follows:

$$f(X_i) = \mathbf{W}^T \mathbf{M}_{i,[CLS]} + \mathbf{b}, \qquad (1)$$

where $\mathbf{W} \in \mathbb{R}^{d \times 3}$ and $\mathbf{b} \in \mathbb{R}^3$ are task-specific parameters of the linear layer. The loss function $\mathcal{L}$ used to optimize our system is formulated as follows:

$$\mathcal{L} = - \sum_{(X_i, Y_i) \in \mathcal{D}} \mathbf{y}_i \cdot \log \text{softmax}(f(X_i)) \qquad (2)$$

where $\mathbf{y}_i \in \{0, 1\}^3$ is the one-hot vector for Y_i.

4 Experiments

Team Name (or User Name)	**Accuracy**
X-PuDu	68.9
HW-TSC	66.1
PALI	65.4
Nowruz	62.4
DuluthNLP	53.3
Stanford MLab	46.6
niksss	44.2
JBNU-CCLAB	**61.40**

Table 1: Results of our system on the test dataset (Official Leaderboard)

4.1 Experimental setting

Dataset We used the data and labels from SemEval-2022 Task 7 (Roth et al., 2022). The data consists of the article title, sub-heading, masked sentence, previous and follow-up context, and possible filler options with corresponding labels (Subtask A) and ratings (Subtask B). During our preprocessing step, the placeholder in the main sentence is filled with each possible filler option O_i.

Preprocessing We concatenate the previous context, main context, and follow-up context without using the article name and section header; any parenthesis and the content inside, special characters, and redundant whitespaces are removed. No truncation is applied because none of the concatenated data exceeds the maximum token length of the pretrained language models.

For example, suppose that a masked sentence is given as *"You'll see this ______ in the bottom right corner of the screen."* and is filled with each filler text "*option*." Assume that the sentence that precedes it is *"1. Open your iPhone's Phone app. (...),"* and the sentence that follows it *"3. Tap ."* The concatenated input looks like *"Open your iPhone's Phone app. You'll see this option in the bottom right corner of the screen. Tap"*.

Training We fine-tune the model on the training data with a batch size of 32 and 20 of epochs. We use the AdamW optimizer (Loshchilov and Hutter, 2019) and a cosine scheduler with warm-up steps for the initial 5% of the total steps at a learning rate of $1e-5$.

We also create a DeBERTa-based ensemble model using ten models trained on different seeds to obtain the final submission result.

4.2 Official results

For each model, we select the checkpoint with the best accuracy on the validation data as the finetuned model.

As mentioned in Section 4.1, we ensemble 10 finetuned DeBERTa-v3-large models by *max-voting* their outputs to further improve our DeBERTa-based model.

Table 1 shows the official results of our DeBERTa-based ensemble model compared with the other participants' systems.

4.3 Analysis

4.3.1 Comparison of the results on the development set

Table 2 presents the results of the validation data using three different pretrained language models without an ensemble [5]

As shown in Table 2, DeBERTa-large outperforms the baseline models by a decent margin.

Furthermore, Figure 2 shows the detailed confusion matrix of the finetuned DeBERTa-large model for the development set. As shown in Figure 2, DeBERTa-large distinguishes between plausible and implausible sequences reasonably well but has difficulty identifying neutral sequences. In our ex-

[5]While the official evaluation only measures the accuracy of the system, Table 2 lists the precision, recall and f1 score for analysis in detail.

Model	Parameters	Metric			
		Accuracy	**Precision**	**Recall**	**F1-score**
BERT-base	110M	45.36	43.32	43.06	42.67
BERT-large	340M	48.00	43.06	43.32	38.89
RoBERTa-base	125M	51.48	50.20	48.22	46.32
RoBERTa-large	355M	53.12	49.39	49.50	48.28
DeBERTa-v3-base	86M	55.92	49.19	50.84	48.36
DeBERTa-v3-large	304M	**59.88**	**52.92**	**54.20**	**50.81**

Table 2: Comparative results of BERT, RoBERTa, and DeBERTa on the validation dataset. The precision, recall, and F1-score are calculated via macro-average.

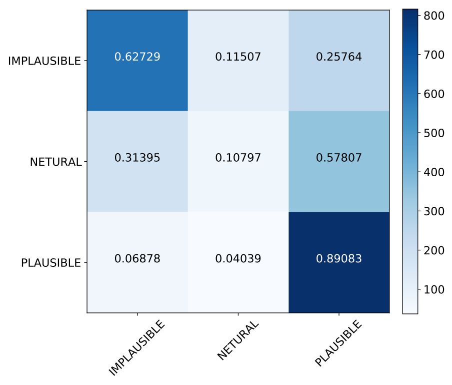


Figure 2: Illustration of the confusion matrix of our DeBERTa-based model on the validation dataset

periment, a similar tendency was also observed on BERT- and RoBERTa-based models.

Overall, from these results, we hypothesize that the ability of a pretrained language model to contextualize representations for the plausibility detection task is more strongly affected by its model structures, such as attention design or refining positional embeddings, rather than its parameter size.

4.3.2 Visualization of [CLS] representation

Figure 3 shows the visualization of the representations of [CLS] tokens using T-SNE (van der Maaten and Hinton, 2008) to compare the abilities of pretrained language models to distinguish between plausible, neutral and implausible sequences. As shown in Figure 3, the context representation distributions of the two DeBERTa models are more coherent and distinctive that those of BERT and RoBERTa. In contrast, no significant differences are observed between the BERT and RoBERTa models.

5 Conclusion

In this study, we compare BERT, RoBERTa and DeBERTa in SemEval-2022 Task 7 Subtask A: Multi-Class Classification. The results show that DeBERTa presents the best performances with improved distinguishable representations. We assume that the substantial changes made to the model structure of DeBERTa, such as disentangled attention, enhanced mask decoder, and RTD objective, would give DeBERTa a significant advantage in the addressed task.

Our final submission, based on an ensemble model comprising 10 fine-tuned DeBERTa-based models, achieved an accuracy of 61.4% on the test data. Our proposed model is ranked fifth out of eight models of teams who reported their papers.

In future studies, it would be worthwhile to explore other pretrained language models, such as ELECTRA, or models resulting from task-specific pretraining.

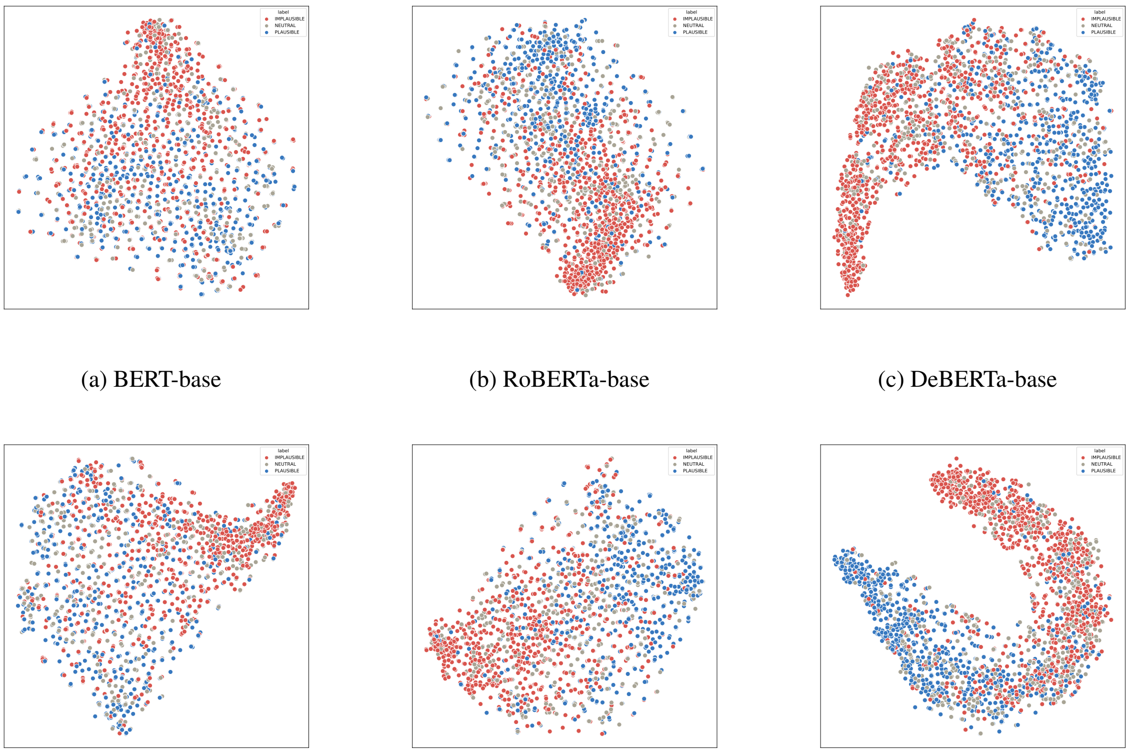
Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-02068, Artificial Intelligence Innovation Hub)

References

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Tech-*



(a) BERT-base (b) RoBERTa-base (c) DeBERTa-base

(d) BERT-large (e) RoBERTa-large (f) DeBERTa-large

Figure 3: Visualization of the representations of the [CLS] token using T-SNE among BERT, RoBERTa, DeBERTa.

nologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

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

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

A Hyperparameters

Table 3 shows the setup of hyper-parameters of our models.

epochs	20
total batch size	32
accumulation steps	4
learning rate	1e-5
optimizer	AdamW
warm-up proportion	0.05
weight decay	0.01

Table 3: Hyperparameters

HW-TSC at SemEval-2022 Task 7: Ensemble Model Based on Pretrained Models for Identifying Plausible Clarifications

Xiaosong Qiao, Yinglu Li, Min Zhang, Minghan Wang,
Hao Yang, Shimin Tao, Ying Qin
Huawei Translation Services Center, Beijing, China
{qiaoxiaosong, liyinglu, zhangmin186, wangminghan,
yanghao30, taoshimin, qinying }@huawei.com

Abstract

This paper describes the system for the identifying Plausible Clarifications of Implicit and Underspecified Phrases. This task was set up as an English cloze task, in which clarifications are presented as possible fillers and systems have to score how well each filler plausibly fits in a given context. For this shared task, we propose our own solutions, including supervised approaches, unsupervised approaches with pretrained models, and then we use these models to build an ensemble model. Finally we get the 2nd best result in the subtask1 which is a classification task, and the 3rd best result in the subtask2 which is a regression task.

1 Introduction

The rapid development of artificial intelligence has also been reflected in the field of NLP, and there have been many heavyweight achievements, such as word2Vec (Mikolov, et al., 2013), Glove (Pennington, et al., 2014), Transformer (Vaswani, et al., 2017). Natural language processing is an important branch of artificial intelligence. Cloze tasks have become a standard framework for evaluating various discourse-level phenomena in NLP, which is an important field in artificial intelligence, many researchers have long been committed to the development of this field. Some prominent examples include the narrative cloze test (Chambers and Jurafksy, 2008), the story cloze test (Mostafazadeh et al., 2016), and the LAMBADA word prediction task (Paperno et al., 2016). Cloze requires the testee to infer from the context, which is very difficult for machines.

The goal of this shared task is to evaluate the ability of NLP systems to distinguish between plausible and implausible clarifications of an instruction. Such clarifications can be critical to ensure that instructions describe clearly enough what steps must be followed to achieve a specific goal. This task was set up as a cloze task. However, different from regular cloze task, there may be zero or more than one correct candidates out of the five options. This presents new challenges for cloze systems.

For subtask 1, it is a classification task that requires the system to classify five candidates into corresponding categories, which are plausible, neutral, or implausible, and the number of each category is not fixed. This means that there may be zero or more than one correct candidates out of the five options, and the same applies to the other two categories. This situation creates new challenges for cloze tasks.

For subtask 2, it is a regression task ask annotators to rate for each clarification option whether it "makes sense in the given how-to guide" (on a scale from 1 to 5) to assess the plausibility of different clarification options.

In this paper, we analyze the characteristics of the shared task and describe out contribution to this cloze task. We build an ensemble model with Deberta-v3 (He P, et al., 2020), Roberta-large (Liu Y, et al., 2019), SBERT (Reimers and Gurevych, 2019), including supervised approaches and unsupervised approaches. Our model had the 2nd best performance in the subtask1 (66.1% Accuracy Score) and the 3rd best performance in the subtask2 (77.4% Ranking Score). The results are encouraging for evaluating various discourse-level phenomena in NLP, although there is much room for improvement.

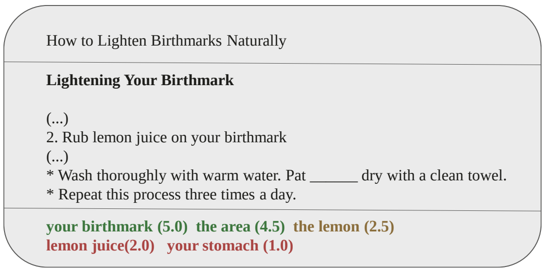
The rest of this paper is organized as follows. Section 2 introduce our approach for the shared task. Section 3 shows the experimental results of our approach and do some analysis. In Section 3, experimental results are compared and discussed. Finally, the whole paper is summarized with a brief conclusion in Section 4.

2 System Overview

In this section, we first describe our data processing steps. We experimented with different ways of processing the data, trying to find the one that worked best for the task. And then, we discuss our solutions with pre-trained models for the shared task, including unsupervised approaches, supervised approaches, and an ensemble model.

2.1 Data Processing

The participants of the shared task were provided a collection of revisions of instructional texts from the how-to website wikiHow. The dataset contains sentences that need to be filled in and its previous context, follow-up context, five options, etc. The data example is as shown in the Figure 1 below:


How to Lighten Birthmarks Naturally

Lightening Your Birthmark

(...)
2. Rub lemon juice on your birthmark
(...)
* Wash thoroughly with warm water. Pat ______ dry with a clean towel.
* Repeat this process three times a day.

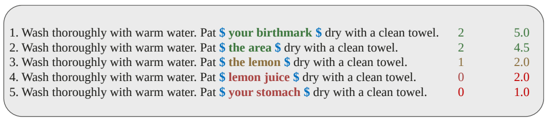
your birthmark (5.0) the area (4.5) the lemon (2.5)
lemon juice(2.0) your stomach (1.0)

Figure 1: Data example

We bring the five options into the positions that need to be filled in, and get a dataset that is five times the size of the original.

For subtask1, a label file was gave which contains the corresponding category of each option of each piece of data, which category does it belong to, plausible, neutral or implausible? We map these three categories to numbers 2, 1, 0, corresponding to plausible, neutral and implausible.

For subtask2, a score file was gave to assess the plausibility of different clarification options. For the score file, we keep it in its native state. In addition, in order to make the model focus on the positions that need to be filled in, we have added special symbols $ on both sides of the blank. After data processing, the data example is as shown in the Figure 2 below:



1. Wash thoroughly with warm water. Pat $ **your birthmark** $ dry with a clean towel.	2	5.0
2. Wash thoroughly with warm water. Pat $ **the area** $ dry with a clean towel.	2	4.5
3. Wash thoroughly with warm water. Pat $ **the lemon** $ dry with a clean towel.	1	2.0
4. Wash thoroughly with warm water. Pat $ **lemon juice** $ dry with a clean towel.	0	2.0
5. Wash thoroughly with warm water. Pat $ **your stomach** $ dry with a clean towel.	0	1.0

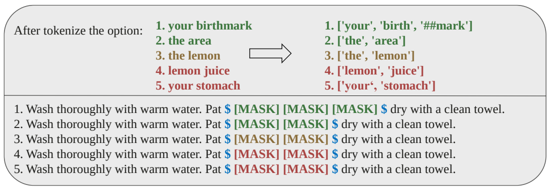
Figure 2: Data example after processing

To get more information that might be useful, we tried a variety of sentence concatenations using different columns in the data. Our experiments show that this is necessary and effective.

2.2 Unsupervised approach

First, in order to get a reliable benchmark on this task, we use unsupervised methods to try to solve the task with BERT. Because the pre-training process of BERT includes masked language model, that is, to replace a small part of words in the text with [MASK], and let the model predict the words replaced by [MASK]. This task is very similar to cloze, so we can use cloze to test BERT's masked language model capability in longer and more [MASK] texts (Ding et al., 2021).

For this task, we tokenize each option to get the number of tokens, and then fill in the blank with the same [MASK] as the number of tokens. We do this because multiple [MASK] work better than a single one. The processing process is shown in the following Figure 3:


After tokenize the option:

1. your birthmark	**1. ['your', 'birth', '##mark']**
2. the area	**2. ['the', 'area']**
3. the lemon	**3. ['the', 'lemon']**
4. lemon juice	**4. ['lemon', 'juice']**
5. your stomach	**5. ['your‘, 'stomach']**

1. Wash thoroughly with warm water. Pat $ **[MASK] [MASK] [MASK]** $ dry with a clean towel.
2. Wash thoroughly with warm water. Pat $ **[MASK] [MASK]** $ dry with a clean towel.
3. Wash thoroughly with warm water. Pat $ **[MASK] [MASK]** $ dry with a clean towel.
4. Wash thoroughly with warm water. Pat $ **[MASK] [MASK]** $ dry with a clean towel.
5. Wash thoroughly with warm water. Pat $ **[MASK] [MASK]** $ dry with a clean towel.

Figure 3: Data example after filled with [MASK]

A pooling operation is added to the output of BERT to generate a fixed-size sentence embedding vector. The tokens embedding of [MASK] obtained from the pre-trained model would be used for classification with different pooling strategies. In our experiment, three pooling strategies were used for comparison:

- MEAN strategy
  Calculate the average value of each token output vector of option to represent the sentence vector.
- MAX strategy
  Take the maximum value of each dimension of all output vectors of option to represent the sentence vector.
- SUM strategy
  Take the sum value of each dimension of all output vectors of option to represent the sentence vector.

2.3 Supervised approach

After getting a benchmark with an unsupervised method, we want to get some experimental results with a supervised method. First, we still conduct some experiments to screen out the model with better performance from several models. The models we use include Deberta-v3, Roberta-large, SBERT, BERT (Devlin et al., 2018), etc. The final experimental results will be displayed in the experimental section.

And then, in the above part, we mentioned filling in the blanks with [MASK], and using the embedding of [MASK] is directly used for classification. This is naturally associated with the similarity between [MASK] and options. Intuitively, [MASK] should be the most similar to the plausible option, and the least similar to the implausible option. So we use SBERT to calculate the similarity between the sentences after filling in [MASK] and filling in the options. After getting the similarity, we classify it by threshold optimization. The schematic diagram of the data process is shown in the following Figure 4:

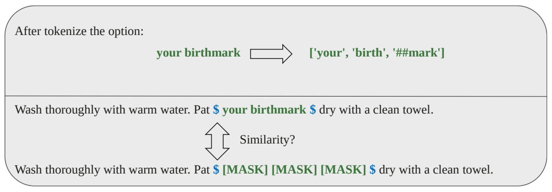

After tokenize the option:
your birthmark
['your', 'birth', '##mark']
Wash thoroughly with warm water. Pat $ your birthmark $ dry with a clean towel.
Similarity?
Wash thoroughly with warm water. Pat $ [MASK] [MASK] [MASK] $ dry with a clean towel.

Figure 4: Data process for calculate

The sentence obtained by filling [MASK] into the blank part and the sentence obtained by filling the option into the blank part are used as the input of the model, and then the embedding representation of [MASK] and the option is obtained by average pooling, as u and v respectively. We concatenate the values of u and v and the absolute value of their differences for classification tasks, and we also calculate the similarity between u and v for the task. The process is shown in Figure 5.

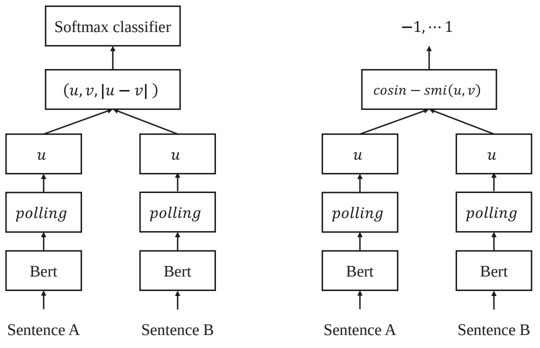


Figure 5: The process of SBERT

2.4 Model ensemble

Through Sections 2.3.1 and 2.3.2, we have obtained the results of several models. By comparing the classification results between different models, there are large differences, which means that for the classification results of the same data, the Model I may classify it into IMPLAUSE, but the Model II may classify it as NEUTRAL. This makes it possible for us to further improve the classification effect through the model ensemble.

The voting method is an ensemble learning model that follows the majority principle, and reduces variance through the integration of multiple models, thereby improving the robustness and generalization ability of the model. We adopt the voting method commonly used in ensemble learning, which is an ensemble learning model that follows the principle of majority rule by the minority, and reduces variance through the integration of multiple models, thereby improving the robustness and generalization ability of the model. We used four models (Roberta based on unsupervised method, and Roberta-large, Deberta-v3, SBERT based on supervised method) as benchmarks for ensemble learning. The structure of ensemble model is shown in the Figure 6.

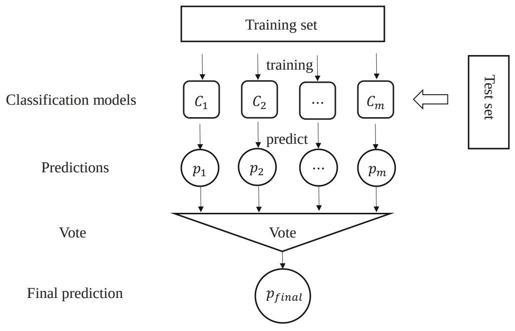


Figure 6: The structure of ensemble model

3 Experimental Results

In the following experimental part, all the data used for the experiment adopts the data processing method we introduced in Section 2.1.

3.1 Unsupervised approach results

We propose an attempt to use an unsupervised approach to benchmark this task in Section 2.2, and propose three strategies for dealing with [MASK]. The experimental comparison of the three strategies is gave by Table 1, there is little

Model	Pooling	Train Accuracy	Dev Accuracy
Bert-large	Mean	0.4373	**0.5152**
Bert-large	Sum	0.4434	0.4920
Bert-large	Max	0.4330	0.5150
Roberta-large	Mean	0.4678	**0.5788**

Table 1: Pooling strategy for unsupervised approach

Model	Train Accuracy	Dev Accuracy
Bert-base	0.5071	0.5394
Bert-large	0.5599	0.5613
Roberta-large	0.5260	0.5710
Deberta-v3	0.4403	**0.6326**

Table 2: Screen out the model with better performance from several models with [CLS] embedding

Model	Strategy	Train Accuracy	Dev Accuracy
Roberta-large	Classification	0.5463	0.5665
Roberta-large	Similarity	0.6337	**0.6272**
Bert-base	Classification	0.6298	0.5237
Bert-base	Similarity	0.7034	0.4553

Table 3: Experiment results of SBERT

Model	Train Accuracy	Dev Accuracy	Test Accuracy
Roberta-large unsupervised	0.4678	0.5788	--
Roberta-large supervised	0.5260	0.5710	--
Deberta-v3	0.5624	0.6485	0.622
SBERT(Roberta-large)	0.6337	0.6272	--
Ensemble	--	**0.7088**	**0.661**

Table 4: Results for subtask1 with ensemble model

Model	Train Rank	Dev Rank	Test Rank
Roberta-large unsupervised	--	0.6112	--
Roberta-large supervised	--	0.6370	--
Deberta-v3	0.6137	**0.7784**	0.747
SBERT(Roberta-large)	--	0.6560	--
Ensemble	--	0.7752	**0.774**

Table 5: Results for subtask2 with ensemble model

difference between MEAN strategy and Max strategy. We ended up using MEAN strategy to get a benchmark (57.88% Accuracy Score) with Roberta-large.

3.2 Supervised approach results

For supervised methods, although we did some experiments to try to find a better embedding than [CLS] for this task, we didn't get it. So we still ended up screening out the model with better performance from several models with [CLS] embedding. The results of model screening are given in Table 2.

In Section 2.3 we propose to use SBERT to try to solve this task. We conduct experiments with direct classification and computing similarity respectively. Table 3 gave the experimental results of SEBRT.

3.3 Model ensemble results

After obtaining several benchmarks using the unsupervised method and the supervised method, respectively, in the supervised method, by adjusting the parameters of several models, such as adjusting the batch size, learning rate or freezing some parameters in the model. We end up with 11 results including the above for ensemble learning。Due to space limitations, we no longer list the training results after adjusting the model parameters or freezing some parameters here. For each model, we list its best results. The results are given in Table 4.

For subtask 2, we just converted the above model from a classification task to a regression task, and also adjusted the training parameters, froze some model parameters, and obtained eleven kinds of results. The ensemble learning is carried out by the method of averaging, and the final result is obtained and shown in Table 5.

3.4 Discussion

A phenomenon can be observed from the experimental results: when the model has not fully converged on the training set, the best result of the model on the validation set has already appeared. Especially when using Deberta-v3, when the best results (63.26%) appear on the validation set, the model's accuracy score on the training set is only 44.03% in the classification task. This phenomenon also occurs in the regression task. But the difference is that the difference between the results of the training set and the validation set of the model in the classification task is much smaller than that in the regression task.

We therefore consider that there is noise in the training set, which is especially evident in classification tasks. To verify that there is really noise in the data, we compared part of the data in the training set with the data on the wikiHow website, as shown in the figure below. It can be seen that the sentences that appear in the original text in time are still marked as Neutral or Implausible in the training set. Figure 7 shows an example of original data with id-20 that may be incorrect.

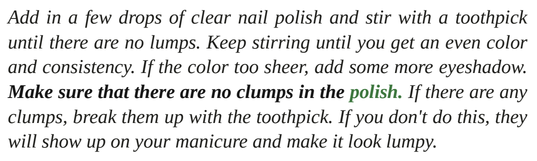

Add in a few drops of clear nail polish and stir with a toothpick until there are no lumps. Keep stirring until you get an even color and consistency. If the color too sheer, add some more eyeshadow. ***Make sure that there are no clumps in the polish.*** *If there are any clumps, break them up with the toothpick. If you don't do this, they will show up on your manicure and make it look lumpy.*

Figure 7: Data in the original paragraph

We tried Label Smoothing (Müller et al., 2019) and Self-Adaptive Training (Huang et al., 2020) to solve the problem of data noise. Although there is no significant improvement in the model's performance, it speeds up the model Convergence rate during training.

4 Conclusion

In this paper, we describe the Identifying Plausible Clarifications of Implicit and Underspecified Phrases shared task held within SemEval-2022 and present the design, the data, the results, and the systems for the shared task. The participants of the shared task were provided a collection of revisions of instructional texts from the how-to website wikiHow. The shared task is challenging, partly due to the relatively small training data and label noise.

We develop an ensemble model of NLP to distinguish between plausible and implausible clarifications of an instruction, achieving the 2nd best performance in the subtask1 and the 3rd best performance in the subtask2. For some of the problems reflected in this task, such as data noise, non-identically distributed data, there is still a lot of research space.

References

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised learning of narrative event chains. In *ACL 2008, Proceedings of the 46th Annual Meeting* *of* ***the Association for Computational Linguistics, June*** *15-20, 2008, Columbus, Ohio, USA*, pages 789–797. The Association for Computer Linguistics.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James F. Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In NAACL HLT 2016, *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 839–849. The Association for Computational Linguistics.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7*, 2021. OpenReview.net.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Minjie Ding, Mingang Chen, Wenjie Chen, and Lizhi Cai. 2021. English cloze test based on BERT. In *Knowledge Science, Engineering and Management -14th International Conference, KSEM 2021, Tokyo, Japan, August 14-16, 2021, Proceedings, Part II*, volume 12816 of *Lecture Notes in Computer Science*, pages 41–51. Springer.

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

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4696–4705.

Lang Huang, Chao Zhang, and Hongyang Zhang. 2020. Self-adaptive training: beyond empirical risk minimization. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.*

Michael Roth, Talita Anthonio, and Anna Sauer. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022).*

DuluthNLP at SemEval-2022 Task 7: Classifying Plausible Alternatives with Pre–trained ELECTRA

Samuel Akrah & Ted Pedersen
Department of Computer Science
University of Minnesota
Duluth, MN 55812 USA
{akrah001, tpederse}@d.umn.edu

Abstract

This paper describes the DuluthNLP system that participated in Task 7 of SemEval-2022 on Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts. Given an instructional text with an omitted token, the task requires models to classify or rank the plausibility of potential fillers. To solve the task, we fine–tuned the models BERT, RoBERTa, and ELECTRA on training data where potential fillers are rated for plausibility. This is a challenging problem, as shown by BERT-based models achieving accuracy less than 45%. However, our ELECTRA model with tuned class weights on CrossEntropyLoss achieves an accuracy of 53.3% on the official evaluation test data, which ranks 6 out of the 8 total submissions for Subtask A.

1 Introduction

Instructional texts (e.g., How To Guides) describe how to accomplish a given goal and are integral to our daily lives. One popular source is WikiHow[1] which is an online platform that allows users to collaborate to create and maintain such guides. This kind of documentation must be clear, and if it is not then this is a key reason that prompts revisions of underspecified instructions in WikiHow (Anthonio et al., 2020a).

One important problem for NLP is to determine if a given instructional text is in need of clarification or revision. SemEval-2021 Task 7 (Roth et al., 2022) extends this problem by requiring models to score five possible fillers based on how well they can plausibly fit a given context. Task 7 includes two subtasks. Subtask A classifies the possible fillers as IMPLAUSIBLE, NEURAL, and PLAUSIBLE. Subtask B requires systems to rank the fillers on a scale of 1 to 5, where a higher score means more plausible. We only participated in Subtask A and used a variety of BERT-based methods.

[1]https://www.wikihow.com

2 Task Data

The training, development and test data were supplied by the organizers of SemEval–2022 Task 7 (Roth et al., 2022). The dataset is based on the WikiHowToImprove Corpus (Anthonio et al., 2020b) which consists of edits of 2.5 million sentences from WikiHow. The authors show that edits are primarily made to clarify instructional texts and that the distinction between older and revised versions of sentences can be modelled computationally.

Each instance in the data is divided into an Article title, a Section header, and a Sentence nested between a Previous and Future context. Each instance also includes five potential fillers, each of which is annotated as IMPLAUSIBLE, NEUTRAL, or PLAUSIBLE, which serves as the basis of Subtask A. There is a also a plausibility score of 1 to 5 which is the basis of Subtask B (which we did not participate in).

Table 1 shows 3 training example templates made up of the concatenation of the Previous context, the Sentence with the blank to be filled, and the Future context. Each template is filled with each of the five possible fillers to generate the training examples (5 per template). We start with 3,995 training templates where the filler is not specified. We create an instance for each of the five possible fillers for each template, giving us a total of 19,975 training examples. We remove extraneous content such as bullet points and numbers in order to make the examples more readable. In a later approach, we highlight each filler in the generated instance with a special "[filler]" token, a step that yields further performance gains on the development data (Table 3) but achieves no corresponding gains on the official test results.

3 Methodology

We experimented on three large pre–trained language models for Subtask A, including BERT,

Previous context	Sentence	Followup context
State what you have contributed to the company. By doing it this way, it is going to show that you have done your job and been an asset, thus the raise is well-deserved. *	**P:** If you believe your [...] of time working at this company warrants a raise or promotion, say that as well. **Fillers: A. continuity B. window C. abundance D. length E. appreciation**	It is best to tell them all the reasons you believe you deserve this increase.
An all weather strategy often keeps you always afloat compared to one planned for normal market behavior. Planning for a failure is always better than failing to plan	**P:** Uncertainties of [...] can be classified into four levels **Fillers: A. public opinion B. future markets C. the future D. this sort E. their futures**	Level one gives a fairly clear view of the future, and an inkling of what to expect.
Since wikis are often volunteer-driven projects, wikigifts can go a long way in showing someone how much you appreciate their efforts. Find or create awards specific to that wiki.	**P:** On wikiHow, for example, you can Make Award Templates on [...] and post them on people's talk pages. **Fillers: A. books B. facebook C. graph D. wikiHow E. earth**	Publicize that you gave the wikigift.

Table 1: Training Example Templates for SemEval–2022 Task 7. Each possible replacement (filler) for the omitted token [...] must be ranked as PLAUSIBLE (blue), NEUTRAL(orange), or IMPLAUSIBLE (RED).

RoBERTa and ELECTRA. BERT (Bidirectional Encoder Representational from Transformers) (Devlin et al., 2018), is a Transformer-based language model trained using Masked Language Modeling (MLM) to predict the masked tokens based on the surrounding context. In MLM, a given percentage of the tokens of an input sequence is masked, and BERT is tasked to predict the orginal tokens. With the MLM approach, BERT was able to produce good results when transferred to downstream NLP tasks, becoming the new benchmark for other pre–trained models. The authors of the ELECTRA paper (Clark et al., 2020), however, note that the MLM approach only learns from the masked tokens (about 15%) of any given example, thus requiring substantial compute resources to train a language model using MLM.

The next model we used was RoBERTa, which is essentially a replication of BERT (Liu et al., 2019) which adjusts key hyperparameters and uses larger amounts of data during pre–training. RoBERTa improves upon BERT when trained longer, using larger mini–batches over more data.

Similarly, we experimented with ELECTRA (Clark et al., 2020), a pre–trained model that uses Replaced Token Detection as a pre–training objective. This distinguishes real inputs from plausible but synthetically generated ones coming from a small masked language model. The authors argue that ELECTRA improves compute efficiency during pre–training, and can match or exceed the performance of BERT and its variants when fine–tuned on downstream tasks.

The ELECTRA model includes two Transformer models, a generator and a discriminator. The generator emulates a small Masked Language Model by predicting the original token of a masked-out token. Like the Masked Language Model in BERT, some samples of the input sequence to the generator are replaced with [MASK]. The predicted results of the generator are fed as inputs to the discriminator.

For each token in the sequence, the discriminator predicts whether it is the original or the generated one. This means that the discriminator is able to learn from all input tokens for any given example, with the model loss calculated over all the tokens. This is what sets ELECTRA apart from BERT, and a major reason for ELECTRA's greater compute efficiency.

ELECTRA addresses a drawback of Masked Language Models, which is that the masked tokens are only used during pre–training and are omitted during fine–tuning. This pre–train fine-tune mismatch contributes to a loss in performance. Replaced Token Detection, in which ELECTRA distinguishes between real tokens and their plausible fakes, is easily transferable to fine–tuning. This is particularly true for Subtask A of SemEval-2022 Task 7, which requires a model to classify how fillers will plausibly fit an omitted token.

4 System Description

This section introduces DuluthNLP's approach which relied on BERT, RoBERTa, and ELECTRA.

Hyperparameter	Value
Learning Rate	4e-5
Adam ϵ	1e-8
Optimizer	AdamW
Learning rate decay	Linear
Weight Decay	0
Batch Size	16
Train Epochs	10

Table 2: Hyperparameter Values for Fine-Tuning.

Model	Accuracy
ELECTRA with class weights	0.556
RoBERTa with class weights	0.552
BERT with class weights	0.522
ELECTRA	0.443
BERT	0.441
Logistic Regression	0.348
Random Guessing	0.267
RoBERTa	0.177

Table 3: Experimental Results on Development Data.

We discuss our fine–tuning process, and then how class weights were tuned.

4.1 System Description

We first fine–tune BERT on Subtask A. Using our pre–processed dataset as inputs, we build and train a classifier on top of the BERT model to learn how plausible each filler fits the blank in each sentence.

Using the BERT-base uncased tokenizer on our inputs, we then train our BERT model using the Adam Optimizer with a linear scheduler with warmup; a CrossEtropy Loss with adjusted class weights; and a learning rate of 4e-5 for 10 epochs (see Table 2). We train our model twice, once without class weight tuning, and a second time with tuned class weights.

We used these same hyperparameters for fine-tuning RoBERTa (Liu et al., 2019) and ELECTRA (Clark et al., 2020). As our experimental results will show, the most accurate results were obtained with ELECTRA.

We used the HuggingFace PyTorch implementations of the BERT, RoBERTa, and ELECTRA (Wolf et al., 2019). We fine-tuned our models using 2 Nvidia Quadro RTX 8000 GPUs.

4.2 Class Weights

Class imbalances for classification tasks are often caused by imbalances in the dataset. However, for Subtask A the training data is reasonably balanced and includes 7339 (36%) PLAUSIBLE labels, 7162 (36%) NEUTRAL labels, and 5474 (27%) IMPLAUSIBLE labels.

While the task training data does not have significant imbalances, our model predictions on the development set initially skewed towards NEUTRAL. Over 52% of all the predictions were NEUTRAL, as shown in Table 4.

Our model corrects these imbalances by applying class weights that penalize NEUTRAL labels. This helped to reduce predictions for NEUTRAL labels to 28%, as shown in Table 5. This is at least closer to the actual distribution of 18% NEUTRAL in the development data and helps to improve accuracy.

The selection of optimal weights was based on random search, which has been shown to be more efficient for parameter optimization than grid search (Bergstra and Bengio, 2012). To achieve this, we initially defined a 3-tuple list of random weights, and for each tuple, we set the class weights for the CrossEntropy Loss function and trained our model. From the list, we selected the class weights with the best performance for further tuning.

5 Experimental Results

In this section we present the results of our models on both the development data and the test data as used in the official evaluation scored by the task organizers.

We used the Logistic Regression model from scikit-learn[2] as a baseline method. It incorporates binarized Ngram counts (Wang and Manning, 2012).This obtained an accuracy of 34.8% on the development data for Subtask A.

As shown in Table 3, accuracy on the development data without class weights were lower. The RoBERTa model, in particular, achieved accuracy of 17.7%, the lowest among the three language models, and even lower than random guessing.

However, with tuned class weights, all three of the pre–trained models achieved accuracy above 50%. ELECTRA and RoBERTa obtained nearly identical scores of 55.2% versus 55.6%. We decided to use ELECTRA as our official evaluation

[2] https://scikit-learn.org/

		Predicted			Total	%
		0	**1**	**2**		
Actual	**0**	249	580	140	969	38%
	1	53	255	135	443	18%
	2	45	456	587	1088	44%
	Total	347	1291	862	2500	
		14%	52%	34%		

Table 4: Confusion Matrix with ELECTRA Before Weight Tuning on Development Data. The labels (0,1,2) refer to (IMPLAUSIBLE, NEUTRAL, PLAUSIBLE) respectively. Accuracy is 42.4%.

		Predicted			Total	%
		0	**1**	**2**		
Actual	**0**	332	340	297	969	38%
	1	70	128	245	443	18%
	2	60	221	807	1088	44%
	Total	462	689	1345	2500	
		18%	28%	54%		

Table 5: Confusion Matrix with ELECTRA After Weight Tuning on Development Data. Accuracy is 50.7%.

method because of its very consistent performance with the class weights adjustment and its somewhat lower energy consumption as compared to other large language models.

Our official results on the Subtask A evaluation data for Task 7 Subtask A were 53.3% with our ELECTRA model with class weights. The top ranked system in the task obtained accuracy of 68%. DuluthNLP ranked 6 among 8 systems.

6 Error Analysis

The classes predicted by our model on the development data prior to the official evaluation were skewed to NEUTRAL, as discussed earlier. We observed this with various different pre–trained models including BERT, RoBERTa, and ELECTRA with various different hyperparameter settings. Despite our best efforts the DuluthNLP system never reached accuracy above 45%.

We addressed this by adding class weights to CrossEntropyLoss function used in our models. When we assigned class weights of [1.5, 0.03, 0.7] for the IMPLAUSIBLE, NEUTRAL, and PLAUSIBLE labels, the wrongly predicted scores for the NEUTRAL label reduced to 28%, as shown in Table 5.

What is curious, though, is the difficulty in correctly classifying the IMPLAUSIBLE label. This represents 38% of the actual labels but is 14% of the predicted labels. Even after class weights are set as described above, only 18% of the labels are predicted to be IMPLAUSIBLE, which is still a difference of 20% from the actual IMPLAUSIBLE labels.

We achieved further performance gains on the devset across all the models by highlighting the filler in each data instance with a special "[]" symbol, and this forms the basis for our results in Table 3. This approach did not distribute well over the test data, however.

7 Ethical Considerations

The training of large language models has a dark side: demands for large amounts of compute power and the corresponding energy consumption. Training BERT with the Masked Language Model requires a lot of computational resources.This raises concerns over the accessibility, cost, and environmental impact of such methods (Bender et al., 2021). Whilst we experimented with three BERT-variants, we sought to limit model fine–tuning to the base models (Bert-base, RoBERTa-base, and ELECTRA-base), which require less compute resources than their larger versions. The ELECTRA model, which we used for our official evaluation test results, is computationally efficient, which partly informed our choosing it over the other models for use as our official method during the evaluation stage.

The accuracy of our models are, at best, a little above 50%. This means that roughly half of any of these predictions may be wrong. This is clearly not accurate enough to deploy in a real setting without potentially causing harm. It is easy to imagine the negative impacts of automatically clarified instructions that prove to be inaccurate.

Similarly, the test data and the training set are from the same sample distribution, and we cannot guarantee that our model will achieve similar results for any out-of-distribution test data. In other words, our model, reliant as it is on the contextual representations provided by the pre–trained models, cannot perform well on a completely new task or distribution.

We relied on large language models which were trained on very large corpora. Such text may include stereotypes and biases which are then carried over into the resulting model (Bender et al., 2021). This locks the model to older, less–inclusive understandings that may not reflect more modern views of gender, race, or other questions of identify. To minimize the potential harms of such misrepresentations it would be best if candidate predictions were verified by a human editor.

However, if models like these were deployed and used to auto–suggest revisions to WikiHow, they may constrain the choices of human editors (Miller and Record, 2017). This could create a mindset that uncritically accepts the framed options of the model as legitimate (Alfano et al., 2018). The reviewer may then abandon their own edits because the auto–suggestion seems to provide an answer. One way to minimize such risk is the deployment of Reflection Machines (Cornelissen et al., 2022), a decision support system that will compel users to give reasons for accepting or rejecting suggestions from the model.

Acknowledgements

The authors would like to thank the organizers for the opportunity to participate in SemEval-2021 Task 7. We are also grateful to the three anonymous reviewers for their thoughtful comments and feedback. Finally, we would like to thank Dr. Alexis Elder for her contributions regarding the ethical considerations of this work.

References

Mark Alfano, J. Adam Carter, and Marc Cheong. 2018. Technological seduction and self-radicalization. *Journal of the American Philosophical Association*, 4(3):298–322.

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020a. wikiHowToImprove: A resource and analyses on edits in instructional texts. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5721–5729, Marseille, France. European Language Resources Association.

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020b. wikiHowToImprove: A resource and analyses on edits in instructional texts. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5721–5729, Marseille, France. European Language Resources Association.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(10):281–305.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

N. A. J. Cornelissen, R. J. M. van Eerdt, H. K. Schraffenberger, and W. F. G. Haselager. 2022. Reflection machines: Increasing meaningful human control over decision support systems. *Ethics and Inf. Technol.*, 24(2).

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

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

Boaz Miller and Isaac Record. 2017. Responsible epistemic technologies: A social-epistemological analysis of autocompleted Web search. *New Media & Society*, 19(12):1945–1963.

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Sida Wang and Christopher Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, Jeju Island, Korea. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Stanford MLab at SemEval 2022 Task 7: Tree- and Transformer-Based Methods for Clarification Plausibility

Thomas Yim[*], **Junha Lee**[*], **Rishi Verma**, **Scott Hickmann**, **Annie Zhu**, **Camron Sallade**, **Ian Ng**[†], **Ryan Chi**[†], and **Patrick Liu**[†]
Stanford University
{yimt, junhalee, rishirv, hickmann, anniezhu}@stanford.edu,
{camron, iyhn8192, ryanchi, pliu1}@stanford.edu

Abstract

In this paper, we detail the methods we used to determine the idiomaticity and plausibility of candidate words or phrases into an instructional text as part of the SemEval Task 7: Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts. Given a set of steps in an instructional text, there are certain phrases that most plausibly fill that spot. We explored various possible architectures, including tree-based methods over GloVe embeddings, ensembled BERT and ELECTRA models, and GPT 2-based infilling methods.

1 Introduction

The internet is filled with instructional texts from websites like wikiHow that detail how to perform a variety of tasks (from tying a bow tie to building a deck of stairs). With this increase of quantity and use of instructional texts, it has become increasingly important for them to be clear and unambiguously worded. To this end, we evaluate whether lightweight and neural models are capable of detecting which phrases most plausibly fit into a given series of instructions.

The current revision process requires that a reader potentially identify something wrong about the content, and they report it to the website for alterations. A system such as the one described could automatically identify candidates that are unlikely to exist and report them for human verification.

In this paper, we describe our lightweight and transformer-based models that rank the plausibility of candidate phrases given some previous context.

2 Background

2.1 Task Setup

The data for this task was provided by SemEval Task 7 (Roth et al., 2022). This dataset is an augmented version of WikiHowToImprove (Anthonio et al., 2020), which consists of 2.7 million sentences and their revision histories extracted from the instructional website WikiHow. The dataset extracts over 4000 sentences and revision that represent clarifications of the original text. Each sentence is masked and presented with 5 possible fillers that may represent a clarification. The article title, subsection, previous context, and future context are also provided. Each filler is annotated with a plausibility class label (either IMPLAUSIBLE, NEUTRAL, or PLAUSIBLE), whose prediction is the basis of Subtask A, and a plausibility score on a scale from 1 to 5, whose prediction is the basis of Subtask B.

The plausibility of a given filler is highly dependent on the context. For example, the clarification *birthday* is annotated as implausible for the sentence

2. Send a _____ card. Even if you're able to have the conversation in person, it's worth sending a card.

given the previous context

1. Ask in person whenever possible. Receiving an invitation to be a groomsman is exciting.

as the previous context implies a wedding.

Given this, we simply concatenate each of the provided data inputs into a single string as input for our models.

3 System overview

3.1 Word Embeddings

We first implemented a GloVe-based method (Pennington et al., 2014). To embed all the words, we used the Python Natural Language Toolkit (nltk) library to handle tokenizing all words. These words were pre-processed using a Punkt sentence tokenizer that can handle stripping punctuation from boundaries that would not affect the semantics of the phrase or sentence. We then fit various classical ML models, as listed in Section 3.2, to make

[*]Co-first authors.
[†]Co-senior authors.

predictions on the word vector inputs.

3.2 Lightweight models

- **Ridge regression** is a classification algorithm that minimizes the residual sum of squares.
- **Random forest** is a supervised learning technique that ensembles independent decision trees to yield a result.
- **Gradient Boosting** is a technique that ensembles a number of weak learners (typically decision trees) and optimizes based on a differentiable loss function.
- **Discriminant analysis** is a generative learning algorithm that assumes the data is distributed according to a Gaussian distribution.
- **Multilayer Perceptron** is a multi-layer artificial neural network.

3.3 BERT-like models

BERT is a language representation model first introduced in 2018 that has achieved state-of-the-art results in NLP experiments (Devlin et al., 2018). The model follows a multi-layer transformer-based encoder architecure. It leverages bidirectional self-attention, which enables it to learn context from both preceding and following sentences. Furthermore, it was trained on language modeling tasks – predicting masked tokens from context – which makes it an ideal model for this task. We fine-tune the BERT model to learn the plausibility of various possible fillers.

To do so, we input both the masked sentence with context information (chosen out of resolved pattern, article title, section header, previous context, and follow-up context) and the possible filler, with each element separated by a special separator token. We then train a shallow neural network on top of BERT to recognize the plausibility or implausibility of the filler based on BERT's encoded representation.

In our data set, we used the pre-trained bert-base-uncased tokenizer to prepare the words for the model. In our BERT Class, we used the Sigmoid activation function and added dropout to prevent over-fitting of the model. We froze the first 8 layers of the BERT model and then trained using Cross Entropy Loss, an Adam optimizer function, and a learning rate scheduler. As the model was trained, the program kept track of the best models with the highest performance accuracy. It would check with each epoch and save the model if it improved.

Alongside BERT, we also experiment with other similar BERT-based language models, like ALBERT (Lan et al., 2019) and RoBERTa (Liu et al., 2019). ELECTRA uses the same underlying model as BERT, but through a different pre-training mechanism. This approach corrupts the input by replacing tokens with alternatives, rather than masking it, and trains the model to determine which tokens were corrupted versus part of the original sentence (Clark et al., 2020). This makes ELECTRA another ideal candidate for determining the plausibility of certain fillers. Each of these models was fine-tuned and evaluated in the same manner as the original BERT.

The ELECTRA model of the highest accuracy was achieved using the ELECTRA-small discriminator, two linear layers, a hyperbolic tangent activation function, and a dropout rate of 0.5 before each linear layer. The context information used was the previous and follow-up context.

3.4 Infilling by Language Modeling

This approach is taken from (Donahue et al., 2020). Training data for this model was created by randomly masking words or phrases in a body of text and fine-tuning the GPT-2 model on those masked sentences. We combined the Article Title, Section Header, Previous Context, Sentence, and Follow-up Context into a single string and added an infill mask token where we are predicting the word or phrase. Then, given the surrounding the context, the model returned a set of logits and softmaxed probabilities that ranked the probability of all possible tokens. To handle multitoken words such as "jalapenos" ([474, 282, 499, 28380]), the logits were summed and the probabilities were mutliplied for each individual token. There were additionally phrases like "your hands" ([14108, 2832]) that were multi-token as well.

For the example "How to Store Jalapenos," the sentence needing clarification was "Make sure to wear latex gloves when handling jalapenos or wash [INFILL-WORD] thoroughly after handling." As follows are the options and their returned logits: "your hands": -6.1058 "the jalapenos": -70.2685 "the sun": -22.6745 "the floor": -19.1986 "your underwear": -18.6225. The model accurately determined "your hands" to be more probable than the other options like "the sun", "the floor", and "your underwear" which make little sense in this context. However, "the jalapenos", which received a medium plausible score of 3.0 in the training data was found to be extremely improbable with this model due to the high number of tokens. This was a flaw that was pervasive throughout the usage of this model and is why we decided to continue with the BERT and ELECTRA ensemble models instead.

After the logits were calculated for all the possible infills in the training data, we trained a Ridge linear model to convert from logits to our 1-5 scoring scale. Unfortunately, because of the inaccurate probabilities generated by multi-token words and phrases, there appeared to be no correlation between logits and their labeled scores, leading the Ridge model to predict around 3.33 as a baseline score for most words.

4 Experimental setup

We merge the provided train and dev sets, perform random 75:25 splits of the merged data to use for training and validation. We noted little difference in performance between different random splits.

Predictions were evaluated on accuracy for Subtask A and Spearman correlation for Subtask B. Although we also calculated other metrics such as macro-averaged F1 for Subtask A and mean squared error for Subtask B, we standardized on accuracy and Spearman correlation for consistency in comparing results. In particular, since the classes were close to balanced for Subtask A, using accuracy was not a big issue in overfitting to certain classes.

5 Results

As shown in Table 1, here was some variation in dev set performance between the lightweight models that we experimented with; in particular, Linear and Quadratic Discriminant Analysis performed better than the other models. However, even these performances are very low, achieving a maximal dev set accuracy of only 0.389 with Linear Discriminant Analysis.

After switching to BERT-like models, we generally achieve a significant improvement in classification accuracy. In particular, ELECTRA achieves the highest accuracy across all of our models, with a dev set accuracy of 0.465. An exception is RoBERTa, with which we achieve an accuracy close to that of Linear Discriminant Analysis.

The ILM-based models ultimately failed to improve upon the accuracy of the lightweight models. While the ILM approach might initially seem to be the most promising given the task, it seems that the model's ability to generate the top few most likely options did not correlate with its ability to compare relatively more unlikely potential infills. The model was also pre-trained on a stories database, which may not reflect the context appropriate for instructional texts like WikiHow articles.

For the regression subtask, we only experiment with BERT with a regression head. We find that BERT achieves a Spearman correlation of 0.149 on the dev dev set.

Our final evaluation results on both subtasks are shown in Table 2. Surprisingly, our evaluation scores are slightly higher than our development scores.

6 Conclusion

BERT-like models are the highest performing models for this type of instructional clarifications task. Because BERT has been trained to predict masked tokens, it is naturally better at finding words or phrases that most plausibly fit with the surrounding context. Simple GloVe word embedding models were unable to learn to the requisite complexity of this meta-linguistic task and were unable to break the level of 0.389 accuracy. Meanwhile, ILM-based approaches seemed promising, but in practice failed to accommodate phrases or long words. This method is more effective at generating text and not necessarily determining how plausible an infill sounds in that context.

In terms of future work, we believe that our system may be improved by the usage of large pretrained models such as SpanBERT (Joshi et al., 2019), which are trained using tasks involving multi-token span prediction, which may be more fit to the given task. Despite our lack of results, we believe that the infilling approach is still promising, and hope that it can be adapted to this task going forward.

Acknowledgements

This research effort would not have been possible without the support of Stanford ACMLab. The authors thank Michael Roth, Talita Anthonio, and Anna Sauer for organizing SemEval 2022 Task 7: Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts.

Model	Accuracy
K Nearest Neighbors	0.347
ILM	0.354
Ridge	0.357
Random Forest	0.363
MLP Classifier	0.365
Hist Gradient Boosting	0.366
Quadratic Discriminant Analysis	0.373
RoBERTa	0.387
Linear Discriminant Analysis	0.389
BERT	0.447
ELECTRA	**0.465**

Table 1: Dev set performance (Subtask A)

Subtask	Method	Result
A (Accuracy)	ELECTRA	0.496
B (Spearman R)	BERT	0.194

Table 2: Test set results.

Additionally, the authors would like to recognize Google Colaboratory for their free compute services.

References

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020. wikiHowToImprove: A resource and analyses on edits in instructional texts. In *LREC*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. *CoRR*, abs/2005.05339.

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

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

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

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Nowruz at SemEval-2022 Task 7: Tackling Cloze Tests with Transformers and Ordinal Regression

Mohammadmahdi Nouriborji
NODET, Iran*
mohammadmahdinoori70@gmail.com

Omid Rohanian
Department of Engineering Science
University of Oxford
Oxford, UK
omid.rohanian@eng.ox.ac.uk

David Clifton
Department of Engineering Science
University of Oxford
Oxford, UK
david.clifton@eng.ox.ac.uk

Abstract

This paper outlines the system using which team Nowruz participated in SemEval 2022 Task 7 "Identifying Plausible Clarifications of Implicit and Underspecified Phrases" for both subtasks A and B (Roth et al., 2022). Using a pre-trained transformer as a backbone, the model targeted the task of multi-task classification and ranking in the context of finding the best fillers for a cloze task related to instructional texts on the website Wikihow.

The system employed a combination of two ordinal regression components to tackle this task in a multi-task learning scenario. According to the official leaderboard of the shared task, this system was ranked 4th in both classification and ranking subtasks out of 21 participating teams. With additional experiments, the models have since been further optimised. The code used in the experiments is going to be freely available at https://github.com/mohammadmahdinoori/Nowruz-at-SemEval-2022-Task-7.

1 Introduction

Oxford dictionary defines cloze test as "a test of readability or comprehension in which a person is required to supply words which have been deliberately omitted from a passage" (Oxford University Press, 2022). In the context of NLP, a cloze format task is one in which the context is one or more sentences with masked spans and the model is expected to predict a suitable filler for each span. Cloze-format datasets have become popular in NLP recently as they are relatively easy to create automatically and provide high quality resources for model training (Rogers et al., 2021).

SemEval 2022 task 7 is framed as a cloze task in which the goal is to rank or classify fillers within a given context based on their suitability. The texts are taken from actual articles on an instructional website and the masked spans are placed at locations of edits made by users. We participated in this shared task in both ranking and classification parts and developed a transformer-based model that utilises both classification and regression components at the top layer. The code and the data used in these experiments are publicly available.

2 Related Work

Cloze tasks have been a subject of interest in Natural Language Understanding (NLU) in recent years, especially within the context of reading comprehension, story understanding, and summarisation (Deutsch and Roth, 2019; Sharma et al., 2018; Mostafazadeh et al., 2016). There is evidence that cloze tasks can be used to effectively pretrain or finetune language models in order to perform few shot learning (Schick and Schütze, 2021; Liu et al., 2021).

WikiHow is a community-edited open domain repository that hosts how-to articles on a variety of different subjects. It is possible to track edits made by users and compare different versions. There have been some recent computational works exploring this resource, including modelling of revision requirements (Bhat et al., 2020), and the effect of edits on fluency (Anthonio and Roth, 2020) and vagueness (Debnath and Roth, 2021).

2.1 Masked Language Modelling

Masked Language Modeling (MLM) is a pretraining task which is widely used in transformer-based models. This task forms a self-supervised

[0]National Organization For Development of Exceptional Talents

cloze test by randomly removing some of the tokens of the sentence which will be further filled by the model during training. Prominent transformers including BERT (Devlin et al., 2019) , RoBERTa (Liu et al., 2019), DeBERTa (He et al., 2020) and T5 (Raffel et al., 2019) are trained using MLM as an auxiliary objective. Given the successes of transformers in most areas of machine learning and NLP, it is standard practice to fine-tune them for various down-stream tasks.

2.2 Ordinal Regression

Ordinal Regression (also known as Ordinal Classification) is a type of classification in which labels have order with respect to each other. Examples of ordinal regression tasks include age estimation (Niu et al., 2016), assessment of damage (Ci et al., 2019), and monocular depth estimation (Fu et al., 2018). In ordinal regression, performance of the model is sensitive to the order of the predictions with regards to the labels. For instance, in the age estimation task, the error of the model should be higher when it incorrectly predicts the age of 30 as 10, as opposed to when it predicts the age of 30 as 20. Ordinal Regression is commonly done by breaking the multi-class classification task into several binary classification subtasks within a multi-task learning scenario. The output of these binary classification subtasks should be rank-consistent to achieve good performance.

3 System Description

Figure 1 shows the overall architecture of our model. A pre-trained transformer sits at the base of the architecture. Different variations of this transformer are tried and reported in Sec 5. These include BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), DeBERTa (v1 & 3) (He et al., 2020, 2021), and T5[1] (Raffel et al., 2019) in different configurations.

At the bottom of the network, there are two components, one for regression and the other for classification. At training time, both of these are trained in tandem using ordinal regression and a combined loss. The loss function ensures that the ranks of the labels are kept consistent across these task (Sec. 3.6).

[1]Since T5 is originally an encoder-decoder language model, we only use its pre-trained encoder for our experiments

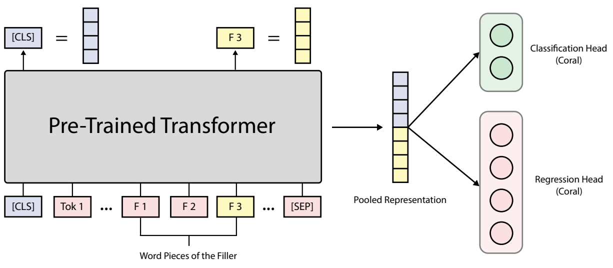


Figure 1: Overall architecture of the model. Note how a pooled representation is produced using the last token of the filler and the CLS token

3.1 Representing the Filler

To classify each filler, it is first placed in the location of the blank in the text to form a full context. The pre-trained transformer is equipped with a word-piece tokeniser that breaks down the filler to subword units and also uses a [CLS] token to represent the entire context [2]. To build a pooled representation for the filler, the contextualised representations for the last word piece and the entire context (i.e. representation of [CLS]) are concatenated and passed on to the next layer.

3.2 Multi-task learning with Ordinal Regression

Once a combined contextualised representation is obtained from concatenation of [CLS] and the filler, we address both subtasks of the shared task in a single multi-task learning architecture. The objective of this model is to predict both the class and the suitability score for each provided filler. The concatenated representation is fed to a fully connected feed-forward layer followed by a GELU activation function (Hendrycks and Gimpel, 2016). This layer projects the representation to a lower dimensional space. Subsequently, the output is passed on to two separate classification and regression heads.

3.3 Decomposing the Problem into Binary Classification tasks

As mentioned in Sec. 3, our model uses a special loss function named coral to perform ordinal regression. This loss function and the training procedure it requires are explained in Cao et al. (2020). Coral layers break a K-class classification problem into $K - 1$ binary classification tasks as part of a multi-task learning scenario. For a coral layer with $K - 1$

[2]T5 does not use CLS for context representation. In that case we just classify the last word-piece with no additional concatenation

units [3], the value of loss is constructed from the sum of $\mathrm{K}-1$ separate binary cross entropy losses belonging to each unit.

To train a coral layer, it is necessary to first transform the original labels to sets of binary labels. This is the step where the notion of order is introduced into the model. Given K classes, we convert each label to a collection of binary labels as follows:

$$f(y,k)=\begin{cases}1 & \text{if}\, k<y\\ 0 & \text{if}\, k\geq y\end{cases} \tag{1}$$

$$Y^{(i)}_{ordinal}=\{f(y^{(i)},k)|k\in\mathbb{W}\wedge k<\mathrm{K}-1\}$$

where $y^{(i)}$ is the original label for the ith training instance and $0\leq y^{(i)}\leq \mathrm{K}-1$.

3.4 Classification and Regression Heads

At the top end of the architecture there are two components for classification and regression. The classification head is a coral layer with two units which is used to address the classification subtask where we have three labels, namely, `Implausible`, `Neutral`, and `Plausible`. Note that the labels in the classification task have inherent order and for the model to be trained effectively, it is important that the training objective penalises misclassifications based on this underlying assumption. For instance, the error of the model should be more when it predicts an `Implausible` sample as `Plausible` compared to when it predicts an `Implausible` sample as `Neutral`. This is the motivation to use ordinal regression losses like coral.

The other head is assigned to the regression subtask and rates each filler for suitability. We converted continuous scores in the $1-5$ range to labels with discrete values in the $0-4$ range by either rounding or flooring the scores. For example, the original scores of $\{1.333, 1.75, 2.5, 3.75, 4.25\}$ are mapped to $\{0,1,2,3,3\}$ by rounding and $\{0,0,1,2,3\}$ by flooring. We can frame the regression task as ordinal classification with five labels by binning the values. A 4-unit coral layer is used to perform this classification. These heads are jointly trained using a combined loss.

[3]A T-unit coral layer, is comprised of T binary classification units which share the same weights but have different biases.

3.5 Constructing Labels and Ranks from Heads

Since coral layers differ from normal dense layers, their output can not be directly converted to labels. Furthermore, the regression subtask is also framed as an additional classification task in our methodology. However, since the purpose is to report continuous scores rather than discrete labels, a unique conversion is necessary for the output of the regression head.

For the classification head, the goal is to output discrete labels in range of $0-2$. Given the output of the classification head for one sample $\hat{C}$, the conversion to the label is defined as follows:

$$f(y)=\begin{cases}1 & \text{if}\, y>0.5\\ 0 & \text{if}\, y\leq 0.5\end{cases} \tag{2}$$

$$c=\sum_{k=1}^{2}f(\sigma(\hat{C}_k))$$

where c is the final label and σ is the sigmoid function.

For the regression head, the goal is to output continuous scores in range of $1-5$. Given the output of the regression head for one sample $\hat{R}$, the conversion to the continuous scores in range of $1-5$ is defined as follows:

$$r=(\sum_{k=1}^{4}\sigma(\hat{R}_k))+1 \tag{3}$$

3.6 Computation of Loss

Since we have three labels for the classification task, ordinal labels would be sets of binary values with the length of two. Given $C^{(i)}$ as the converted classification label for the i_{th} training sample and the $\hat{C}^{(i)}$ as the output of the classification head for the i_{th} training sample, we can define the classification loss as:

$$\begin{aligned} l^{(i)}_c=\sum_{k=1}^{2}&-C^{(i)}_k\log(\sigma(\hat{C}^{(i)}_k))\\ &-(1-C^{(i)}_k)\log(1-\sigma(\hat{C}^{(i)}_k))\end{aligned} \tag{4}$$

For the regression task there are 5 different labels, and accordingly, ordinal labels would be sets of binary values with the length of four. Given $R^{(i)}$ as the converted regression label for the i_{th} training sample and the $\hat{R}^{(i)}$ as the output of the regression head for the i_{th} training sample we can define the regression loss in a similar way:

$$l_r^{(i)} = \sum_{k=1}^{4} -R_k^{(i)} \log(\sigma(\hat{R}_k^{(i)})) \\ -(1 - R_k^{(i)}) \log(1 - \sigma(\hat{R}_k^{(i)})) \quad (5)$$

Since we are aiming to perform both classification and regression, a joint loss is needed to combine losses from the two heads. For a given training batch of size n:

$$L_{total} = \frac{1}{n} \sum_{i=1}^{n} \lambda_c l_c^{(i)} + \lambda_r l_r^{(i)} \quad (6)$$

where λ_c is the weight associated with the classification loss and the λ_r is associated with regression.

4 Dataset and Experimental Setup

The shared task is based on Anthonio et al. (2020), where instructional texts from the website Wikihow [4] are used to create training instances for a cloze task. The deletions in the dataset are based on actual edits made by online users and they are assumed to represent certain types of clarifications to make a point more clear or disambiguate a sentence. Based on the actual edits, alternative fillers are automatically extracted and added to build a number of possible fillers. These texts along with the fillers were later annotated by humans and given plausibility scores from 1 to 5. There is also a separate type of annotation available in which there are 3 labels with discrete plausibility values. The shared task was organised in two separate tracks of classification and regression, depending on what kind of annotation was used for modelling the task. Table 2 & 3 show the basic statistics of the dataset in the shared task.

The dataset for this task consists of six features, named Resolved Pattern, Article title, Section header, Previous context, Sentence, and Follow-up context along with five different fillers for each sample. Resolved Pattern is one of the four following categories: `IMPLICIT REFERENCE`, `ADDED COMPOUND`, `METONYMIC REFERENCE`, and `FUSED HEAD` which indicates the relationship of the fillers with the context. Article title is the name of the article from which the paragraph is selected. Section header is the section from which the article is selected. Previous context is a few sentences before the sentence that contains the filler. Sentence is the sentence that contains the filler.

We obtained our best results when we used a custom formatting using which we can feed all the features to the model as textual input. Table 1 is an example of how we represent each training instance. Note how the 'Text' feature is constructed by concatenating previous and follow-up contexts with the target sentence.

Example input while using all features
Resolved pattern: ADDED COMPOUND Section header: Following a Basic Routine Article title: How to Get Rid of Peeling Skin Text: (...) 6. Never tear away loose skin. (...) 7. Protect your skin from sunlight. Exposure to direct sunlight can weaken your skin further and complicate the [Filler] problem. This is true regardless of whether your skin is peeling due to a sunburn or due to dryness.

Table 1: An example of how each input is formatted for training and inference. Each identifier followed by ":" represents a feature of the dataset represented in textual format.

Variation	implausible	neutral	plausible	total
train	5474 (27.40%)	7162 (35.85%)	7339 (36.75%)	19975 (100%)
dev	982 (39.28%)	602 (24.08%)	916 (36.64%)	2500 (100%)

Table 2: Statistics of the data for subtask A

Variation		1	2	3	4	5	total
train	F	2254	4123	6259	5321	2018	19975
train	R	1053	4421	4034	8441	2026	19975
dev	F	645	458	481	596	320	2500
dev	R	386	596	376	639	503	2500

Table 3: Statistics of the data for subtask B. R and F represent rounded and floored scores, respectively

5 Results and Discussion

The official test results for both subtasks are presented in Table 5. These are the best results that we have obtained on the test set prior to the end of the competition. On the official leaderboard of the shared task, our system was ranked 4th in both the ranking and classification tasks.

We have since performed extensive analyses on the effect of hyperparameters on different varia-

[4]A wiki-style online collection of how-to articles accessible at `https://www.wikihow.com`

tions of our models, and since we do not have access to the true labels in the test set, we report our best results on the dev set. As can be seen in Table 4, we have produced our best results using DeBERTa-V3$_{large}$. We have noticed two important factors that influence the final performance of the models. The first factor is batch size. Our best results were obtained on lower batch sizes of 8 and 16. The choice of rounded or floored numbers for subtask B is also a significant factor. The reason for this is that the distribution of labels changes depending on the normalisation method used.

6 Conclusion

In this work we developed a set of transformer-based models powered with ordinal regression to tackle an NLP cloze task as part of the SemEval 2022 shared task 7. The goal was to assign suitability scores or labels to several different provided fillers given each context and masked span.

Using a combined architecture based on ordinal regression that used training labels from both subtasks, we developed and trained models with a multi-task learning objective. The proposed system was ranked 4th out of 21 teams in both tracks of the shared task. In the subsequent analyses in the post evaluation phase, we have showed the effectiveness of this architecture in addressing this task. We compared different variations of our models and explored the effects of hyperparameters on model performance. The code and analyses are going to be publicly available.

References

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020. wikiHowToImprove: A resource and analyses on edits in instructional texts. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5721–5729, Marseille, France. European Language Resources Association.

Talita Anthonio and Michael Roth. 2020. What can we learn from noun substitutions in revision histories? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1359–1370, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Irshad Bhat, Talita Anthonio, and Michael Roth. 2020. Towards modeling revision requirements in wikiHow instructions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8407–8414, Online. Association for Computational Linguistics.

Wenzhi Cao, Vahid Mirjalili, and Sebastian Raschka. 2020. Rank consistent ordinal regression for neural networks with application to age estimation. *Pattern Recognition Letters*, 140:325–331.

Tianyu Ci, Zhen Liu, and Ying Wang. 2019. Assessment of the degree of building damage caused by disaster using convolutional neural networks in combination with ordinal regression. *Remote Sensing*, 11(23):2858.

Alok Debnath and Michael Roth. 2021. A computational analysis of vagueness in revisions of instructional texts. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 30–35, Online. Association for Computational Linguistics.

Daniel Deutsch and Dan Roth. 2019. Summary cloze: A new task for content selection in topic-focused summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3720–3729, Hong Kong, China. Association for Computational Linguistics.

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

Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. 2018. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2002–2011.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

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

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Backbone Transformer	Model	Batch Size	Scores	Accuracy (subtask A)	Spearman's rank (subtask B)
BERT	Bert_{base}	8	R	57.80%	0.6020
	Bert_{base}	8	F	57.48%	0.6038
	Bert_{base}	16	R	56.68%	0.5950
	Bert_{base}	16	F	57.92%	0.5960
	Bert_{large}	8	R	57.60%	0.6323
	Bert_{large}	8	F	59.12%	0.6341
RoBERTa	RoBERTa_{base}	8	R	59.16%	0.6610
	RoBERTa_{base}	8	F	58.68%	0.6582
	RoBERTa_{base}	16	R	57.32%	0.6743
	RoBERTa_{base}	16	F	59.44%	0.6695
	RoBERTa_{large}	8	R	58.68%	0.6799
	RoBERTa_{large}	8	F	60.20%	0.6928
DeBERTa-V3	DeBERTa-V3_{base}	8	R	61.72%	0.7194
	DeBERTa-V3_{base}	8	F	62.64%	0.7260
	DeBERTa-V3_{base}	16	R	63.12%	0.7287
	DeBERTa-V3_{base}	16	F	63.84%	0.7326
	$\text{DeBERTa-V3}_{large}$	8	R	**64.12%**	0.7406
	$\text{DeBERTa-V3}_{large}$	8	F	**64.12%**	**0.7411**
T5	T5-Encoder_{base}	8	R	58.88%	0.6598
	T5-Encoder_{base}	8	F	59.48%	0.6573
	T5-Encoder_{base}	16	R	59.24%	0.6526
	T5-Encoder_{base}	16	F	59.08%	0.6568
	T5_{large}	8	R	61.76%	0.6949
	T5_{large}	8	F	62.24%	0.6907

Table 4: Results on dev set for both subtasks, R and F represent rounded and floored scores respectively (as mentioned in Sec. 3.4)

Model	Batch Size	Scores	Accuracy (subtask A)	Spearman's rank (subtask B)
RoBERTa_{large}	8	F	61.00%	0.6700
$\text{DeBERTa-V1}_{large}$	8	F	61.00%	0.6900
$\text{T5-Encoder}_{large}$	16	F	**62.40%**	**0.7070**

Table 5: Official results on leaderboard for both subtasks

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

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Zhenxing Niu, Mo Zhou, Le Wang, Xinbo Gao, and Gang Hua. 2016. Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Oxford University Press. 2022. cloze, n.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension. *arXiv preprint arXiv:2107.12708*.

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Rishi Sharma, James Allen, Omid Bakhshandeh, and Nasrin Mostafazadeh. 2018. Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 752–757, Melbourne, Australia. Association for Computational Linguistics.

A Appendix

In this section, we are going to share the details of the hyperparameters used for the fine-tuning of our models and the final training procedure used for the submission.

A.1 Hyperparameters

We fine-tuned all models for 5 epochs while keeping the embedding layers of all models frozen. We used the AdamW as our optimizer with a cosine learning rate schedular from the Hugging Face library and a weight decay of 0.00123974 and an initial learning rate of $1.90323e - 05$. Also, the λ_l and λ_r (as mentioned in 3.6) are set to 0.5 in all of the reported experiments as shown in Table 6.

Hyperparamters	
Epochs	5
Optimizer	AdamW
Learaning Rate Scheduler	Cosine
Initial Learning Rate	1.90323e-05
Weight Decay	0.00123974
λ_l	0.5
λ_r	0.5

Table 6: Details of the Hyperparamters used for fine-tuning

A.2 Training Procedure for Submission

Once we found the best hyperparameters and models using Dev dataset, We used a combination of Training and Dev data to train our final models for submission. With this approach, we have been able to achieve a 3% improvement on accuracy and up to 0.05 improvement on Spearman's rank correlation. Additionally, we found that combining Train and Dev data is noticeably less effective in the smaller models such as Bert$_{base}$ or RoBERTa$_{base}$, however, this is a better strategy when it comes to larger models such as RoBERTa$_{large}$ or T5$_{large}$. Based on this, we suppose that with further tuning. DeBERTa-V3$_{large}$ can potentially surpass the state-of-the-art if it has access to the combination of Train and Dev data.

X-PuDu at SemEval-2022 Task 7: A Replaced Token Detection Task Pre-trained Model with Pattern-aware Ensembling for Identifying Plausible Clarifications

Junyuan Shang[1], **Shuohuan Wang**[1], **Yu Sun**[1],
Yanjun Yu[2], **Yue Zhou**[2], **Li Xiang**[2] and **Guixiu Yang**[2]
[1] Baidu Inc., China
[2] Shanghai Pudong Development Bank, China
{shangjunyuan, wangshuohuan, sunyu02}@baidu.com
{yuyj6, zhouy93, xiangl3, yanggx1}@spdb.com.cn

Abstract

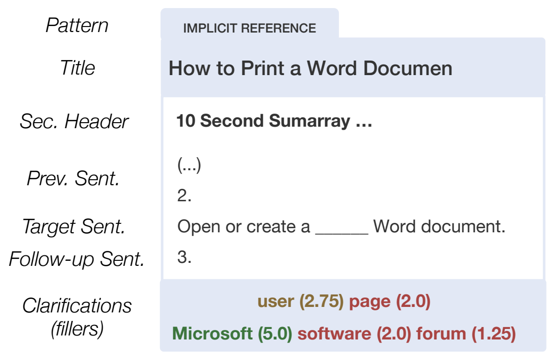
This paper describes our winning system on SemEval 2022 Task 7: *Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts*. A replaced token detection pre-trained model is utilized with minorly different task-specific heads for SubTask-A: *Multi-class Classification* and SubTask-B: *Ranking*. Incorporating a pattern-aware ensemble method, our system achieves a 68.90% accuracy score and 0.8070 spearman's rank correlation score surpassing the 2nd place with a large margin by 2.7 and 2.2 percent points for SubTask-A and SubTask-B, respectively. Our approach is simple and easy to implement, and we conducted ablation studies and qualitative and quantitative analyses for the working strategies used in our system.

1 Introduction

The Internet's ever-increasing size has made it easy to find instructional texts such as articles in wikiHow[1], on almost any topic or activity. Regular revisions of these how-to manuals are necessary to ensure that instructions communicate the procedures required to attain a certain goal precisely. This shared task is introduced by Roth et al. (2022), whose intention is to find ways to improve instructional texts, evaluate to what extent current NLP systems are able to handle implicit, ambiguous, and underspecified language, and go beyond the surface form of a text and take multiple plausible interpretations into account. Thus, the proposed NLP systems should be capable of distinguishing between plausible and implausible clarifications of an instruction shown in Figure. 1.

The shared task consists of two subtasks:

- **SubTask-A: Multi-Class Classification**. The goal is to predict a class label (IMPLAUSIBLE, NEUTRAL, PLAUSIBLE) given the clarification[2] and its context.
- **SubTask-B: Ranking**. The goal is to predict the plausibility score on a scale from 1 to 5 given the clarification and its context.



Pattern	IMPLICIT REFERENCE
Title	How to Print a Word Documen
Sec. Header	10 Second Sumarray ...
Prev. Sent.	(...) 2.
Target Sent.	Open or create a ______ Word document.
Follow-up Sent.	3.
Clarifications (fillers)	user (2.75) page (2.0) Microsoft (5.0) software (2.0) forum (1.25)

Figure 1: An example randomly chosen from the dev set. Each sample is associated with five clarifications labeled (PLAUSIBLE, NEUTRAL or IMPLAUSIBLE) and scored on a scale from 1.0 to 5.0.

In this paper, we describe our winning system for both subtasks. We built our system based on a replaced token detection (RTD) task pre-training model. The idea is that the replaced token detection task is similar to this shared task which focuses on distinguishing semantically similar words/phrases. To close the gap the model trained between the pre-training phase and fine-tuning phase, we reused the pre-trained language modeling head during fine-tuning on Task 7. Then, two-layers MLPs are applied on the mean-pooled hidden states of a clarification (filler) given the context. For subtask A, we utilized the cross-entropy loss for the multi-class classification. For subtask B, a sigmoid function was used to impose restrictions on the output of the system on a scale from 1 to 5. Finally, we

[1]https://www.wikihow.com/Main-Page

[2]A clarification is a word/phrase that was inserted to specify information in the instruction.

trained multiple models and aggregated the predictions with a pattern-aware ensemble strategy. Our system achieved the best overall performance in the shared task with a 68.9% accuracy score (subtask A) and 0.807 Spearman's rank correlation score (subtask B). The outcomes are promising for improving the clarification of instructional texts.

2 Background

Pre-trained models (Devlin et al., 2018; Liu et al., 2019; Sun et al., 2019; Clark et al., 2020) have achieved state-of-the-art results in various Natural Language Processing (NLP) tasks. Recent works (Raffel et al., 2019; Brown et al., 2020; Sun et al., 2021) have shown that more generalization ability and superior performance can be achieved by pre-training models with billion or trillion parameters. Thus, we pursued the competitive pre-trained models such as DeBERTa (He et al., 2020) and large-scale pre-trained models ERNIE (Sun et al., 2021) whose effectiveness has been validated in the standard GLUE (Wang et al., 2018) and SuperGLUE benchmark (Wang et al., 2019).

However, in our initial experiment, we found that the aforementioned models, though have promising results on sentence-level or paragraph-level tasks, failed to distinguish the word/phrase-level semantically similar clarifications (fillers) in a given context. We believe the failure is due to the way these models were trained using a masked language modeling (MLM) task in the pre-training phase. MLM aims to map tokens with similar semantics to the embedding space that are close to each other instead of distinguishing them.

Based on the above finding, we believe what we need is a discriminator (Clark et al., 2020; He et al., 2021) pre-trained via a replaced token detection (RTD) task which is more aligned with this shared task. In RTD, the discriminator needs to determine if a corresponding token is either an original token or a token replaced by the generator. Formally, the loss function for the discriminator is as follows:

$$\mathcal{L}_{RTD} = -\sum_i \log p\left(\mathbb{1}\left(\tilde{x}_i = x_i\right) \mid \tilde{\mathbf{X}}, i\right) \quad (1)$$

where $\tilde{\mathbf{X}}$ is the input sequence constructed by replacing masked tokens with plausible tokens sampled from a generator, and the indicator function $\mathbb{1}(\cdot)$ distinguishes whether the plausible tokens are generated or the original ones.

3 Method

In this section, we will describe the strategies we used in our system in detail. In Section. 3.1, the system is presented on how we formalize the data as input, basic modules, and task-specific design for each subtask. Then, we describe the optimization object for each subtask (see Section. 3.2). Finally, we introduce a pattern-aware ensemble strategy to further boost the performance beyond a normal ensembled model in Section. 3.3.

3.1 System Description

As illustrated in Figure. 2, the framework for both tasks is nearly the same which consists of 4 parts, namely the input, a basic model, a pre-trained head, and a task-specific head.

The input sequence. Each sample is constructed by joining the *Pattern*, *Title*, *Section Header*, *Previous Sentence*, *Target Sentence* and *Follow-up Sentence* in order demonstrated in Figure. 1. Each candidate phase is filled in in its original position in the *Target Sentence*. When modeling the filled target sentence independently, the training set will be 5 times larger than the original since each target phrase has five candidates.

Basic Model. The pre-trained transformer is our starting point. The basic model takes as input the sequence $\tilde{x}$ and outputs the contextual representation of each token as follows:

$$\mathbf{H}_b = \text{Transformer}(\tilde{x}) \quad (2)$$

where $\mathbf{H}_b \in \mathbb{R}^{n\times d}$ with n tokens and d dimension.

Pre-trained Head. During the pre-training phase, a language modeling head is appended for a language modeling task. The head is usually discarded in the fine-tuning phase. However, in our experiment, we found better performance can be achieved when reusing the pre-trained head. Formally, the language modeling head takes as the input $\mathbf{H}_b$ and output representations for task-specific head as follows:

$$\mathbf{H}_p = \text{LN}(\text{Act}(\mathbf{H}_b\mathbf{W}_1 + \mathbf{b}_1)) \quad (3)$$

where $\mathbf{W}_1 \in \mathbb{R}^{d\times d}, \mathbf{b}_1 \in \mathbb{R}^d$ is the weight and bias, $\text{Act}(\cdot)$ and $\text{LN}(\cdot)$ are the activation function and the layernorm layer (Ba et al., 2016) respectively.

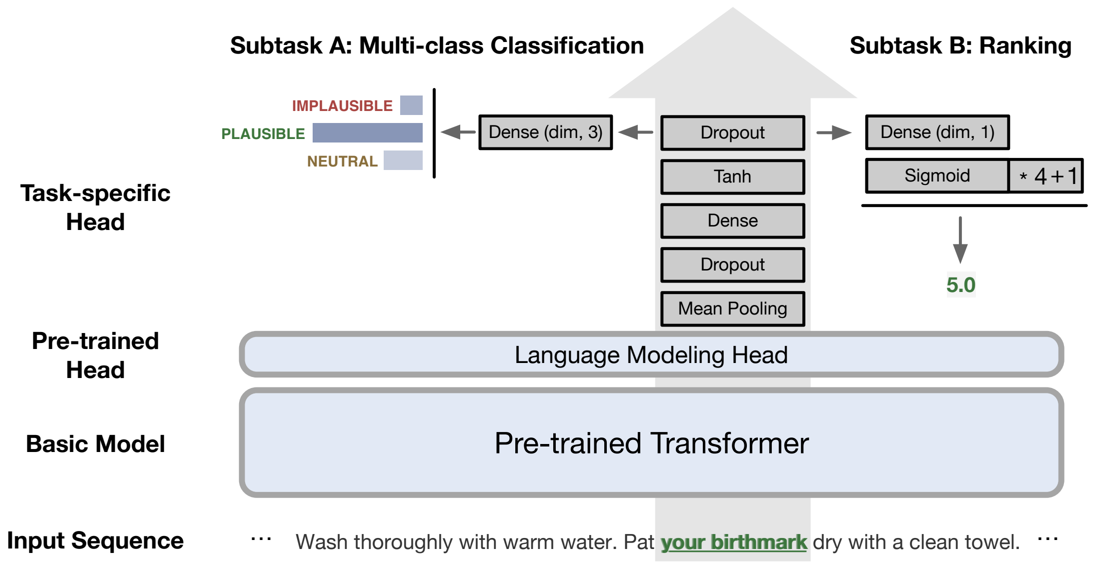


Figure 2: The illustration of our system.

Task-specfic Head. As there are several tokens after tokenizing the target phrase, we apply a mean pooing layer for the hidden states of the target phrase denoted as $\mathbf{H}_{p,i:j}$ as follows:

$$\mathbf{h}_t = \frac{\sum_i^j \mathbf{H}_{p,i}}{j-i} \tag{4}$$

where $\mathbf{h}_t \in \mathbb{R}^{1\times d}$ is the mean embedding of the target phase, i, j are the start and end token index of the tokenized target phrase respectively. Then, we sequentially appended a dropout layer, a dense layer with a Tanh activation function and a dropout layer for $\mathbf{h}_t$ as follows:

$$\tilde{\mathbf{h}}_t = \text{Dropout}(\text{Tanh}(\text{Dropout}(\mathbf{h}_t)\mathbf{W}_2 + \mathbf{b}_2)) \tag{5}$$

where $\tilde{\mathbf{h}}_t \in \mathbb{R}^{1\times d}, \mathbf{W}_2 \in \mathbb{R}^{d\times d}, \mathbf{b}_2 \in \mathbb{R}^d$ are the enhanced embedding of the target phase, learnable weight and bias respectively. Finally, the $\tilde{\mathbf{h}}_t$ is transformed to fit the three-class classification task and regression task as follows:

$$\tilde{\mathbf{y}}_c = \text{Softmax}(\tilde{\mathbf{h}}_t\mathbf{W}_3 + \mathbf{b}_3) \tag{6}$$

$$\tilde{y}_r = \text{Sigmoid}(\tilde{\mathbf{h}}_t\mathbf{W}_4 + \mathbf{b}_4) * 4 + 1 \tag{7}$$

where $\tilde{\mathbf{y}}_c \in \mathbb{R}^{1\times 3}, \mathbf{W}_3 \in \mathbb{R}^{d\times 3}, \mathbf{b}_3 \in \mathbb{R}^3$ are the probabilty distribution, learnable weight and bias for subtask A, and $\tilde{y}_r \in \mathbb{R}^1, \mathbf{W}_3 \in \mathbb{R}^{d\times 1}, \mathbf{b}_3 \in \mathbb{R}^1$ are the regression score, learnable weight and bias for subtask B. The Sigmoid function restricts the range of output space between 0 to 1, then we shift the number by multiplying four and adding one. The above method successfully restrict the regression score within the golden score on a scale of 1 to 5.

3.2 Optimazation Object

For subtask A, we utilized the cross-entropy loss for multi-class classification as follows:

$$\mathcal{L}_{ce} = -\sum_i^N \log(\tilde{\mathbf{y}}_c^i[y_c^i]) \tag{8}$$

where N is the number of training samples, y_c^i is the golden label for i-th sample, $\tilde{\mathbf{y}}_c^i[y_c^i]$ means the predicted probability of the golden label.

For subtask B, we used the mean squared error loss for regression as follows:

$$\mathcal{L}_{reg} = \frac{1}{N}\sum_i^N (\tilde{y}_r^i - y_r^i)^2 \tag{9}$$

3.3 Pattern-aware Ensembling

Ensemble is the commonly used technique where multiple diverse models are trained to predict an outcome, then aggregates the prediction of each model resulting in the final prediction. In our experiment, we observed that the model fine-tuned with different hyper-parameters have different preference on the *Resolved Pattern*[3]. Thus, we aggregates the prediction of each model seperately based on the performance on a subset split by the given *Resolved Pattern* attribute.

[3]Descriptions of the resolved pattern can be found in https://competitions.codalab.org/competitions/35210#participate

Pattern\Dataset	Train	Validation	Test
ADDED COMPOUND	5000	625	625
FUSED HEAD	4995	625	625
IMPLICIT REFERENCE	4980	625	625
METONYMIC REFERENCE	5000	625	625
Total	19975	2500	2500

Table 1: Data statistics in train, validation, and test set on the different patterns.

Hyper-parameter	Model
Dropout	0.1
Warmup Ratio	0.1
Learning Rates	{5e-6, 7e-6, 9e-6, 1e-5}
Batch Size	{32, 48, 64}
Weight Decay	0.01
Epoches	5
Learning Rate Decay	Linear

Table 2: Hyper-parameters for fine-tuning on both subtasks.

4 Experiment

4.1 Data

We use the training, validation and test data provided for SemEval 2022 Task 7 without introducing extra data. The data statistic is summarized in Table. 1 where there is a balanced distribution among different patterns.

4.2 Experimental Setup

DeBERTa (He et al., 2020, 2021), XLMR (Conneau et al., 2019) and ERNIE (Sun et al., 2021) are used as the pre-trained language model. We fine-tune the models using the AdamW optimizer (Kingma and Ba, 2014) with the default hyper-parameter, and additional fine-tuning hyper-parameters are listed in Table. 2. Experiments are carried out using eight Nvidia A100 GPUs.

4.3 Evaluation Method

For subtask A, the evaluation metric is the accuracy score. The model must predict one of the following labels: {IMPLAUSIBLE, NEUTRAL, PLAUSIBLE}.

For subtask B, the submission will be scored using Spearman's rank correlation coefficient, which compares the predicted plausibility ranking over all test samples to the gold ranking.

Task	**SubTask-A**		**SubTask-B**	
	Dev	Test	Dev	Test
2nd Place	-	66.10	-	0.7850
Ensembled Model	71.08	66.50	0.8260	0.7950
+ Pattern-aware	**75.20**	**68.90**	**0.8441**	**0.8070**

Table 3: Performance of models on dev set and official test set.

#	**Models**	**SubTask-A**
	MLM-based Models	
1	XLMR-Large	61.14
2	ERNIE	61.73
	RTD-based Models	
3	DeBERTa-V3-Large	**67.25**
4	#3 without pre-trained head	65.96

Table 4: Ablation studies on SubTask A with respect to the accuracy score on the dev set. (We reported the mean results with at least three runs.)

4.4 Results

Our ensembled prediction on test set placed first in the competition, with a 68.9% accuracy score for subtask A and a 0.8070 Spearman's rank correlation coefficient for subtask B. As shown in Table. 3. Our system outperforms the second-place system by 2.8 and 2.2 percent points respectively. The organizers predict an upper bound of 77.1% accuracy score and 0.89 ranking correlation based on the manual annotations. As a result, there's still a lot of room for growth.

4.5 Ablation Studies

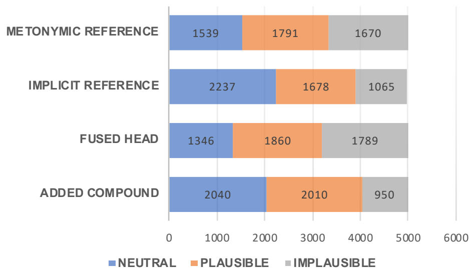


Figure 3: The label distribution of different pattern on training dataset.

The effectiveness of using a replaced token detection task pre-trained model and recovering the pre-train language modeling head in the task-specific

Pattern	IMPLICIT REFERENCE		METONYMIC REFERENCE		FUSED HEAD		ADDED COMPOUND	
Hyper-paramters\Task	SubTask-A	SubTask-B	SubTask-A	SubTask-B	SubTask-A	SubTask-B	SubTask-A	SubTask-B
LR:1e-5, BSZ:32	65.12	0.8321	67.36	**0.8427**	67.68	0.8400	64.64	**0.8272**
LR:9e-6, BSZ:32	64.96	**0.8340**	**69.60**	0.8408	**71.84**	**0.8418**	63.20	0.8251
LR:1e-5, BSZ:64	71.84	0.8286	69.28	0.8382	65.12	0.8347	68.96	0.8134
LR:9e-6, BSZ:64	**72.32**	0.8265	**69.60**	0.8424	64.96	0.8325	**69.28**	0.8142

Table 5: Performance of the DeBERTa-V3-Large with different fine-tuning hyperparameters on the dev set. A model can't win all the subtasks on a subset split by the given pattern attribute. (LR and BSZ are abbreviations for learning rate and batch size.)

head have been revealed in Table. 4. The hypothesis that utilizing a model pre-trained by a similar task aligned with SemEval-2022 Task 7 contributes a lot is supported by comparing #1,#2 and #3. The performance of the model improved even more after reusing the pre-trained LM head (#3 and #4). The assumption is that the hidden states from the pre-trained head contain more information learned during the pre-training phase for distinguishing semantically similar tokens.

The effectiveness of the pattern-aware ensembling has been shown in Table.3. On subtasks A and B, pattern-aware ensembling outperformed the standard ensemble technique by 2.4 and 1.2 percent points, respectively, compared to the standard ensemble method.

The model trained with different hyperparameters may perform better on one pattern but not on another, as seen in Table. 5. For example, on the `FUSED HEAD` pattern, the model (LR:9e-6, BSZ:32) has the highest accuracy score of 71.84% but the lowest accuracy scores of 64.96% and 63.20% on `ADDED COMPOUND` and `IMPLICIT REFERENCE` pattern, respectively. The model (LR:9e-6, BSZ:64), on the other hand, has the lowest score on `FUSED HEAD` pattern but the best result on `ADDED COMPOUND` and `IMPLICIT REFERENCE` pattern. By visualizing the label distribution in Figure. 3, we infer that the phenomenon is related to a distribution difference in which `FUSED HEAD` pattern contains the lowest number of the label *NEUTRAL*, and the label `PLAUSIBLE` dominates the `ADDED COMPOUND` and `IMPLICIT REFERENCE` patterns.

5 Conclusion

We built a system for identifying plausible clarifications of implicit and underspecified phrases in instructional texts which is useful for improving the clarification of instructional texts. The system leverages the strength of a replaced token detection pre-trained discriminator and therefore performs extremely well on this shared task with the same goal to distinguish semantically similar tokens. In particular, we proposed a pattern-aware ensembling strategy to aggregate multiple predictions separately based on the pattern when there is a label distribution difference among patterns. On SemEval-2022 Task 7, the system achieved the best performance in both subtasks.

In future work, it's promising to incorporate the replace token detection task in a large-scale pre-trained model with billion, or even trillion parameters.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

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

PALI at SemEval-2022 Task 7: Identifying Plausible Clarifications of Implicit and Underspecified Phrases in Instructional Texts

Mengyuan Zhou and **Dou Hu** and **Mengfei Yuan** and **Meizhi Jin** and **Xiyang Du** and **Lianxin Jiang** and **Yang Mo** and **Xiaofeng Shi**
Ping An Life Insurance Company of China, Ltd.
{ZHOUMENGYUAN425, HUDOU470, YUANMENGFEI854, JINMEIZHI005, DUXIYANG037, JIANGLIANXIN769, MOYANG853, SHIXIAOFENG309}
@pingan.com.cn

Abstract

This paper describes our system used in the SemEval-2022 Task 7(Roth et al.): Identifying Plausible Clarifications of Implicit and Underspecified Phrases. Semeval Task7 is an more complex cloze task, different than normal cloze task, only requiring NLP system could find the best fillers for sentence. In Semeval Task7, NLP system not only need to choose the best fillers for each input instance, but also evaluate the quality of all possible fillers and give them a relative score according to context semantic information. We propose an ensemble of different state-of-the-art transformer-based language models(i.e., RoBERTa and Deberta) with some plug-and-play tricks, such as Grouped Layerwise Learning Rate Decay (GLLRD) strategy, contrastive learning loss, different pooling head and an external input data prepreccess block before the information came into pretrained language models, which improve performance significantly. The main contributions of our system are 1) revealing the performance discrepancy of different transformer-based pretraining models on the downstream task; 2) presenting an efficient learning-rate and parameter attenuation strategy when fintuning pretrained language models; 3) adding different constrative learning loss to improve model performance; 4) showing the useful of the different pooling head structure. Our system achieves a test accuracy of 0.654 on subtask1(ranking 4th on the leaderboard) and a test Spearman's rank correlation coefficient of 0.785 on subtask2(ranking 2nd on the leaderboard).

1 Introduction

Cloze tasks have become a standard framework for evaluating various discourse-level phenomena in NLP. Some prominent examples include the narrative cloze test(Hoshino and Nakagawa, 2007), the story cloze test (Xie et al., 2020), and the LAMBADA word prediction task(Paperno et al., 2016). In these tasks, NLP systems are required to make a prediction about the filler of a cloze that is most likely to continue the discourse. However, these existing cloze tasks focus on the accuray of choosen fillers, ignore evaluating the absolute quality of all possible predictions.

The goal of Semeval 2022 Task7 is to evaluate the ability of NLP systems to distinguish between plausible and implausible clarifications of an instruction. The task is formulated as a complex cloze task, which involve two sub tasks. In Sub task 1, for each(sentence, filler) pair, NLP system need to classify four fillers into plausible, implausible or neutral and the evaluating indicator is accuracy. In Sub task 2, for each pair, NLP system need to predict scores for five fillers and the evaluating indicator is Spearman's rank correlation coefficient.

Since 2018, NLP models have adopted the concept of pre-training on a diverse corpus of unlabelled text, followed by supervised finetuning on specific tasks. Pretrained models are built to simulate anthropomorphic learning, wherein existing knowledge can be adapted to new tasks without doing any training on these tasks from scratch - a requirement of traditional machine learning models. By now, these pre-trained large language models such as Bert(Devlin et al., 2018), Roberta(Liu et al., 2019),Xlm-roberta (Conneau et al., 2019), Deberta(He et al., 2021) has been widely used to solve all kinds of language understanding tasks. Additionally, fine-tuning self-supervised pre-trained models has significantly boosted state-of-the-art performance on natural language processing (NLP) tasks. Many evidence showing models with pretrained commonsense knowledge can be well applied in the field of cloze task(Cui et al., 2020), because cloze task needs commonsense language knowledge and genera language knowledge.

Additionally, there are many training tricks can be used to improve the performance and generalization ability of the large pre-trained language models. First, adding contrastive learning loss in supervised task, such as Ntxent loss(Chen et al., 2020). Second, in case of pretrained language models' catastrophic forgetting in funtuning period, we set different learning rate and weight decay rate for different pre-trained language model layers(Zhang et al., 2021). Third in order to get the best sentence embedding, trying different pooling head is necessary. Inspired by these discoveries, we designed two NLP system for sub-task1 and sub-task2.

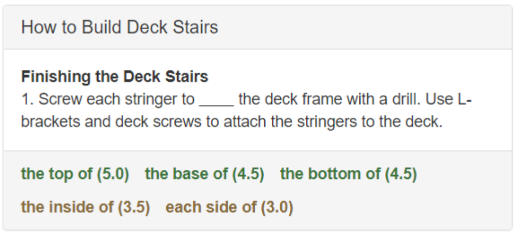
2 Task Setup

Formally, each instance in dataset is composed of 5 sentences, 5 fillers, 1 clarification phenomena and 1 score:

- $Article\ title$: title of the wikiHow article in which the sentence occurs.
- $Section\ header$: heading of the section which the sentence is part of.
- $Previous\ context$: a couple of sentences that occur before the sentence in question - omissions are marked by "(...)".
- $Sentence$: the sentence with a placeholder "..." that marks where the fillers should be inserted.
- $Follow-up\ context$: a couple of sentences that occur after the sentence in question - omissions are marked by "(...)".
- $Filler1 - Filler5$: the five different fillers.
- $Score$: is the quality score of each candidate word, range from 1 to 5.
- $Resolved\ pattern$: name of the clarification phenomenon (cf. list above: implicit reference, fused head, added noun, metonymic reference.)

As showing in figure 1, this task is a cloze task, using fillers(Filler1 to Filler5) to insert a blank at the position in the text(e.g. Screw each stringer to ___ the deck frame with a drill, use L-brackets and deck screws to attach the stringers to the deck.). Additional, article title(), section header,previous and fellow-up sentence, resolved pattern are given.

Sub-task 1 is a classification sub-task, the evaluation metrics is overall accuracy. According to rules, converting each real-valued gold score to a class label as follows:


How to Build Deck Stairs

Finishing the Deck Stairs
1. Screw each stringer to ____ the deck frame with a drill. Use L-brackets and deck screws to attach the stringers to the deck.

the top of (5.0) the base of (4.5) the bottom of (4.5)
the inside of (3.5) each side of (3.0)

Figure 1: Data Instance.

- $score <= 2.5$: IMPLAUSIBLE
- $2.5 < score < 4$: NEUTRAL
- $score >= 4$: PLAUSIBLE

Subtask2 is a regression subtask, our system need to predict each instance's gold plausibility score. The submissions will be scored based on Spearman's rank correlation coefficient which compares the predicted plausibility ranking over all test instances with the gold ranking.

3 Data Summary and Analysis

3.1 Data Construction

We try two different methods to preprocess input data.

Strategy One: We simply concatenate resolved pattern, article title, section header, previous context, sentence and follow-up context first. And then fill the blank spaces with 5 fillers separately. Finally in order to highlight fillers information, we add special symbols "<e>" before and after fillers.

Strategy Two: Since Resolved pattern is kind of category feature which is different than the other text features, We first replace resolved pattern with their explanations. And then connecting explanations to the other information. Finally, adding special symbols "<e>" before and after fillers. The explanation of resolved pattern showing below:

MPLICIT REFERENCE: In the original version of a sentence, there is an implicit reference to a previously mentioned entity. The revision makes this reference explicit.

FUSED HEAD: In the original version, there is a noun phrase where the head noun is missing. The revision adds that noun.

ADDED NOUN: The revision adds a compound modifier to a noun to make its meaning more specific.

METONYMIC REFERENCE: In the original version, a noun is used in a metonymy. The revision

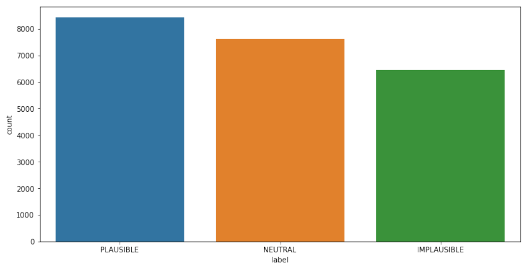


Figure 2: Label Distribution.

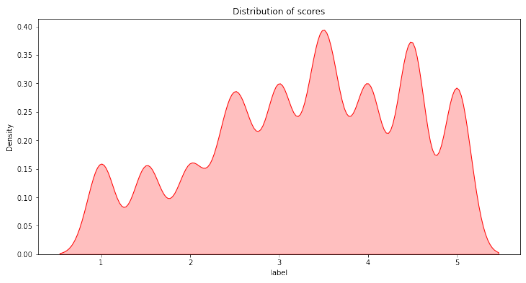


Figure 3: Score Distribution.

makes the particular component or aspect of a noun explicit that is meant.

We adopt two kinds of preprocess methods in our experiment and Strategy Two showing a better performance.

3.2 Data Analysis

We can see the label distribution and score distribution in Figure 2 and Figure 3. The largest number of labels is a plausible and the average score is near 3.5.

4 Methodology

For this task, we have tried a variety of modeling, optimization methods, learning rate schedule and different constrative learning loss. Details are described below.

4.1 Model Design

We design same model architectures for this two sub-task. Our model is based on different pre-trained models, such as roberta, xlmroberta and deberta. After these pre-trained block, we set different pooling head to replace cls head in order to get better sentence information. We try 3 different head, mean-max pooling head, cls head and lstm + attention head, and all the pooling structures showing good performance in our experiments.

CLS Pooling Head: CLS Pooling Head is the last layer hidden-state of the first token of the sequence (classification token) further processed by a Linear layer and a Tanh activation function. The Linear layer weights are trained from the next sentence prediction (classification) objective during pretraining. We reset cls top linear layer weights in finetuning. We believe this weights are over fitting NSP task that have a bad effect on further finetuning.

Mean-Max Pooling Head: We consider the last hidden state [batch, maxlen, hidden_state], then take max across maxlen dimensions to get max pooling embeddings. For mean pooling, we also consider the last hidden state, the average across max length dimensions to get averaged/mean embedding. Finally we concatenate this two embedding and further processed by a Linear layer.

LSTM plus Attention Pooling Head: Since LSTM network is inherently suitable for processing sequential information, we can use a LSTM network to connect all token of last hidden state [batch, maxlen, hidden_state], and the output of the all LSTM cell [batch, nums_LSTM_cell, LSTM_hidden_state] is used as input of next dot-product attention module. After dot-product attention module, we pass the pooled output to a fully connected layer for label prediction.

4.2 Training Details

Our system adopt grouped layer-wise learning rate decay(GLLRD)(Ginsburg et al., 2018) as main learning rate and weight decay strategy. GLLRD is a method that applies higher learning rates for top layers and lower learning rates for bottom layers. This is accomplished by setting the learning rate of the top layer and using a multiplicative decay rate to decrease the learning rate layer-by-layer from top to bottom.

In our experiment, We set 3 parameter group for 24-layers pretrained large language models, first group include 0 to 7 pretrained layers; second group include 8 to 15 pretrained layers; third group include 16-23 pretrained layers. We design a base learning rate 1e-5 for group 2(8-15 pretrained layers); A lower learning rate 1e-5/1.6 for group 1; A higher learning rate 1e-5 * 1.6 for group3; And a much higher learning rate(2e-4) for top layers.

The goal is to modify the lower layers that encode more general information less than the top layers that are more specific to the pre-training task. This method is adopted in fine-tuning several recent pre-trained models, including Roberta, Xlm-roberta and Deberta-v3(Zhang et al., 2021). Additionally, we adopt cosine_warmup in our learning rate scheduler and we adopt AdamW with opening bias correction. For task7 dataset, if not open bias correction, will lead to huge fluctuations in model performance.

4.3 Loss Function Design

Our system involve 3 kinds of loss, Classification loss(CrossEntropy loss), regression loss(MSE loss) and contrastive loss(NTXent loss). Inspired by recent contrastive learning algorithms, our system adopt NTXent loss as sencond loss which is proposed in SimCLR. Contrastive loss learns representations to maximize agreement between differently augmented views of the same data example in the latent space.In this task, our system need to evaluate the quality of all possible fillers, NTXent loss can help system get more robust sentence representation to classify all fillers.

For NTXent loss, We randomly sample a minibatch of N examples and define the contrastive prediction task on pairs of augmented examples derived from the minibatch, resulting in 2N data points.We do not sample negative examples explicitly. Instead, given a positive pair, similar to (Chen et al., 2017), we treat the other 2(N − 1) augmented examples within a minibatch as negative examples. Let $sim(u, v) = \mathbf{u}^\top v/||u||||v||$ denote the dot product between ℓ_2 normalized u and v (i.e.cosine similarity). Then the loss function for a positive pair of examples (i, j) is defined as (Chen et al., 2020):

$$\ell_{i,j} = -\log \frac{\exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_k\right)/\tau\right)} \tag{1}$$

where $\mathbb{1}_{[k\neq i]} \in (0, 1)$ is an indicator function evaluating to 1 iff $[k \neq i]$ and τ denotes a temperature parameter. The final loss is computed across all positive pairs, both (i, j) and (j, i), in a minibatch.

In subtask 1, we adopt the weighted average method to obtain the final loss between CrossEntropy loss and NTXent loss. The method is as follows:

$$Loss_{total} = Loss_{CE} + \alpha Loss_{NTX}, \tag{2}$$

where $Loss_{CE}$ is CrossEntropy loss,$Loss_{NTX}$ is NTXent loss, $\alpha = 0.1$

In subtask 2, we adopt the final weighted average method to obtain the final loss between MSE loss and NTXent loss. The method is as follows:

$$Loss_{total} = Loss_{MSE} + \alpha Loss_{NTX} \tag{3}$$

where $Loss_{MSE}$ is MSE loss,$Loss_{NTX}$ is NTXent loss, $\alpha = 0.1$.

5 Experiments

5.1 Experiment Settings

In order to get better performance in this few-sample dataset, We apply AdamW as an optimization algorithm with 10% steps of warmup and open the the correct_bias item (Zhang et al., 2021). For hyperparamete, we fine-tune the uncased, 24-layer $Roberta_{Large}$, $Xlm - Roberta_{Large}$ and $Deberta_{Large}$ model with batch size 40, dropout 0.2, cosine_warmup 1e-2. Additionally we adopt grouped layer-wise learning rate decay strategy with base learning rate 1e-5, weight-ratio 1.6 and a much higher learning 2e-4 for top pooling layers, mentioned in 4.2 Training Details. We used stratified k-fold method to split training data into 5 folds.

5.2 Experimental Results

We separate trained our system for sub-task1 and sub-task2. In both sub-task, we adopt sentence embedding to settle the further classification and regression works. As showing in Table 1, for each method, the score we report here is the average score of the experiment results. From Table 1, we see that the deberta-v3 model showing the best overall performance on both sub-task1 and sub-task2. In sub-task1, we can find deberta model is at least 2% higher than roberta model and 1.8% higher than xlm-roberta model. On sub-task2, deberta model's improvement is much higher, compared with roberta and xlm-robera, deberta has an improvement of more than 4.5% and 3.1% respectively. More important, data construction method 2 replacing resolved pattern with their explanations also provided a performance boost, around 0.7% in both sub-task. GLLRD strategy and contrastive loss bring a great improvement, neary 1% and 0.5%. Different pooling head also bring different influence in final score, Lstm + Attention head showing the best performance, which can reach 0.649 in sub-task1 and 0.782 in sub-task2 . Totally, after trying different method and model fusion, our system

Pretrained model	Data Construction Method	Pooling head	GLLRD	Contrastive loss	ACC @ subtask1	Spearman coefficient @ subtask2
roberta-large	Strategy One	CLS Pooling	False	—	0.602	0.696
	Strategy Two	CLS Pooling	False	—	0.608	0.704
	Strategy Two	CLS Pooling	True	—	0.619	0.714
	Strategy Two	CLS Pooling	True	NTXent	0.624	0.718
	Strategy Two	Mean-Max Pooling	True	NTXent	0.626	0.717
	Strategy Two	LSTM plus Attention Pooling	True	NTXent	0.628	0.725
xlm-roberta-large	Strategy Two	CLS Pooling	True	NTXent	0.626	0.738
	Strategy Two	Mean-Max Pooling	True	NTXent	0.625	0.736
	Strategy Two	LSTM plus Attention Pooling	True	NTXent	0.629	0.740
deberta-v3-large	Strategy Two	CLS Pooling	True	NTXent	0.647	0.771
	Strategy Two	Mean-Max Pooling	True	NTXent	0.645	0.773
	Strategy Two	LSTM plus Attention Pooling	True	NTXent	0.649	0.782
Multi model Fusion	—	—	—	—	0.654	0.785

Table 1: Experiment results for sub-task1 and sub-task2

reach a test accuracy of 0.654 on sub-task1 and a test Spear-man coefficient of 0.785 on sub-task2.

6 Conclusion

This paper propose a complex system with GLLRD strategy, contrastive loss, input data construction block, different pretrained models and different pooling head structure. It solves the problem of how to evaluate the quality of all possible fillers in cloze task. Experiments on SemEval task 7 datasets demonstrate that using our system can advance the normal cloze task models.

Acknowledgements

This research was supported by the PingAn Life Insurance. All the work stated in this paper was

conducted during the Semeval-2022 competition.

References

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 767–776. ACM.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Leyang Cui, Sijie Cheng, Yu Wu, and Yue Zhang. 2020. Does BERT solve commonsense task via commonsense knowledge? *CoRR*, abs/2008.03945.

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

B. Ginsburg, I. Gitman, and Y. You. 2018. Large batch training of convolutional networks with layer-wise adaptive rate scaling.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Ayako Hoshino and Hiroshi Nakagawa. 2007. A cloze test authoring system and its automation. In *Advances in Web Based Learning - ICWL 2007, 6th International Conference, Edinburgh, UK, August 15-17, 2007, Revised Papers*, volume 4823 of *Lecture Notes in Computer Science*, pages 252–263. Springer.

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

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Michael Roth, Talita Anthonio, and Anna Sauer. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Yuqiang Xie, Yue Hu, Luxi Xing, Chunhui Wang, Yong Hu, Xiangpeng Wei, and Yajing Sun. 2020. Enhancing pre-trained language models by self-supervised learning for story cloze test. In *Knowledge Science, Engineering and Management - 13th International Conference, KSEM 2020, Hangzhou, China, August 28-30, 2020, Proceedings, Part I*, volume 12274 of *Lecture Notes in Computer Science*, pages 271–279. Springer.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

niksss at SemEval-2022 Task 7: Transformers for Grading the Clarifications on Instructional Texts

Nikhil Singh
Manipal University Jaipur
`nikhil3198@gmail.com`

Abstract

This paper describes the 9th place system description for SemEval-2022 Task 7. The goal of this shared task was to develop computational models to predict how plausible a clarification made on an instructional text is. This shared task was divided into two Subtasks A and B. We attempted to solve these using various transformers-based architecture under different regime. We initially treated this as a text2text generation problem but comparing it with our recent approach we dropped it and treated this as a text-sequence classification and regression depending on the Subtask.

1 Introduction

Instructional texts which are in the form of step-by-step instruction to achieve a particular goal are sometimes ambiguous to make out what is being talked about. To ensure that instructions describe clearly enough what steps must be followed, some clarifications are made in the places of ambiguity. This task (Roth et al., 2022) revolves around automating the grading of a particular clarification made on the instructions into plausible implausible and neutral (SubTask A) and on a finer scale where it is required to rank potential clarifications from 1 to 5 (SubTask B).

Previous work, related to this involves a Shared Task (Roth and Anthonio, 2021) which was a binary classification task, in which systems had to predict whether a given sentence in context requires clarification or not. This shared task uses the same dataset that is the wikiHowToImprove dataset (Anthonio et al., 2020) but with some variations. Instead of a binary classification task, this task is shaped as a cloze task in which, clarifications are presented as possible fillers and systems have to score how well each filler plausibly fits in a given context. A data instance can be seen in Figure 1.

Seeing the performance of BERT over BiLSTMs in (Bhat et al., 2020), we decided to build upon that and explore the use of other transformers-based models.

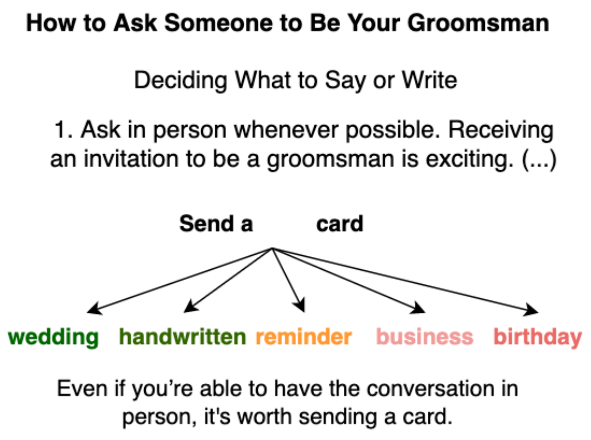

How to Ask Someone to Be Your Groomsman
Deciding What to Say or Write
1. Ask in person whenever possible. Receiving an invitation to be a groomsman is exciting. (...)
Send a card
wedding
handwritten
reminder
business
birthday
Even if you're able to have the conversation in person, it's worth sending a card.

Figure 1: A data instance of Task 7

2 System Overview

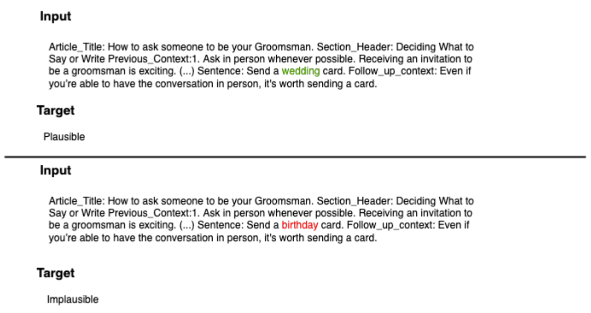
Data Pre-Processing

In order to convert the provided data into the form required for a text sequence classification model, we filled the blanks in the provided text with the provided potential fillers and associated their respective labels as shown in Figure 2. The resulting data were then divided into three parts training(80 %), validating(10 %), and testing(10 %).

2.1 Subtask A

For this task, we mainly experiment with two transformers-based models that differ fundamentally in the manner they were trained. Initially, we treated this as a text to text generation task using T5 (Raffel et al., 2019). The detailed steps involved in this experiment is present below.

- The Article title, previous context, the sentence (with filler), and the follow-up context was sequentially laid out one after the other.


Input

Article_Title: How to ask someone to be your Groomsman. Section_Header: Deciding What to Say or Write Previous_Context:1. Ask in person whenever possible. Receiving an invitation to be a groomsman is exciting. (...) Sentence: Send a wedding card. Follow_up_context: Even if you're able to have the conversation in person, it's worth sending a card.

Target

Plausible

Input

Article_Title: How to ask someone to be your Groomsman. Section_Header: Deciding What to Say or Write Previous_Context:1. Ask in person whenever possible. Receiving an invitation to be a groomsman is exciting. (...) Sentence: Send a birthday card. Follow_up_context: Even if you're able to have the conversation in person, it's worth sending a card.

Target

Implausible

Figure 2: Data input for both T5 and BERT

- Keywords such as Article Title,Section Header,Previous Context,Sentence and Follow up context were included in the input string to indicate the respective content for the T5 model.
- The model was trained in a supervised manner using Cross-Entropy for 2 Epochs with a batch size of 2 and gradient accumulation steps of 8 making an effective batch size of 16. The rest of the Hyper-parameters were as follows:max seq length=512,learning rate=3e-4,adam epsilon=1e-8 and a seed value of 42 to keep the model deterministic.
- The model took approximately 1.5 hours to train on Nvidia's P100 GPU with a memory of 16Gb.
- The complete experiment was done on Google Colab Pro.
- The model architecture can be seen in Figure 3.

The second model used was a BERT (Devlin et al., 2018) based model.

The detailed steps involved in this experiment is present below.

- The pre-processing of the data was the same as T5 but with all the instruction constituents clubbed into a single text, i.e. all the keywords such as Article Title,Section Header,Previous Context,Sentence and Follow up context were dropped and the resulting sequence was a continuous text sequence.
- The resulting input sequence was tokenized using a BertTokenizer from Huggingface and is passed through the bert-base-uncased model to embed it into a 768 dimensional feature vector containing the syntactical information of the input string.
- The feature vector is then passed through a dropout layer to increase the regularization which in-turn increases the generalizability of the model.
- The model was trained in a supervised manner in a multiclass classification regime for 5 Epochs with a batch size of 32. Rest of the Hyper-parameters are shown in Table 2. A seed value of 42 to keep the model deterministic.
- The model took approximately 25 minutes to train on Nvidia's P100 GPU with a memory of 16Gb.
- The complete experiment was done on Google Colab Pro.
- The model architecture can be seen in Figure 4

Model	**Accuracy(%)**
T5-base	40.28
bert-base-uncased	44.40

Table 1: Result on the hold-out test set for Subtask A

When compared to the T5 model the BERT-based model was not just effective but also more efficient and took lower time for training and inference and therefore became our official submission for this subtask. See Table 1 for results on the hold-out test set.

The Experiment setup has been shown in Table 2. All the Experiments were performed using Huggingface Transformers Library. [1]

2.2 Subtask B

Seeing the success of BERT over T5, we make necessary changes to convert the model used for Subtask A into a regression model. See Figure 5. See Table 3 for results on the hold-out test set. The experiment setup can be seen in Table 4.

3 Results

3.1 Subtask A

The submitted systems were evaluated using the Accuracy metric(Sklearn footnote). We were officially ranked 7th in Subtask A with an accuracy

[1]https://huggingface.co/

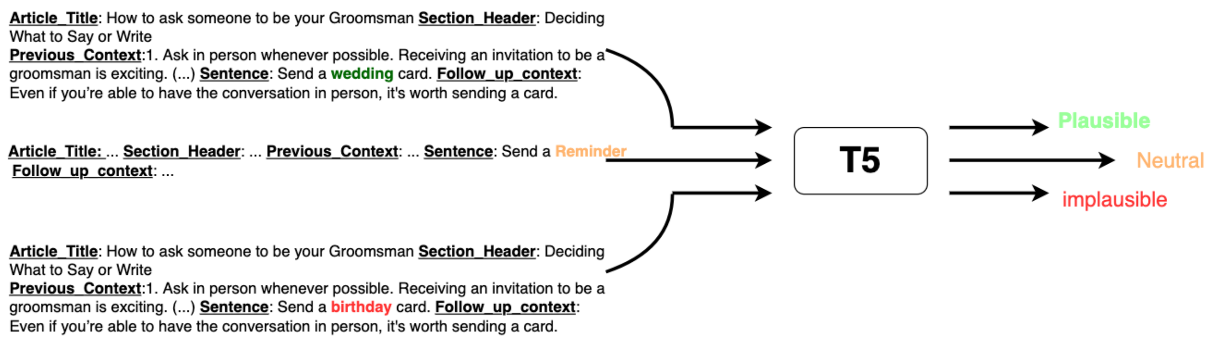

Article_Title: How to ask someone to be your Groomsman Section_Header: Deciding What to Say or Write
Previous_Context:1. Ask in person whenever possible. Receiving an invitation to be a groomsman is exciting. (...) Sentence: Send a wedding card. Follow_up_context: Even if you're able to have the conversation in person, it's worth sending a card.
Article_Title: ... Section_Header: ... Previous_Context: ... Sentence: Send a Reminder Follow_up_context: ...
Article_Title: How to ask someone to be your Groomsman Section_Header: Deciding What to Say or Write
Previous_Context:1. Ask in person whenever possible. Receiving an invitation to be a groomsman is exciting. (...) Sentence: Send a birthday card. Follow_up_context: Even if you're able to have the conversation in person, it's worth sending a card.
T5
Plausible
Neutral
implausible

Figure 3: Working of T5 Model

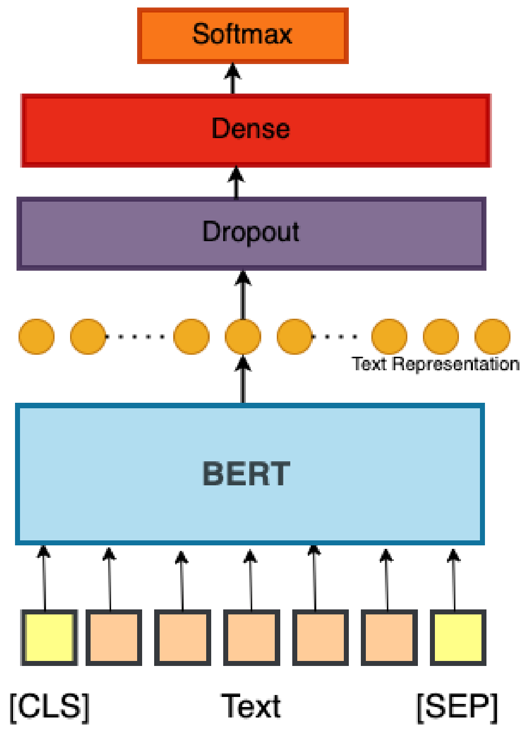


Figure 4: Model Architecture for Subtask A

Model	MSE
bert-base-uncased	44.40

Table 3: Result on the hold-out test set for Subtask B

Parameter	Value
Model	bert-base-uncased
Max sequence Length	256
Batch Size	8
Learning rate	2e-5
Weight decay	Linear
Momentum	0.9
Optimizer	AdamW [2]
Epochs	5
Loss	Cross Entropy

Table 2: Experimental Setup for Subtask A

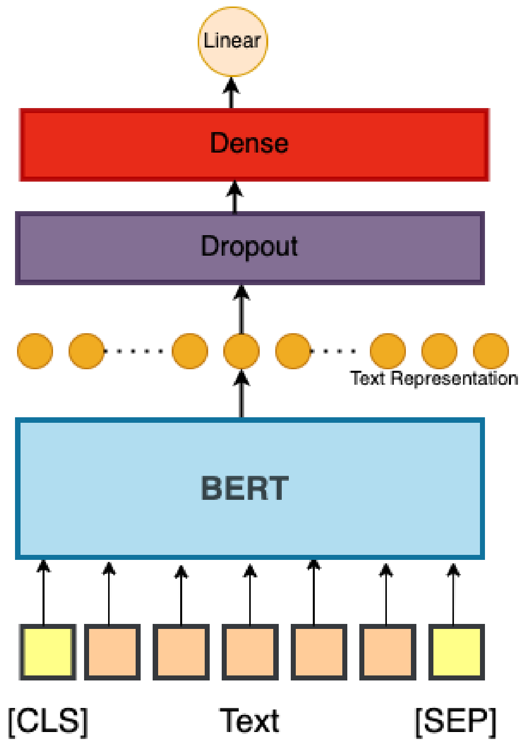


Figure 5: Model Architecture for Subtask B

Parameter	Value
Model	bert-base-uncased
Max sequence Length	256
Batch Size	8
Learning rate	2e-5
Weight decay	Linear
Momentum	0.9
Optimizer	AdamW
Epochs	5
Loss	Mean Squared Error

Table 4: Experimental Setup for Subtask B

score of 44.200% which is substantially above a naive majority class baseline of 39% and comparable to the baseline presented by the task organizers.

3.2 Subtask B

For Subtask B, the submissions were evaluated using Spearman's rank correlation coefficient(SRCC) which compares the predicted plausibility ranking overall test instances with the gold ranking. We ranked 5th with an SRCC of 0.25200.

4 Error Analysis

After examining the predictions from the submitted model, we saw that the model struggled significantly in distinguishing between neutral and either of plausible/implausible clarifications as there's a very slight difference between them. This problem increases further when we created a feature vector from the same sentence with changed filler word as it leads to a very slight change in the vector, hence leading the model confused to distinguish between individual classes having almost similar feature distribution. The performance can also be attributed to the distribution of the labels in development set and our submitted model might have overfitted to the development set, leading to a further decrease in performance in the test set.

5 Conclusion

We developed a system to classify the clarification made on instructional text into varying levels of plausibility using a transformer based language model with a limited attention span only taking a limited context around the filler. The recent advancement in transformer based models, such as BigBird (Zaheer et al., 2020) and Longformer (Beltagy et al., 2020) which can take up longer context into consideration are more preferred for a task like this.

References

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020. wikiHowToImprove: A resource and analyses on edits in instructional texts. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5721–5729, Marseille, France. European Language Resources Association.

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

Irshad Bhat, Talita Anthonio, and Michael Roth. 2020. Towards modeling revision requirements in wikiHow instructions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8407–8414, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Michael Roth and Talita Anthonio. 2021. UnImplicit shared task report: Detecting clarification requirements in instructional text. In *Proceedings of the 1st Workshop on Understanding Implicit and Underspecified Language*, pages 28–32, Online. Association for Computational Linguistics.

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 Task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297.

SemEval-2022 Task 8: Multilingual news article similarity

Xi Chen
U. Mass. Amherst
xchen4@umass.edu

Ali Zeynali
U. Mass. Amherst
azeynali@umass.edu

Chico Q. Camargo
University of Exeter
f.camargo@exeter.ac.uk

Fabian Flöck
GESIS
f.floeck@gmail.com

Devin Gaffney
Meedan
devin@meedan.com

Przemyslaw A. Grabowicz
U. Mass. Amherst
grabowicz@cs.umass.edu

Scott A. Hale
University of Oxford and Meedan
scott.hale@oii.ox.ac.uk

David Jurgens
University of Michigan
jurgens@umich.edu

Mattia Samory
GESIS
mattia.samory@gesis.org

Abstract

Thousands of new news articles appear daily in outlets in different languages. Understanding which articles refer to the same story can not only improve applications like news aggregation but enable cross-linguistic analysis of media consumption and attention. However, assessing the similarity of stories in news articles is challenging due to the different dimensions in which a story might vary, e.g., two articles may have substantial textual overlap but describe similar events that happened years apart. To address this challenge, we introduce a new dataset of nearly 10,000 news article pairs spanning 18 language combinations annotated for seven dimensions of similarity as SemEval 2022 Task 8. Here, we present an overview of the task, the best performing submissions, and the frontiers and challenges for measuring multilingual news article similarity. While the participants of this SemEval task contributed very strong models, achieving up to 0.818 correlation with gold standard labels across languages, human annotators are capable of reaching higher correlations, suggesting space for further progress.

1 Introduction

Consider the following question: Given a pair of "hard" news articles,[1] are they covering the same news story? Answering this question likely requires knowing specific aspects of the events covered: what happened, where and when it happened, who was involved, and why and how it happened (Pan and Kosicki, 1993; Klein and Martínez, 2009; Dijk, 1988).

Effectively modeling the similarity of news stories holds substantial practical benefits in structuring the content of the hundreds of thousands of news articles generated every day.[2] Given the volume of articles, an effective measure of news story similarity enables clustering and identification of event coverage in news media (Rupnik et al., 2016; Bisandu et al., 2018). Commercial news aggregation services, as provided by, e.g., Google News or Apple News, perform a similar clustering approach, yet are primarily monolingual and have not been made openly available or extensively researched beyond proprietary solutions. In addition, quantifying news article similarity allows the comparison of news outlets in terms of their coverage, understanding which stories consume much of the media agenda, as well as tracking the diffusion of news stories through a media ecosystem and over time. Being able to measure these aspects is important for a host of research questions in media and communication studies including, for example, agenda setting (McCombs, 2005). Another highly desirable property of such methods is to be applicable in multilingual settings, to detect news stories covered across languages in an increasingly globalized news ecosystem (Rupnik et al., 2016).

Assessing the similarity of two news articles introduces new challenges not found in traditional semantic textual similarity. Most importantly, methods for semantic textual similarity typically measure the extent to which two arbitrary documents are "the same," without concretely specifying the meaning of this similarity, or only do so in broad strokes (cf. Agirre et al., 2012; Lee et al., 2005; Nguyen et al., 2014). One byproduct of this vague application domain and under-specification is that

[1]"Hard news" is characterized as having a high level of newsworthiness demanding immediate publication (Tuchman, 1972). In our use, we aim to exclude opinion, features, and other forms of journalistic pieces not mainly concerned with covering current events as in Flaxman et al. (2016).

[2]For example, the source we use for metadata, Media Cloud, collects 629K articles per day.

agreeing on gold labels is notoriously difficult, at least at the full-length document level (Nguyen et al., 2014; Westerman et al., 2010), as even human labelers are dependent on specific instructions and/or knowledge of the absolute space of documents to label, to understand the relative concept of "similar" (Bär et al., 2011). News article similarity is thus more related to attempts to compare narratives (Chambers and Jurafsky, 2009; Miller et al., 2015; Chaturvedi et al., 2018), which require understanding the structure and content to assess similarity.

Here, we introduce SemEval 2022 Task 8 for the task of quantifying news similarity, a hard discourse-level task. Stories often include a variety of descriptions, people, and entities that may appear in another, dissimilar story. Further, the temporal nature of news means that as real-life events evolve, stories describing the same event may include new details or entities—possibly becoming a new news story altogether. For this task, we create a high-quality dataset by annotating pairs of news article for similarity in 10 different languages on several dimensions, e.g, geographic, temporal, and narrative similarity. Participants in this task were given a large collection of news articles, with 4,918 pairs receiving ground-truth similarity labels, and were asked to estimate the overall similarity of 4,902 news article pairs given to participants without labels for any dimension. The task is also challenging due to its large language diversity: the training data consists of 8 language combinations, while the evaluation dataset has 18 language combinations including three languages not appearing at all in the training data.

2 Data

2.1 Data Collection

The metadata and full text of news articles was collected from Media Cloud, an open-source platform aggregating millions of stories published online (Roberts et al., 2021). We collected the metadata and full text of all news articles from January 1, 2020 to June 30, 2020 in 10 languages, thanks to white-listing by Media Cloud. Overall, this collection includes news articles in the following languages: English (31M articles), Spanish (8.2M), Russian (7.2M), German (3.2M), French (3.2M), Arabic (2.9M), Italian (2.4M), Turkish (1.1M), Polish (595K), and Mandarin Chinese (342K). We hired and trained annotators with fluency in these languages. The total dataset consists of about 60M articles. Article metadata includes dates, headlines, and URLs of articles. To filter, match, and sample pairs of news articles for annotation, we apply a series of processing steps to the dataset (Figure 1), described below. The annotated data is available on Zenodo while the full text of most webpages annotated is available in a special collection at the Internet Archive. We have also created a Python package to crawl and process the webpages.

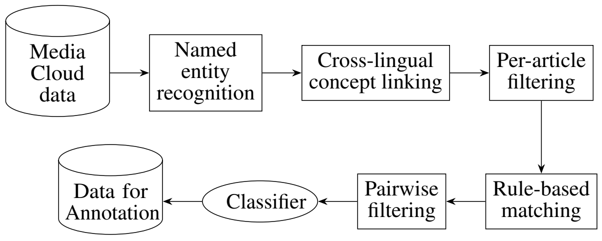


Figure 1: Filtering and pair matching pipeline.

Filtering. We applied a series of filtering steps to clean the data. First, we filtered out articles that miss one of basic metadata attributes: story ID, URL, title, or text. Second, we dropped data points that do *not* correspond to news articles of social or political importance[3] and very short articles whose word count is less than 100. Third, we filtered out articles that have titles or URLs that exactly match a newer news article. After applying these filtering steps, the numbers of articles per language are: English (10M articles), Spanish (4.6M), Russian (1.8M), German (1.3M), French (1.2M), Arabic (1.8M), Italian (1.5M), Turkish (655K), Polish (369K), and Mandarin Chinese (205K).

Matching and Sampling of News Pairs. Randomly sampled pairs of news articles are unlikely to be related. Therefore, a major design point in our pilot work was to identify meaningful candidate pairs. We experimented with document embeddings (Cr5: Josifoski et al., 2019), sentence embeddings (Sentence BERT: Reimers and Gurevych, 2019) applied to headlines and lead paragraphs, and named entities (spaCy, polyglot, and Babelfy; Moro et al., 2014) to identify similar articles. With extensive pilot study, we devised an efficient sampling pipeline (Figure 1). First, the named entities of each article are extracted using spaCy and

[3]Irrelevant websites include: "reddit.com," "facebook.com," "twitter.com," "fb.com," "wikipedia.org," "epochtimes.com," "youtube.com," "slideshare.net". We also dropped any url with 'sport' in it.

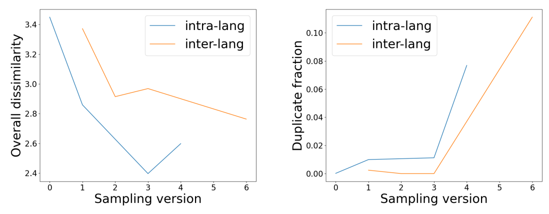


Figure 2: The average dissimilarity of news article pairs (left) and the fraction of duplicates (right) per sampling version.

polyglot.[4] For monolingual pairs, we select pairs of articles having high Jaccard similarity of these named entities. For cross-lingual pairs, we attempt to match the named entities to Wikipedia article titles and store the Wikidata concept ids of matching Wikipedia articles, which are language agnostic. We then select cross-lingual pairs of articles having high Jaccard similarity of these Wikidata concept ids.

To remove duplicate articles (i.e., articles that have the same or nearly the same text, but are published with different titles and URLs), we drop all pairs of articles that share one or more long sentences (of 40 or more characters) or where the Jaccard similarity of article text is higher than a certain threshold. Once the training set of news article pairs was annotated, we trained a logistic regression classifier that was used for further sampling. The features included in the classifier are: the word counts of both articles, the number of common words, the number of common named entities, cosine similarity of the named entities with BM25 embeddings (Robertson and Zaragoza, 2009), text Jaccard similarity, and an exponentially decaying function of publication date difference.

The pipeline was updated over time to increase the fraction of OVERALL similar pairs among samples (Figure 2, left). Version 1 of our sampling pipeline selects pairs based solely on the Jaccard similarity of named entities without any classifier, since initially no labeled data was unavailable. Version 2 introduces a temporal window where only articles published within a few days from each other are considered. Version 3 introduces a minimal threshold for Jaccard similarity of named entities. Versions 4 and 5 count the reappearance of words for Jaccard similarity and implement a more efficient similarity computation in Cython, respectively. Version 6 removes the word reappearance counts after an evaluation of its effectiveness. We note that while improvements to matching and sampling increased the fraction of similar news articles, we also experienced a small increase in the fraction of duplicate news articles (Figure 2, right).

[4]To scale the pipeline to tens of millions of articles, we use the efficient, simple language models rather than the transformer models in spaCy version 3.

2.2 Annotation

Annotation guidelines were developed through an iterative process, grounded in media studies literature on news. After several pilot annotation rounds, we formed a detailed codebook for seven dimensions of similarity. The questions were:

GEO How similar is the geographic focus (places, cities, countries, etc.) of the two articles?

ENT How similar are the named entities (e.g., people, companies, organizations, products, named living beings), excluding previously considered locations appearing in the two articles?

TIME Are the two articles relevant to similar time periods or describing similar time periods?

NAR How similar are the narrative schemas presented in the two articles?

OVERALL Overall, are the two articles covering the same substantive news story? (excluding style, framing, and tone)

STYLE Do the articles have similar writing styles?

TONE Do the articles have similar tones?

Annotators answered each question using a four-point Likert scale with the options, “Very Dissimilar,” “Somewhat Dissimilar,” “Somewhat Similar,” and “Very Similar.” In this paper, we represent these ordinal labels as numbers from 4 (Very Dissimilar) to 1 (Very Similar). In addition, each question can be answered with the option “Other”, which is used mainly for marking pairs of duplicate news articles and unavailable articles, e.g., due to a paywall or take-down (annotators were asked to report such cases via a free-text comment). The annotation codebook defines each dimension and gives examples with explanations of labeled news article pairs (e.g., Table 1).

To achieve the desired linguistic diversity and magnitude of news annotation we trained 25 annotators hired across 3 institutions (GESIS, UMass, UMich), out of which 10 labeled over 1,000 news article pairs during the course of roughly six months (Table 2). Annotators were compensated €12 per hour at GESIS and $15 per hour at UMass and UMich.

We implemented a custom annotation interface in Ruby and MongoDB that assigns news articles pairs at random within the language abilities of

Article 1	Article 2	GEO	ENT	TIME	NAR	STY	TONE	OVERALL
NYC testing two more people for coronavirus	New York City Reports 2 Additional Suspected Cases of Coronavirus	VS	VS	VS	VS	VS	VS	VS
Video of a man beating his girlfriend mercilessly goes viral	Curry house in Worcester improves from one-star to five food hygiene rating	VD	SD	SD	VD	SS	SD	VD
All with flu symptoms to be tested for Covid-19 in Chandigarh	ICMR study points towards possible community transmission of coronavirus COVID-19 in India	SD	SD	SS	SD	SS	SS	SD

Table 1: Example annotated pairs. The pairs were annotated based on the full-text of the articles. Each of the seven dimension is annotated with a Likert scale with four options: Very Similar (VS), Somewhat Similar (SS), Somewhat Dissimilar (SD), and Very Dissimilar (VD). The articles in the first pair released very similar information about two people tested Coronavirus positive at New York City. The second pair is very dissimilar since one article described the violence against a women while the other one reported the rating improvement of a restaurant. They shared nothing in common. The final pair overlapped somewhat in terms of GEO (India), ENT (Indian Council of Medical Research, ILI, severe acute resparatory illness), and TIME. The two articles, however, still refer to different events.

id	items	shared	seconds/item	correlation
1	1,657	809	296	0.88
2	2,311	495	344	0.86
3	1,197	611	237	0.85
4	1,178	794	213	0.85
5	134	98	153	0.84
6	1,036	626	128	0.84
7	1,302	345	220	0.84
8	466	398	224	0.84
9	787	208	233	0.84
10	887	368	506	0.83
11	1,062	466	387	0.82
12	361	321	311	0.81
13	262	213	165	0.81
14	139	135	235	0.79
15	1,076	716	71	0.77

Table 2: Annotators and their statistics: the number of labeled items (news article pairs), the number of shared items (also annotated by another annotator), median number of seconds to label an item, and Pearson correlation of their OVERALL labels with the mean labels of other annotators. Only annotators with at least 100 labels are shown.

each annotator. To engage and motivate annotators, the interface also provides feedback to annotators in the form of basic statistics such as the number of annotations, and the inter-rater agreement of the top annotators. The interface also shows past annotations and highlights disagreements, which were discussed at biweekly video conference meetings.

Codebook, Training, & Annotation. The annotation process has multiple stages (Figure 3). All annotators read the codebook and attended a training session at which the codebook and annotations of example pairs were discussed. The annotators then completed 30 practice annotation pairs independently. After completing each practice pair, annotators were able to view the gold standard labels that had been agreed by all of the SemEval task authors. Each gold standard label was accompanied by a written explanation of why the label was assigned. Question and answer sessions were held at which the annotations were discussed as well.

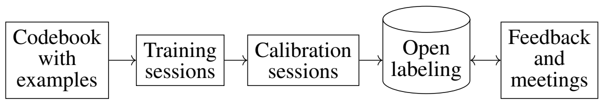


Figure 3: Annotator workplan.

Annotators then labeled another 30 gold standard pairs having detailed explanations, which we used to calibrate annotators' understanding of the codebook. Any disagreements were discussed until agreement was reached. The practice and calibration pairs were all English-language articles, which was a shared language ability between all our annotators. All news article pairs with gold standard labels and explanations, as well as the codebook, are released on Zenodo.

After these practice and calibration activities, pairs were annotated by a variable number of annotators, usually 1, 2, or 3, in the "open labeling" phase, where news article pairs were served to annotators continuously. Annotators were given feedback in the annotation interface on their agreement

languages	annotations	mean(OVERALL)
en	5,189	2.92
de	2,166	2.56
es	955	2.40
zh	866	2.24
de-en	863	3.20
tr	817	2.79
pl	584	2.36
ar	572	2.41
es-en	504	2.79
it	411	2.65
es-it	320	2.29
ru	289	2.78
zh-en	253	3.04
fr	184	2.39
de-fr	116	1.88
pl-en	77	2.38
de-pl	35	1.69
fr-pl	11	1.91

Table 3: The number of annotations and the mean OVERALL label (the higher, the more dissimilar) across the 10 languages and their combinations.

	GEO	ENT	TIME	NAR	OVERALL	STYLE	TONE
Krippen.	0.73	0.69	0.57	0.69	0.77	0.46	0.38
Gwet	0.81	0.67	0.75	0.69	0.76	0.69	0.67

Table 4: Inter-rater agreement measures, Krippendorf's α and Gwet's AC_1, for each labeled dimension.

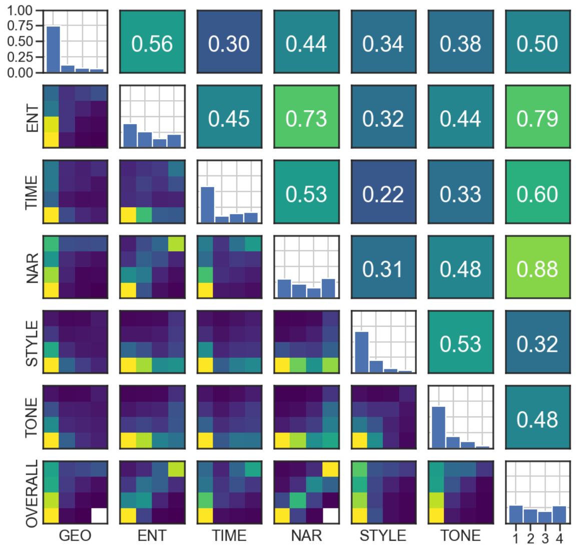


Figure 4: Histograms and Pearson correlations of every pair of scores in the labelled data.

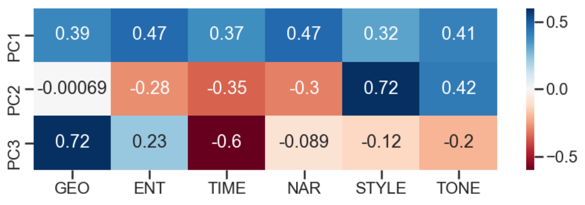


Figure 5: Heatmap showing the coefficients of the first three principal components of variation in the scores.

with other annotators, met regularly to discuss disagreements, and had an open channel for communication on a shared Slack instance. After annotating English-language pairs, non-English pairs were introduced and discussed with annotators. Finally, cross-language pairs were also introduced. The total number of annotations and average OVERALL label per each language pair is shown in Table 3.

Inter-annotator Agreement. The inter-rater agreement on the OVERALL similarity dimension is very high, with a Krippendorff's α of 0.77. We note that the distribution over labels is generally not uniform, e.g., the labeled news article pairs are skewed towards "Very Similar" in TIME, STYLE, and TONE (Figure 4). Gwet's AC_1 is known to be less sensitive to non-uniform marginal label distributions (Gwet, 2008), and it suggests a good agreement in all dimensions (Table 4).

Annotators vary in terms of the quantity and quality of the provided annotations. To compare the performance of annotators to the performance of models, we measure the inter-rater agreement of each annotator in a way that corresponds to the score of models, i.e., as a Pearson correlation between the labels of that annotator and a series of average labels from other annotators. Note, however, that this correlation is measured over labels contributed by the given annotator to both training and evaluation datasets, it is biased by the language-abilities of the annotator, and the mean label does not include the ego annotator. Our top 5 annotators consistently reach very high agreement scores of 0.85–0.88, whereas bottom 5 annotators reach agreement scores of 0.73–0.80 (Table 2).

2.3 Statistics of the Labeled Dataset

Figure 4 illustrates the relationship between the multiple similarity dimensions in the annotated dataset. The bar charts on the diagonal represent the distribution of annotations, from 1 (Very Similar) to 4 (Very Dissimilar). Panels below the diagonal represent two-dimensional histograms, and panels above the diagonal report the Pearson correlation between different dimensions, namely GEO, ENT, TIME, NAR, STYLE, TONE, and OVERALL.

language(s)	train	eval	mean(OVERALL)
ar	274	298	2.41
de	857	608	2.57
de–en	531	185	3.18
de–fr		116	1.88
de–pl		35	1.69
en	1,800	236	2.86
es	570	243	2.34
es–en		496	2.79
es–it		320	2.29
fr	72	111	2.39
fr–pl		11	2.00
it		411	2.65
pl	349	224	2.35
pl–en		64	2.35
ru		287	2.78
tr	465	275	2.74
zh		769	2.22
zh–en		213	3.07
Totals	4,918	4,902	2.62

Table 5: The number of news article pairs in the training and evaluation datasets by language and the mean OVERALL label (the higher, the more dissimilar).

NAR and ENT show the highest correlation with OVERALL (0.88 and 0.79 respectively). These two dimensions also provide the largest contributions to the variation in annotations, as indicated by the first component of the PCA shown in Figure 5.

There is no significant difference between the training and evaluation datasets with respect to the labels of any similarity dimension, and these results are also found when the dataset is disaggregated by language pair.

3 Task

3.1 Task Description & Rules

The Task was created on CodaLab.org[5] and advertised with alongside the other SemEval 2022 tasks. Participants were told, "The task is: Given a pair of news articles, are they covering the same news story? This SemEval task aims to develop systems that identify multilingual news articles that provide similar information. This is a document-level similarity task in the applied domain of news articles, rating them pairwise on a 4-point scale from most to least similar."

Participants were given 60 English-language pairs for trial data in August 2021. The training data was released to participants in two batches: the first batch was released on September 15, 2021, and the second was released on November 4, 2021. The training data consisted of article pairs in 8 different language combinations (Table 5).

Due to copyright restrictions, we were unable to release the raw text of the news articles included in the training data. In lieu of this, we developed and shared a Python package to download the text of news articles. For the training data, the downloader tries to fetch the articles from the Internet Archive or the live web and parse them with `newspaper3k`.[6] This mirrored the actions of annotators who were given links to the articles on the Internet Archive and live web.

The evaluation data was released on January 10, 2022, and consists of 4,902 pairs of news articles across 18 languages. For the evaluation data, we only included pairs of articles where both news articles were available on the Internet Archive.[7]

For both the training and evaluation datasets, we removed any article pairs where one or more annotators labeled the OVERALL similarity as "Other". This usually indicated that the pair was unavailable or not a news article.

The evaluation period ran from January 10 to February 3, 2022. This date reflected the extra time needed to download the articles as well as a short extension due to technical issues with the Codalab system. Participants were allowed to submit up to 5 submissions per day and 1,000 submissions overall.

3.2 Baselines

Our baseline models use SVC with linear kernel, logistic regression, random forest, and XGBoost (Chen and Guestrin, 2016). For feature selection, we found positive correlation between the fraction of "Very Similar" news article pairs and their Jaccard similarity in terms of named entities, as well as full text (Figure 6). Thus we evaluate three sets of features in the baseline models: set-A (Jaccard similarity of named entities), set-B (set-A and text Jaccard similarity), and set-C (set-B and word count difference).

3.3 Evaluation and Ranking

The teams were evaluated using Pearson's r correlation with the mean OVERALL labels on the

[5]`https://competitions.codalab.org/competitions/33835`

[6]Some participants found better success at parsing the articles using `trafilatura` (Barbaresi, 2021)

[7]A special collection on the Internet Archive now includes most webpages.

evaluation data. In ranking teams, we were inspired by recent work (e.g., Dodge et al., 2019) on estimating model performance while recognizing that not all systems solving a task are on equal footing. Specifically, some teams may have submitted more or fewer submissions due to time, computational budget, or model performance. Having varied numbers of submissions for each team/system creates an opportunity for rethinking how to estimate how well the system actually does.

Our approach ranks teams by bootstrapping their expected rank under certain constraints. We assume that for most teams, submissions are an exploration of the hyperparameter/model configuration space of their system. Each submission's score is then informative of the distribution of its expected performance. To create the official Task rankings, we bootstrap the expected rank from all teams' submissions. Specifically, we bootstrap rankings by sampling an equal number of submissions (n=5) from the most-recent 50 submissions of each team and then use the maximum score from each team's sampled submissions to compute one ranking of all teams. To get our final ranking, we repeat this process to sample n=10,000 rankings and take the average rank for each team across these samples. In essence, this process measures the expected ranking if each team was given the same number of hyperparameter/configuration searches.

In practice, our new ranking approach largely does not change the ranking from simply ordering teams by their highest-performing submission. However, a handful of teams did shift positions. The relative stability suggests that models were not affected by different hyperparameter/configuration selections.

4 Results

The task received over 500 public submissions from over 30 participants. Next, we provide an overview of the baselines and the approaches that have been adopted by the 19 teams who participated in the competition's leaderboard and submitted a description of their systems.

4.1 Summary of the Approaches

The teams explored a staggering range of approaches, including multimodal systems that encode the articles' images and knowledge-based features (Zosa et al., 2022). Systems were evaluated on their ability to assess news similarity of pairs of

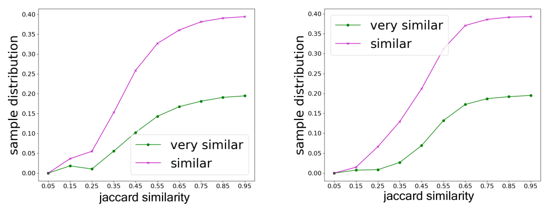


Figure 6: Sample distribution within different Jaccard similarity of named entities (left) and text (right). The "Similar" class includes both the "Very Similar" and "Somewhat Similar" labels

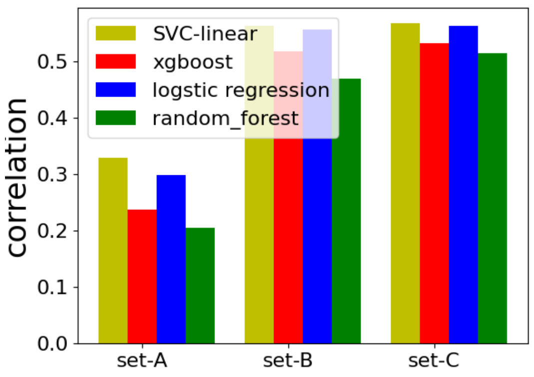


Figure 7: Baseline performances for feature sets.

cross-lingual news articles, and on a secondary task involving only pairs of articles in English. While some teams developed dedicated systems for the two evaluations, cross-lingual and English-only, the great majority of such systems were variations of a single design. For the sake of conciseness, in the remainder of the section we will restrict discussion to the cross-lingual news similarity task and to the best-performing systems. Table 6 reports salient characteristics of these systems, in order of their ranking using Pearson's correlation coefficient. The participant describe each system in finer detail and offer valuable insights on adapting them to the English-only subtask and on negative results (Nai et al., 2022; Wangsadirdja et al., 2022; Pisarevskaya and Zubiaga, 2022; Zosa et al., 2022; Singh et al., 2022; Giovanni et al., 2022; Hajjar et al., 2022; Chen et al., 2022; Joshi et al., 2022; Heil et al., 2022; Sandeep et al., 2022; Xu et al., 2022; Kuimov et al., 2022; Ishihara and Shirai, 2022; Luo et al., 2022; Jobanputra and Rodriguez, 2022; Dufour et al., 2022; Bhavsar et al., 2022; Stefanovitch, 2022).

TEAM	TRANSFORMERS							CROSS-LANG.		DATA HANDLING						TECHNIQUE	
	XLM-RoBERTa	**MPNet**	**SBERT**	**mBERT**	**other**	**bi- or cross-emb.**	**finetune/pre-train**	**multilang. emb.**	**translation**	**fields used**	**splices text**	**intermediate sim**	**NER**	**DA/external data**	**others**	**ensemble/stacking**	**additional tasks**
HFL	x					cross	yes	x	x	TB	x			x		x	**
GateNLP-UShef					LaBSE	bi	yes	x	x	TB			x	x		x	
DataScience-Polimi		x				bi	yes	x	x	TBD				x		x	
ITNLP2022	x				infoxlm	cross	yes	x	x	TBK				x		x	****
EMBEDDIA			x			bi	yes	x		TB				x			
HuaAMS	x					cross	yes	x	x	B				x		x	
WueDevils			x		USE	bi	no	x		TBP		x				x	
DartmouthCS				x		neither	yes	x	x	B						x	**
Nikkei				x	bert	bi	yes	x	x	TB		x		x		x	
YNU-HPCC				x		cross	yes	x		B							
SkoltechNLP					xlm-mlm	bi	yes	x		B							
Team Innovators					DeBERTa	cross	no		x	TBD	x			x			***
TCU				x		cross	yes	x	x	B							
OversampledML		x				neither	no	x	x	TB	x	x	x				**
BL.Research			x		NER-tf, BART	neither	no		x	TB	x	x	x		*		
LSX_team5			x			neither	no		x	B		x					
TMA					LASER	neither	no	x		TBDP	x	x					
dina	x					cross	no		x	B	x		x				
IIIT-MLNS					distilbert	bi	yes	x		TBDK			x	x		x	

Table 6: A summary of submitted models ordered by their performance. For each TEAM, the table reports common choices in terms of TRANSFORMER architecture, approaches for tackling CROSS-LINGUAL input, DATA HANDLING such as feature engineering and augmentation, and learning TECHNIQUE. Legend: T = title, B = body, D = description, K = keywords, P = publication date, * = sentiment, topics, geocoding, ** = 6 subdimensions, *** = semantic similarity, hyperpartisan news, **** = 3 subdimensions

4.2 Rankings and Variation Across Languages

The final rankings for the multilingual task as well as the English-language only subset are shown in Supplemental Table A1. Overall performance on the multilingual task (as measured with Pearson's r) ranged from 0.35 to 0.82 with a mean of 0.66 and a median of 0.72.

The highest single-language performance was achieved on French (median 0.84, max 0.87) and French–Polish (median 0.82, max 0.95) pairs. The worst performance was on German–French pairs (median 0.60, max 0.72). Supplemental Figure A1 shows the distribution of the best scores achieved by each team in each language.

Among the baseline models, we find that the SVC performs best, while the Jaccard similarities of named entities and text matter more than word count difference (Figure 7). However, the majority of submitted models perform significantly better than the baseline models.

4.3 Nuanced Inputs: Multiple Fields, Fine-tuning, & Feature Engineering

In addition to the main body of the articles, most systems leveraged information from multiple fields such as their titles and descriptions. All systems involved deep neural embeddings of those fields, with all but one team using Transformer-based architectures. The top-ranking system used several techniques to optimize an XLM-RoBERTa-based model without further feature engineering. *Accurately embedding multiple fields of the articles appears a crucial source of performance. Systems that engaged in fine-tuning or continued pre-training the embeddings scored higher on average.* Yet, there was no clear pattern on which architecture would produce performant representations for the task. In particular, the teams offered mixed evidence on the superiority of bi-embedding over cross-embedding approaches for the task. For example, teams Nikkei and SkoltechNLP found bi-encoders to outperform cross-encoders, whereas team HFL found the opposite (Ishihara and Shirai, 2022; Kuimov et al., 2022; Xu et al., 2022).

To improve upon the baseline of the sole embeddings (albeit often marginally), 10 teams experimented with additional feature engineering. Several teams explored forms of keyword and named-entity extraction. These approaches were arguably promising in that they mirrored the process of sampling,[8] though the results offer no conclusive ev-

[8]The sampling process was not shared with teams.

idence. Similarly, teams also tested strategies to focus the input around the most informative parts of the articles. This was due to multiple factors: first, the limitations of Transformer-based architectures which can only handle limited-length input; second, a small but consistent number of errors in automatically parsing the articles adding noise to the text; and last, the nature of the task: according to the "inverted pyramid" writing style, the start of a news article often summarizes the most important information. Thus, participants experimented with splicing the article body, which led to performance improvements.

4.4 Tackling Generalization: Multilingual, Augmentation, & Learning Strategies

A challenging characteristic of the task is the presence of cross-lingual pairs of articles—with several new language combinations introduced first in the evaluation data. The teams approached the challenge by resorting to multilingual embeddings or machine translating the articles to a high-resource language. *The best-performing systems employed a combination of both approaches, multilingual embeddings and translation, as part of a broader strategy for data augmentation.* Furthermore, the best-performing systems *resorted to forms of ensemble learning* such as stacking, which offered a further way to improve the generalization of the models (with the exceptions of teams TCU and dina (Luo et al., 2022; Pisarevskaya and Zubiaga, 2022)). With few exceptions (see Jobanputra and Rodriguez, 2022), optimizing for multiple tasks also seems to improve performance—e.g., the top-ranking system jointly learns all seven dimensions of similarity provided in the training data.

4.5 Simplicity–Performance Trade-Offs

While the best-performing systems explore sophisticated designs and techniques, the teams also suggested simpler methods that prove surprisingly effective. In fact, several teams found that simple systems outperformed more complex approaches in their experiments. A system that relies on pretrained embeddings without fine-tuning achieved a performance of 0.759 Pearson's correlation coefficient (Wangsadirdja et al., 2022) vis-à-vis the top score of 0.818 (Xu et al., 2022). Similarly, a baseline regressing over two features—shared named entities and cosine similarity between the article embeddings—scored as high as 0.677 (Sandeep et al., 2022). Finally, several teams reported performance improvement by using lexical features without particular adaptation to the cross-lingual settings of this task (e.g., teams Nikkei & TMA: Ishihara and Shirai, 2022; Stefanovitch, 2022). In a nutshell, carefully reflecting the characteristics of the task into the system design can lead to good performance even with simpler models.

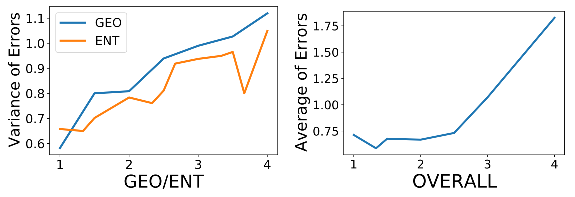


Figure 8: The variance of models' error against pairs with the same GEO/ENT as a function of GEO/ENT (left) and an average of models' error against pairs that were at least somewhat similar in GEO, ENT, TIME, and NAR (right).

4.6 Error Analysis

In this section, we analyze the errors of submitted models. Twenty-one teams achieved an accuracy of at least 0.70, and we focus our error analysis on these teams.

First, we compute the correlation between each model's error (absolute difference between the predicted OVERALL and the OVERALL reported by the annotators) and the sub-dimensions (GEO, ENT, TIME, and NAR) for each pair. We find a strong Pearson correlation between the variance of errors and the GEO and ENT sub-dimensions: the correlation for GEO is 0.97, while for ENT it is 0.88 (Figure 8, left).

We hypothesized that if there is a pair with high similarity in terms of GEO, ENT, TIME, and NAR but dissimilar in terms of the OVERALL label, then models will have difficulties against this pair. To test this hypothesis, we select only pairs that are Somewhat/Very Similar in terms of GEO, ENT, TIME, *and* NAR dimensions. Then, we report how the average error of models varies for different OVERALL ratings. We expect that the average error will be higher for pairs with higher OVERALL values (i.e., pairs that are more dissimilar overall). Figure 8 (right) shows the result of this analysis. We can see that there is a strong correlation between the average of error and the OVERALL label. The Pearson correlation between the error and OVERALL for the selected pairs is 0.88.

5 Discussion

Multilingual news article similarity is a challenging problem despite sharing some characteristics with Semantic Text Similarity. Participants in this SemEval task tried a number of innovative approaches to the problem. Systems that used multiple parts of the article (headline, body, publication date) and systems that fine-tuned or otherwise trained embeddings generally performed better than those that did not. The best-performing systems generally combined multilingual embeddings and translation. Nonetheless, there was no clear consensus as to the best architectures, embedding models, or preprocessing to perform on the data.

There were clear variations across languages, and more work is needed to create multilingual systems that work across diverse language combinations. Errors were particularly common when the news articles shared some similarity in terms of their geographic focus, temporal focus, named entities, and narratives but were nonetheless dissimilar overall. While the best-submitted model achieved a very high correlation of 0.82 with gold standard labels, the best human annotator reached 0.88 correlation, which suggests ample space for further progress.[9]

Our dataset is drawn from the first half of 2020 and covers several geopolitical events (e.g., BlackLivesMatter) as well as the first wave of the COVID-19 pandemic. The nearly 10,000 annotated pairs of news articles across 18 combinations of 10 different languages will enable exciting developments in natural language processing methods as well as social science studies of how the global media reported on this unique period.

Acknowledgments

This research has received funding through the Volkswagen Foundation. We thank Media Cloud for data access. We thank the Internet Archive which made it possible for participants to all have access to the same data. We are deeply grateful to the annotators and task participants: thank you.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Daniel Bär, Torsten Zesch, and Iryna Gurevych. 2011. A reflective view on text similarity. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 515–520.

Adrien Barbaresi. 2021. Trafilatura: A web scraping library and command-line tool for text discovery and extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 122–131, Online. Association for Computational Linguistics.

Nidhir Bhavsar, Rishikesh Devanathan, Aakash Bhatnagar, Muskaan Singh, and Tirthankar Ghosal. 2022. Team Innovators @ SemEval-2022 for Task 8 : Multi-Task Training with Hyperpartisian and Semantic Relation for Multi-Lingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation.*

Desmond Bala Bisandu, Rajesh Prasad, and Musa Muhammad Liman. 2018. Clustering news articles using efficient similarity measure and n-grams. *International Journal of Knowledge Engineering and Data Mining*, 5(4):333–348.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610.

Snigdha Chaturvedi, Shashank Srivastava, and Dan Roth. 2018. Where have i heard this story before? identifying narrative similarity in movie remakes. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 673–678.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Zhongan Chen, Weiwei Chen, Yunlong Sun, Hongqing Xu, Shuzhe Zhou, Bohan Chen, Chengjie Sun, and Yuanchao Liu. 2022. ITNLP2022 at SemEval-2022 Task 8 : Pre-trained Model with Data Augmentation and Voting for Multilingual News Similarity. In *The 16th International Workshop on Semantic Evaluation.*

[9]Note that these two correlations are computed on different sets of news article pairs, since the performance of a model is estimated on the evaluation set, while the correlation score of humans is computed on all their annotations.

Teun A van Dijk. 1988. *News as discourse*. University of Groningen.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A Smith. 2019. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2185–2194.

Sébastien Dufour, Mohamed Mehdi Kandi, Karim Boutamine, Camille Gosset, Mokhtar Boumedyen Billami, Christophe Bortolaso, and Youssef Miloudi. 2022. BL . Research at SemEval-2022 Task 8 : Using various Semantic Information to evaluate document-level Semantic Textual Similarity. In *The 16th International Workshop on Semantic Evaluation*, pages 1–8.

Seth Flaxman, Sharad Goel, and Justin M. Rao. 2016. Filter Bubbles, Echo Chambers, and Online News Consumption. *Public Opinion Quarterly*, 80(S1):298–320.

Marco Di Giovanni, Thomas Tasca, and Marco Brambilla. 2022. DataScience-Polimi at SemEval-2022 Task 8 : Stacking Language Models to Predict News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Kilem Li Gwet. 2008. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48.

Joseph Hajjar, Weicheng Ma, and Soroush Vosoughi. 2022. DartmouthCS at SemEval-2022 Task 8: Predicting Multilingual News Article Similarity with Meta-Information and Translation. In *The 16th International Workshop on Semantic Evaluation*.

Stefan Heil, Karina Kopp, Konstantin Kobs, Albin Zehe, and Andreas Hotho. 2022. LSX _ team5 at SemEval-2022 Task 8 : Multilingual News Article Similarity Estimation based on Pre-Trained Transformers , ConceptNet Embeddings and Word Mover ' s Distance. In *The 16th International Workshop on Semantic Evaluation*.

Shotaro Ishihara and Hono Shirai. 2022. Nikkei at SemEval-2022 Task 8 : Exploring BERT-based Bi-Encoder Approach for Pairwise Multilingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Mayank Jobanputra and Lorena Martin Rodriguez. 2022. OversampledML at SemEval-2022 Task 8 : When multilingual news similarity met Zero-shot approaches. In *The 16th International Workshop on Semantic Evaluation*.

Sagar Joshi, Dhaval Taunk, and Vasudeva Varma. 2022. IIIT-MLNS at SemEval-2022 Task 8 : Siamese Architecture for Modeling Multilingual News Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Martin Josifoski, Ivan S. Paskov, Hristo S. Paskov, Martin Jaggi, and Robert West. 2019. Crosslingual document embedding as reduced-rank ridge regression. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM.

Christian Klein and Matías Martínez. 2009. Wirklichkeitserzählungen. felder, formen und funktionen nicht-literarischen erzählens. In *Wirklichkeitserzählungen*, pages 1–13. Springer.

Mikhail Kuimov, Daryna Dementieva, and Alexander Panchenko. 2022. SkoltechNLP at SemEval 2022 Task 8 : Multilingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Michael D Lee, Brandon Pincombe, and Matthew Welsh. 2005. An empirical evaluation of models of text document similarity. In *Proceedings of the annual meeting of the cognitive science society*, volume 27.

Xiang Luo, Yanqing Niu, and Boer Zhu. 2022. TCU at SemEval-2022 Task 8 : A Stacking Ensemble Transformer Model for Multilingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Maxwell McCombs. 2005. A look at agenda-setting: Past, present and future. *Journalism studies*, 6(4):543–557.

Ben Miller, Jennifer Olive, Shakthidhar Gopavaram, and Ayush Shrestha. 2015. Cross-document non-fiction narrative alignment. In *Proceedings of the First Workshop on Computing News Storylines*, pages 56–61.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: A unified approach. *Transactions of the Association for Computational Linguistics (TACL)*, 2:231–244.

Zihan Nai, Jin Wang, and Xuejie Zhang. 2022. YNU-HPCC at SemEval-2022 Task 8 : Transformer-based Ensemble Model for Multilingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Dong Nguyen, Dolf Trieschnigg, and Mariët Theune. 2014. Using crowdsourcing to investigate perception of narrative similarity. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 321–330.

Zhongdang Pan and Gerald M Kosicki. 1993. Framing analysis: An approach to news discourse. *Political communication*, 10(1):55–75.

Dina Pisarevskaya and Arkaitz Zubiaga. 2022. Team dina at SemEval-2022 Task 8 : Enhancing Pre-trained Language Models for Semantic Similarity. In *The 16th International Workshop on Semantic Evaluation*.

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

Hal Roberts, Rahul Bhargava, Linas Valiukas, Dennis Jen, Momin M. Malik, Cindy Bishop, Emily Ndulue, Aashka Dave, Justin Clark, Bruce Etling, Rob Faris, Anushka Shah, Jasmin Rubinovitz, Alexis Hope, Catherine D'Ignazio, Fernando Bermejo, Yochai Benkler, and Ethan Zuckerman. 2021. Media Cloud: Massive Open Source Collection of Global News on the Open Web. In *Proceedingsofthe Fifteenth InternationalAAAIConferenceonWeb andSocial Media (ICWSM2021)*.

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

Jan Rupnik, Andrej Muhic, Gregor Leban, Primoz Skraba, Blaz Fortuna, and Marko Grobelnik. 2016. News across languages-cross-lingual document similarity and event tracking. *Journal of Artificial Intelligence Research*, 55:283–316.

Sai Sandeep, Sharma Chittilla, and Talaat Khalil. 2022. HuaAMS at SemEval-2022 Task 8 : Combining Translation and Domain Pre-training for Cross-lingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Iknoor Singh, Yue Li, Melissa Thong, and Carolina Scarton. 2022. GateNLP-UShef at SemEval-2022 Task 8 : Entity-Enriched Siamese Transformer for Multilingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Nicolas Stefanovitch. 2022. Team TMA at SemEval-2022 Task 8 : Lightweight and Language-Agnostic News Clustering. In *The 16th International Workshop on Semantic Evaluation*.

Gaye Tuchman. 1972. Objectivity as strategic ritual: An examination of newsmen's notions of objectivity. *American Journal of Sociology*, 77(4):660–679.

Dirk Wangsadirdja, Felix Heinickel, Simon Trapp, Albin Zehe, Konstantin Kobs, and Andreas Hotho. 2022. WueDevils at SemEval-2022 Task 8 : Multilingual News Article Similarity via Pair-Wise Sentence Similarity Matrices. In *The 16th International Workshop on Semantic Evaluation*.

Steve J Westerman, Timothy Cribbin, and Julie Collins. 2010. Human assessments of document similarity. *Journal of the American Society for Information Science and Technology*, 61(8):1535–1542.

Zihang Xu, Ziqing Yang, Yiming Cui, and Zhigang Chen. 2022. HFL at SemEval-2022 Task 8 : A Linguistics-inspired Regression Model with Data Augmentation for Multilingual News Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Elaine Zosa, Emanuela Boros, Boshko Koloski, and Lidia Pivovarova. 2022. EMBEDDIA at SemEval-2022 Task 8 : Investigating Sentence , Image , and Knowledge Graph Representations for Multilingual News Article Similarity. In *The 16th International Workshop on Semantic Evaluation*.

Appendix

Team	Rank	Max Score	Mean Score
HFL	1	0.818	0.788
GateNLP-UShef	2	0.801	0.781
cyk1337	3	0.792	0.682
ITNLP2022	4	0.784	0.633
EMBEDDIA	5	0.784	0.685
L3i	6	0.783	0.688
DataScience-Polimi	7	0.790	0.656
HuaAMS	8	0.771	0.759
WueDevils	9	0.759	0.711
DartmouthCS	10	0.748	0.509
aim	11	0.748	0.686
Nikkei	12	0.743	0.718
SkoltechNLP	13	0.734	0.596
Andi	14	0.726	0.723
Team Innovators	15	0.733	0.690
BUT	16	0.726	0.588
sebduf	17	0.706	0.701
BL.Research	18	0.703	0.688
OversampledML	19	0.701	0.679
TCU	20	0.715	0.511
Ormus	21	0.701	0.567
LSX_team5	22	0.572	0.572
dina	23	0.507	0.228
Elena_Shu	24	0.492	0.332
naizihan	25	0.475	0.411
TMA	26	0.507	0.352
IIIT-MLNS	27	0.441	0.301
rahul19266	28	0.350	0.268
EAS	29	0.391	0.163

(a) Multilingual Setting

Team	Rank	Max Score	Mean Score
HFL	1	0.872	0.839
EMBEDDIA	2	0.855	0.704
L3i	3	0.855	0.786
WueDevils	4	0.857	0.822
DataScience-Polimi	5	0.873	0.770
DartmouthCS	6	0.845	0.647
cyk1337	7	0.837	0.725
ITNLP2022	8	0.833	0.777
aim	9	0.839	0.773
GateNLP-UShef	10	0.833	0.813
SkoltechNLP	11	0.871	0.716
BL.Research	12	0.828	0.820
sebduf	13	0.824	0.821
OversampledML	14	0.814	0.794
Team Innovators	15	0.829	0.764
HuaAMS	16	0.804	0.792
naizihan	17	0.783	0.676
BUT	18	0.779	0.685
Andi	19	0.771	0.762
Nikkei	20	0.765	0.742
Ormus	21	0.767	0.673
TCU	22	0.755	0.743
LSX_team5	23	0.683	0.683
TMA	24	0.740	0.557
Elena_Shu	25	0.623	0.421
dina	26	0.624	0.306
EAS	27	0.659	0.346
IIIT-MLNS	28	0.542	0.350
rahul19266	29	0.366	0.299
us241077	30	0.226	0.226

(b) English-only Setting

Table A1: Rankings for each team in the official mulingual setting and in the optional English-only setting, in which one additional team participated. The final team rankings shown here were computed through the bootstrapping process described in §3.3. We additionally report the maximum and mean scores (Pearson r) for each team, which largely correspond to the same ranking as our bootstrapping process.

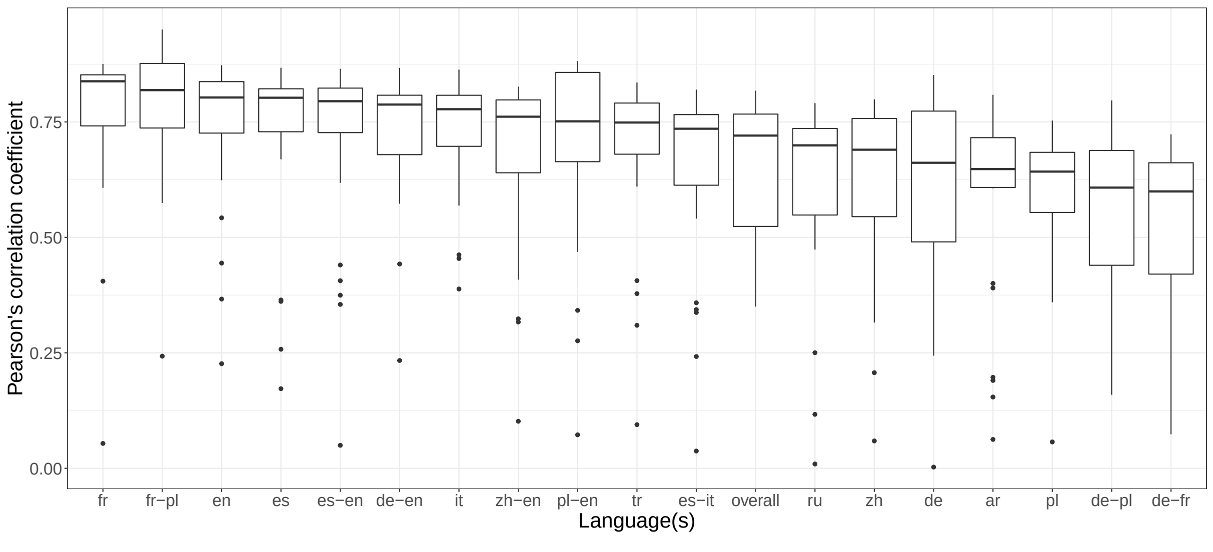


Figure A1: Distribution of the highest Pearson's correlation coefficients achieved by each team per language.

EMBEDDIA at SemEval-2022 Task 8: Investigating Sentence, Image, and Knowledge Graph Representations for Multilingual News Article Similarity

Elaine Zosa[*]
University of Helsinki
Helsinki, Finland
elaine.zosa@helsinki.fi

Emanuela Boros[*]
University of La Rochelle
La Rochelle, France
emanuela.boros@univ-lr.fr

Boshko Koloski[*]
Jožef Stefan Institute
Jožef Stefan Institute IPS
Ljubljana, Slovenia
boshko.koloski@ijs.si

Lidia Pivovarova
University of Helsinki
Helsinki, Finland
lidia.pivovarova@helsinki.fi

Abstract

In this paper, we present the participation of the EMBEDDIA team in the SemEval-2022 Task 8 (*Multilingual News Article Similarity*). We cover several techniques and propose different methods for finding the multilingual news article similarity by exploring the dataset in its entirety. We take advantage of the textual content of the articles, the provided metadata (e.g., titles, keywords, topics), the translated articles, the images (those that were available), and knowledge graph-based representations for entities and relations present in the articles. We, then, compute the semantic similarity between the different features and predict through regression the similarity scores. Our findings show that, while our proposed methods obtained promising results, exploiting the semantic textual similarity with sentence representations is unbeatable. Finally, in the official SemEval-2022 Task 8, we ranked fifth in the overall team ranking cross-lingual results, and second in the English-only results.

1 Introduction

Detecting news stories related to a single theme and combining them into news clusters has been an increasing interest in the creation of news aggregators that consolidate thousands of articles from different publishers and websites (Pranjić et al., 2020). Tracking similarity of news coverage between different outlets or regions has also been urgent and challenging. For example, whether previously with Ebola or recently with the COVID-19 pandemic, monitoring and containment of infectious disease outbreaks has remained a key component of public health strategy to contain the diseases. The ability to track disease outbreaks in an accurate manner is critical in the deployment of efficient intervention measures. As such reports may not only be in English, there is also a need for effective multilingual systems. Hence, recent research has been focused on the area of identifying similarities between documents, phrases, stories, etc.

Semantic textual similarity (STS) deals with determining how similar two groups of sentences are by measuring their semantic similarity. Over the years, several solutions were proposed to assess STS. The most general approach is pre-training on massive datasets before fine-tuning on subsequent downstream tasks (Jiang et al., 2020; Raffel et al., 2019; Lan et al., 2019; Yang et al., 2019; Liu et al., 2019; Sanh et al., 2019). Other works considered finding the similarity by classifying texts using BERT-based models (Devlin et al., 2019) with a pair of sentences packed together as input (Yang et al., 2019; Liu et al., 2019; Sanh et al., 2019; Wang et al., 2019).

The SemEval-2022 Task 8 (*Multilingual News Article Similarity*) (Chen et al., 2022) aimed at developing systems that identify multilingual news articles that provide similar information by rating them on a real-valued $[1-4]$ scale, from most to least similar.

In this paper, we cover several techniques and propose different methods for finding the multilingual news article similarity by exploring different aspects of the dataset. We consider that the textual content, the provided metadata (e.g. title, keywords, topics), representative images corresponding to the news articles, and knowledge graph-based representations for entities and relations present in the articles, would draw on a multiplicity of modes, all of which contribute to the meaning and the main story

[*] Equal contribution from all the authors.

of the news articles. Moreover, we also translate the articles to a high-resource language (English) in order to assess the ability of our models in an English-only context. Therefore, we investigate the multimodality of the data by experimenting with sentence, image, and knowledge graph embeddings in two scenarios: (1) by directly computing the semantic similarity between the different features and (2) by learning through regression and predicting the similarity scores.

2 Data

The training data has 4,964 article pairs from seven languages (English, German, Spanish, Arabic, Polish, Turkish, and French) and gold standard similarity scores for six dimensions (*Geography, Entities, Time, Narrative, Style, Tone*), plus the *Overall* score. The final evaluation data has 4,902 pairs and three "surprise" languages that were not present in the training data (Chinese, Italian, and Russian)[1].

	Train	**Eval**
Monolingual pairs	4,387	3,462
Cross-lingual pairs	577	1,440
Unseen language pairs	NA	2,000
Total	4,964	4,902
Top image	6,755	7,569

Table 1: Training and evaluation data statistics.

Moreover, the metadata includes the article titles, several specific topics and keywords, and links to representative images. The statistics of the training and final evaluation data are in Table 1. Since some of our methods use images, we also report in the table a total number of images we were able to download for the datasets. We use only images from the URL specified as *top_image* in the JSON files of the articles.

3 Experiments

Next, we detail all our approaches and perform a detailed error analysis[2]. The evaluation is performed in terms of Pearson correlation. Our results are presented in Table 2. Each type of approach is detailed with the corresponding pre-trained models[3]. Also, each type of model has an id corresponding to the subsection number is detailed (1a, 2b, etc.).

3.1 Semantic Textual Similarity

A straightforward solution for finding the similarity between two texts is approaching it with sentence embeddings. Thus, we start our experimental setup by encoding the articles with Sentence-BERT (SBERT) (Reimers and Gurevych, 2019), a modified pre-trained BERT (Devlin et al., 2019) that uses a siamese and triplet network structure to derive semantically meaningful sentence embeddings that can be compared using cosine similarity. We explore this approach by encoding the articles with SBERT and and using the cosine similarity of articles pairs as the predicted *Overall* score. For these experiments, we used the default hyperparameters provided by Reimers and Gurevych (2019).

Similarity based We first concatenate the title and the textual content of each article, and due to the multilingual characteristic of the data, we encode the textual sequence with a pre-trained multilingual SBERT model and compute the Pearson correlation between the cosine similarity of these sentence embeddings and the gold labels, results presented in Table 2 (1a). Then, we experiment with machine translating all the non-English articles to English using Google Translate and use an English SBERT model. The results are presented in Table 2 (1b).

Regression based We fine-tune the SBERT model on the multilingual pairs, results presented in Table 2 (1c) and on the machine-translated articles, results presented in Table 2 (1d). For fine-tuning, we use only the *Overall* score as the target similarity score. Since the similarity scores provided in the training data are in the range [1-4] from *most to least* similar, we normalize the *Overall* scores (the scores provided by cosine similarity are in the $[0, 1]$ range from *least to most* similar.

3.2 Image Similarity & Regression

We download the images from the *top_image*, and as we can see in Table 1, out of 9,928 articles (4,964 pairs) in the train set, only 6,755 articles (68%) had a viable image. Out of 9,804 articles in the evaluation set, only 7,569 had a viable image (77%). For both, only around 60% of the articles had an image that could be used. Moreover, only around half of the pairs in both sets have representative images for both articles. Nonetheless, we attempt using

[1]For both sets, we were able to download around 98% of the articles.

[2]Our code is available at `https://github.com/bkolosk1/semeval-2022-MNS`

[3]All models are available at `https://huggingface.co/`.

Model		Pearson-r
Semantic Textual Similarity & Regression		
(1a) SBERT (PARAPHRASE-MULTILINGUAL-MPNET)	Similarity	0.6713
(1b) SBERT (ALL-MPNET) - Google Translate	Similarity	0.7139
(1c) SBERT (PARAPHRASE-MULTILINGUAL-MPNET)	Regression	0.7396
(1d) SBERT (ALL-MPNET) - Google Translate	**Regression**	**0.7835**
Image Similarity & Regression		
(2a) Images (CLIP-VIT-PATCH32)	Similarity	0.2991
(2b) Cross-images (CLIP-VIT-PATCH32)	Similarity	0.2607
(2c) Images (CLIP-VIT-PATCH32)	Regression	0.1043
(2d) Images (VIT-LARGE-PATCH32)	Regression	0.1124
Knowledge Graph Similarity & Regression		
(3a) KGm+LSA+SBERT (DISTILBERT+XLM-ROBERTA+ROBERTA)	Similarity	0.7128
(3b) KGm+LSA+SBERT (DISTILBERT)	Regression	0.5134
Text & Image Regression		
(4a) Text+metadata (XLM-ROBERTA-LARGE)	Regression	0.7773
(4b) Text+metadata+images (XLM-ROBERTA-BASE+CLIP-VIT-PATCH32)	Regression	0.7020
(4c) Text+metadata+images (XLM-ROBERTA-LARGE+VIT-LARGE-PATCH32)	Regression	0.7335

Table 2: Correlation between similarity scores from the proposed models and the *Overall* score.

them in our approaches. We experiment with two recent pre-trained models, CLIP (Radford et al., 2021) and ViT (Dosovitskiy et al., 2020).

Similarity based We obtain the image embeddings with CLIP, compute the cosine similarity between the paired images, and report the Pearson correlation between the obtained similarities and the gold labels. The results are presented in Table 2 (2a). For the missing images, we assign the default cosine similarity of 0.5. We also experiment with an alternative strategy, which takes advantage of the fact that CLIP is a multimodal model and produces images and text embeddings in the same space, *Cross-images*. In this strategy, we compute all possible similarities between data points: image-to-image, text-to-text, and image-to-text. In the best case, when both images are available, this results in a total of four similarity values. In the worst case scenario, when no images are present, only the similarity between texts is used. If only one image is available, the strategy results in two similarities: text-to-image and text-to-text. The final score is obtained by averaging the similarities available. Surprisingly, this strategy works slightly worse than an approach based solely on images, as can be seen in Table 2 (2b).

Regression based This method is detailed in Section 3.4. The results are presented in Table 2 (2c and 2d).

3.3 Knowledge Graph Similarity & Regression

We use the Wikidata5m (Wang et al., 2021) knowledge graph (KG) to retrieve knowledge-based features as used by Koloski et al. (2022). Similarly, we exploit six different knowledge graph embeddings: transE (Bordes et al., 2013), rotatE (Sun et al., 2019), complEx (Trouillon et al., 2016), distmult (Yang et al., 2015), simplE (Kazemi and Poole, 2018), and quate (Zhang et al., 2019). We use GraphVite (Zhu et al., 2019), a system for training node embeddings, pre-trained on aforementioned embeddings of the Wikidata KG. For these experiments, we use the translated articles. We concatenate the title and text of the articles to search n-grams of sizes 1, 2, and 3, as potential concepts appearing in the KG. After extracting potential candidates, we extract the embeddings of the candidates from the KG. In addition, we generate latent semantic analysis (LSA), SBERT and stats representations as done by Koloski et al. (2021). The results are in Table 2 (3a and 3b).

Similarity based First, we generate all ten feature spaces. Next, we generate combinations of feature spaces (1,024 combinations in total), we concatenate and normalize them (KGm). Finally, we find thresholds to estimate the similarity scores, with respect to the *Overall* label. Our best results are presented in Table 2 (3a).

Regression based We utilize all six of the aforementioned KG representations, LSA and DistilBERT (Sanh et al., 2019) SBERT representations. Next, we use a singular value decomposition (SVD) to generate a new latent space of the devised features and we proceed to train a feed-forward neural network on the whole target space. Our best results are presented in Table 2 (3b).

3.4 Text & Image Regression Models

We also propose a classical approach that considers the task of finding the similarity between two articles by considering it as a regression task, and by predicting the similarity for the *Overall* score. This approach consists of a pre-trained and fine-tuned language model (BERT (Devlin et al., 2019) pre-trained on multilingual data). Because these models expect input data in a specific format, we need a special token, [SEP] or <sep>, to mark the end of a sentence or the separation between two sentences, and [CLS] or </s>, at the beginning of a text generally used for classification or regression tasks.

Regression based After the pair of articles are tokenized and together encoded with [CLS] at the start and then separated by [SEP], they are passed through the encoder. Similarly, images are passed through a ViT encoder. For the missing images, we generate a *fake* white image. The BERT output token representations are afterward concatenated with the [CLS] representation and Vit output image representation followed by a linear layer for regression. The learning of the model is conducted end-to-end by optimizing an objective corresponding to *Overall* prediction. For these experiments, we utilized AdamW (Kingma and Ba, 2014) with a learning rate of 1×10^{-5} for 2 epochs with mean squared error (MSE) loss. We also consider a maximum sentence length of 512 (the maximum possible accepted by BERT or RoBERTa). These results are presented in Table 2 (from 4a to 4d).

4 Error Analysis

Semantic Textual Similarity We can substantially improve the English-only model (1d) for STS by fine-tuning not just with monolingual English pairs from the training data but by using all the machine-translated pairs. However, we observe some cases where our best performing fine-tuned model is misled by similar turns of phrase even if the article pair covers different events. We show extracts from an article pair in Table 3 that covers a fire and a traffic accident, respectively. The gold *Overall* score for this pair is 4.0 (very dissimilar) but our best-performing model scores it at 3.1 (somewhat dissimilar) due to the similar phrasing that opens the articles and the mention of the same named entities.

Article1	**Article2**
1492472369 (EN): **At least one person has been confirmed dead**, following Saturday's fire that gutted the Mgbuka Obosi Spare Parts Market in Idemili North **Local Government Area of Anambra** . . . Mr **Edwin Okadigbo**, the Public Relations Officer of the **Nigeria Security and Civil Defence Corps (NSCDC)**, Anambra command . . .	1530831511 (EN): **At least, one person has been confirmed dead** . . . in a road mishap that involved a commercial bus and a motorcycle in Mbosi junction, Ihiala **Local Government Area of Anambra** State on Tuesday . . . Spokesperson of the **Nigeria Security and Civil Defence Corps, NSCDC** in Anambra State, **Edwin Okadigbo** said preliminary . . .

Table 3: Extracts from an article pair with a predicted Overall similarity score of 3.159 by SBERT translated model (1d) and a gold-standard score is 4.0. We highlight similar terms are in bold.

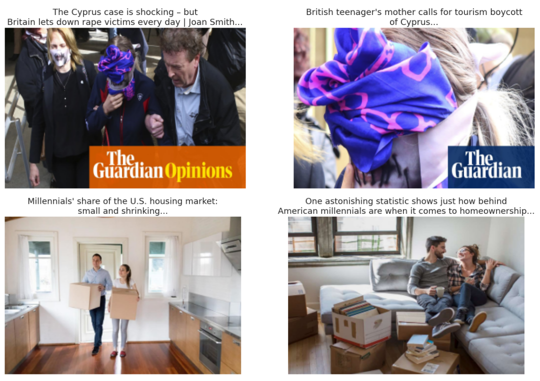

The Cyprus case is shocking – but Britain lets down rape victims every day | Joan Smith...
The Guardian Opinions
British teenager's mother calls for tourism boycott of Cyprus...
The Guardian
Millennials' share of the U.S. housing market: small and shrinking...
One astonishing statistic shows just how behind American millennials are when it comes to homeownership...

Figure 1: Two pairs of similar English articles (gold score of 1.0 for both) correctly predicted by the image-based model (1.28 & 1.0), and incorrectly predicted by SBERT (1.83 & 1.63).

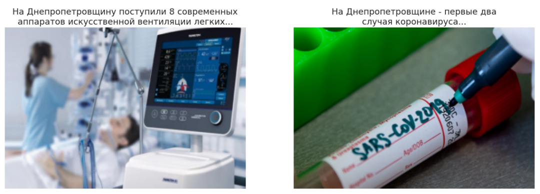

На Днепропетровщину поступили 8 современных аппаратов искусственной вентиляции легких...
На Днепропетровщине - первые два случая коронавируса...
SARS-CoV-2019

Figure 2: A pair of marginally similar Russian articles (gold score of 2.0), which is an unseen language during training, correctly predicted by the image-based model (1.64), and incorrectly predicted by SBERT (2.94).

Image Similarity & Regression We analyze the scores predicted by two textual-based methods, (1d) SBERT with the best scores when using only images (2a). Out of 4,902 pairs in the evaluation

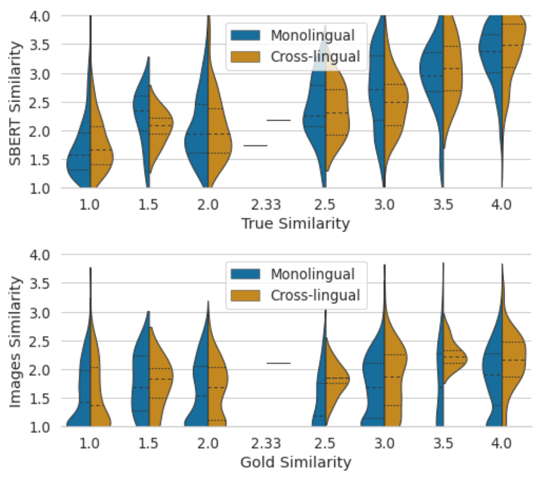


Figure 3: Similarity scores for the article pairs with available images for the image-based model, Images (2a) and Text+metadata (4c).

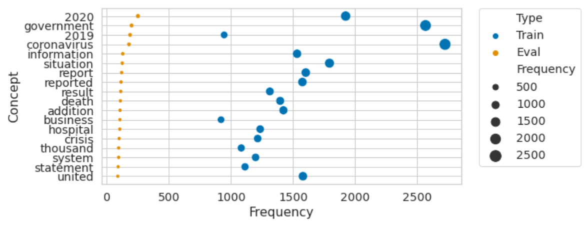


Figure 4: Distribution of KG concepts in the train and evaluation sets.

set (Table 1), only 2,009 have representative images for both news articles. Thus, we look closer at the predictions for these pairs and notice that 13% of them (262 pairs) are correctly predicted by (2a), and not by (1d), all of these being images with either faces or clearly identifiable texts or text boxes, as shown in Figure 1 for two pairs of English articles. We also give an example where this model is able to better distinguish the similarity between two articles in an unseen language (Russian) in Figure 2, where the articles discuss the same topic but describe different events.

Knowledge Graph Similarity & Regression We analyze the representations of articles based on the number of concepts retrieved from the WikiData5m. The most frequent concepts include entities such as *government, coronavirus, epidemic, report, information, death*, and *economy*, showing us that most of the articles report about the COVID-19 pandemic. The distribution of concepts per document is shown in Figure 4. Originally, the Wikidata5m KG is based only on English concepts. We notice a performance drop for the non-English articles, due to the translation to English, some original concepts are lost and replaced with another. For the training set, we retrieved an average of 55 concepts per article, while for the evaluation set we obtained 54 concepts per article. The lowest amount of retrieved concepts was one and the highest was 757.

Text & Image Regression Figure 3 presents the Images (2a) similarity scores in comparison with Text+metadata (4b) and Text+metadata+images (4d) similarity scores. First, the results for Text+metadata (4a) seem to be rather similarly distributed to those provided by SBERT, with a slight difference in the monolingual pairs with a gold score of 1.5, while SBERT generally predicts a similarity of 2.5. When using image representations, not surprisingly, we notice that the results for Images (2a) generally stay around an average of 2.0. This shows that having only around half of the train and evaluation sets with images is not enough to help distinguish between news articles.

SemEval-2022 Task 8 In the official SemEval-2022 Task 8, we ranked fifth in the overall team ranking for multilingual and cross-lingual results, and second in the English-only results. Our best performing model for both is SBERT finetuned on the STS task.

5 Conclusions

In this paper, we covered several techniques for assessing the similarity between multilingual and monolingual news articles in the context of SemEval-2022 Task 8 *Multilingual News Article Similarity*. We notice that, even if using images and knowledge graph representations give promising results, approaching STS with sentence embeddings is still unbeatable. However, images, being a language-agnostic medium, could be helpful if they depict people or text boxes. Future work could include an adaptable inclusion of images (for handling missing images) and the usage of multilingual knowledge graph representations.

Acknowledgements

This work has been supported by the European Union's Horizon 2020 research and innovation program under grants 770299 (NewsEye) and 825153 (EMBEDDIA), and by the ANNA and Termitrad projects funded by the Nouvelle-Aquitaine Region.

References

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4289–4300.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Boshko Koloski, Timen Stepišnik-Perdih, Senja Pollak, and Blaž Škrlj. 2021. Identification of covid-19 related fake news via neural stacking. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 177–188, Cham. Springer International Publishing.

Boshko Koloski, Timen Stepišnik Perdih, Marko Robnik-Šikonja, Senja Pollak, and Blaž Škrlj. 2022. Knowledge graph informed fake news classification via heterogeneous representation ensembles. *Neurocomputing*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

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

Marko Pranjić, Vid Podpečan, Marko Robnik-Šikonja, and Senja Pollak. 2020. Evaluation of related news recommendations using document similarity methods. In *Proceedings of the Conference on Language Technologies and Digital Humanities (JDTH2020)*, pages 81–86, Ljubljana, Slovenia. Inštitut za novejšo zgodovino.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY,* *USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Jiangnan Xia, Liwei Peng, and Luo Si. 2019. Structbert: Incorporating language structures into pre-training for deep language understanding. *arXiv preprint arXiv:1908.04577*.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2731–2741.

Zhaocheng Zhu, Shizhen Xu, Meng Qu, and Jian Tang. 2019. Graphvite: A high-performance cpu-gpu hybrid system for node embedding. In *The World Wide Web Conference*, pages 2494–2504. ACM.

HFL at SemEval-2022 Task 8: A Linguistics-inspired Regression Model with Data Augmentation for Multilingual News Similarity

Zihang Xu[†], **Ziqing Yang**[†], **Yiming Cui**[‡][†], **Zhigang Chen**[†]
[†]State Key Laboratory of Cognitive Intelligence, iFLYTEK Research, China
[‡]Research Center for SCIR, Harbin Institute of Technology, Harbin, China
[†]{zhxu13,zqyang5,ymcui,zgchen}@iflytek.com
[‡]ymcui@ir.hit.edu.cn

Abstract

This paper describes our system designed for SemEval-2022 Task 8: Multilingual News Article Similarity. We proposed a linguistics-inspired model trained with a few task-specific strategies. The main techniques of our system are: 1) data augmentation, 2) multi-label loss, 3) adapted R-Drop, 4) samples reconstruction with the head-tail combination. We also present a brief analysis of some negative methods like two-tower architecture. Our system ranked 1st on the leaderboard while achieving a Pearson's Correlation Coefficient of 0.818 on the official evaluation set.

1 Introduction

In Task 8 (Chen et al., 2022), we are expected to assess the similarity of pairs of multilingual news articles as shown in Table 1. Ten different languages are covered in this task, including Spanish, Italian, German, English, Chinese, Arabic, Polish, French, Turkish and Russian. Task 8 emphasizes more the events themselves described in the news rather than the style of writing or other subjective characteristics. Therefore, it is beneficial to improve the quality of clustering of news articles and to explore similar news coverage across different outlets or regions.

The foundation model (Bommasani et al., 2021) we choose is XLM-RoBERTa (XLM-R) (Conneau et al., 2019) which has been proved to be a powerful multilingual pre-trained language model compared with other models like mBERT (Devlin et al., 2018) and it can process all the languages existing in Task 8. Based on that, a great variety of strategies have been tested along with our exploration like data augmentation (DA), head-tail combination, multi-label loss, adapted R-Drop, etc.

Through this task, we realized the importance of data quality and efficient training schemes in such a cross-lingual setting. By struggling to improve the richness of the data and find out what methods are effective when training such a similarity assessment model, our system[1] ranked 1st in this competition.

2 Background

2.1 Dataset Description

There are 4,964 samples with 8 language pairs in the training set and the test set contains 4,593 samples in 18 different language pairs, the details of which are presented in Table 2. Due to some inaccessible URLs, the training set is slightly smaller than it should be (22 samples missing in total).

The similarity scores of pairs of articles in the provided dataset are rated on a 4-point scale (between 1 and 4) from most to least similar in 7 subdimensions, including *Geography*, *Entities*, *Time*, *Narrative*, *Overall*, *Style* and *Tone* (an example is provided in Appendix). However, only the predictions for *Overall* will be used to evaluate the performance of our systems.

2.2 Related Work

Research on text similarity always attracts people's eyes as it acts as the basis of quite a few NLP downstream tasks like information retrieval (Ponte and Croft, 2017). Previously, some methods based on statistics like BM25 (Trotman et al., 2014) and Edit Distance (Ristad and Yianilos, 1998) are used to evaluate the relevance between two texts but they do not work anymore in cross-lingual settings. Then, after dense word embedding in low dimensions like Word2Vec (Mikolov et al., 2013) was put forward, methods like calculating the cosine similarity (Rahutomo et al., 2012) with the sentence embedding based on each word embedding came into use. However, it is hard for these approaches to capture the latent meaning of the whole article precisely. Nowadays, depending on transformer-based

[1]Our codes are available at https://github.com/GeekDream-x/SemEval2022-Task8-TonyX

Key	Value
Pair_id **Lang1/Lang2**	1626170156_1623571850 de/en
News1	US-Bürgerrechtler verklagen Trump wegen Polizeieinsatzes. Der Einsatz am Montag sei gesetzwidrig gewesen, da die Demonstranten sich friedlich verhalten hätten, ... Tod des Afroamerikaners George Floyd bei einem brutalen Polizeieinsatz in Minneapolis ausgelöst worden. Im Zuge der Proteste kam es immer wieder zu Ausschreitungen.
News2	Joe Biden Addresses The Nation On Race And Trump's Attacks On Protesters Via the Washington Post: Seeking to console a nation riven by nights of violence with a promise to heal its racial wounds, ... — "I can't breathe" — as a mantra. Floyd, an unarmed black man, died after a police officer knelt on his neck in Minneapolis.
Scores	**Geography** 1.0 **Entities** 2.0 **Time** 1.0 **Narrative** 2.0 **Overall** 4.0 **Style** 2.0 **Tone** 1.0

Table 1: An example in the training set.

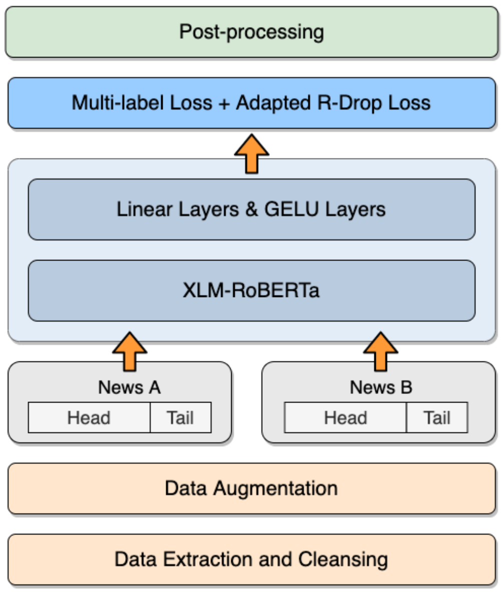

Post-processing
Multi-label Loss + Adapted R-Drop Loss
Linear Layers & GELU Layers
XLM-RoBERTa
News A
Head
Tail
News B
Head
Tail
Data Augmentation
Data Extraction and Cleansing

Figure 1: The overall framework of our system proposed for SemEval-2022 Task 8.

general pre-trained models are becoming the new paradigm and plenty of models for multilingual and cross-lingual settings have been proposed like mBERT (Devlin et al., 2018), ERNIE-M (Ouyang et al., 2020) and XLM-R (Conneau et al., 2019).

3 System Overview

Our baseline system is simply providing a pair of articles to XLM-R and regressing its output from [CLS] token to the manually annotated similarity score by training with Mean Squared Error (MSE). All the optimization methods discussed below are applied based on this architecture and the overall framework of our final system is illustrated in Figure 1. After training with all the positive strategies, we then made an ensemble of the best models on each fold for the final prediction.

3.1 Data Augmentation

In this task, we augmented the training data in two different ways and they will be introduced respectively in the following subsections.

3.1.1 Back Translation

It is clear from Table 2 that the original training set is not sufficient to train XLM-R, so we made use of back-translation to enrich it. As the English pairs account for the largest, we only paid attention to the non-English samples in this stage. Take the French samples for example, by calling Google Translation API[2], we translate the French articles to English and then translate the English texts back to French. As for the cross-lingual pairs with German and English, we only back-translate the German part and then combine it with the corresponding English part to form a new sample.

3.1.2 Translate Train

Another weakness of the original training set is the severe lack of some monolingual language pairs which exist in the test set but not in the training set like Chinese and quite a few cross-lingual language pairs like German to French. To deal with this problem, we planned to generate translate-train data to fill the gap.

In such semantic comprehension tasks, it is undoubted that the richer semantic information is, the better the model performance will be. Therefore, for maintaining the semantic richness to the largest extent, we made an arrangement for the construction of the translate-train set (details are provided in Table 3).

As the average quantity of non-English monolingual samples in the training set is 430, for the sake

[2]https://cloud.google.com/translate

	ar	de	en	es	fr	it	pl	ru	tr	zh	de-en	de-fr	de-pl	es-en	es-it	fr-pl	pl-en	zh-en	Total
Train	274	857	1787	567	72	0	349	0	462	0	574	0	0	0	0	0	0	0	4942
Test	298	611	236	243	111	442	224	287	275	769	190	116	35	498	320	11	64	223	4953
Train+DA	548	1714	1787	1134	461	586	689	401	924	800	1148	317	0	586	586	0	349	800	12830

Table 2: Data distribution in each set. Columns with one language (e.g. "zh") mean the two articles in a pair are in the same language. Columns with two languages (e.g. "zh-en") indicate the corresponding cross-lingual pairs.

Origin	Quantity	Target
en-en	401	ru-ru
	800	zh-zh / zh-en
	586	it-it / es-en / es-it
pl-pl	349	pl-en
de-en	317	de-fr / fr-fr

Table 3: Arrangement for the construction of translate-train set.

of balancing the whole dataset, we decided to round it down to 400 and let it be the number of translated samples for Russian (due to some precision issues, it became 401 accidentally). As we may know, Russian and English both belong to Indo-European Family (Fortson IV, 2011) while Chinese is a member of the Sino-Tibetan Family (Thurgood and LaPolla, 2016), which indicates that there are quite a lot of common characteristics between the two languages like syntactic structures and lexical analysis methods. So, the most English samples in the original training set would help more in understanding Russian instead of Chinese. Therefore, we decided to generate more Chinese pairs and here we just doubled the number for Russian. Furthermore, the English samples left were all used for generating samples in Italian and Spanish.

In order to improve the reusability of those samples newly translated already, some work on recombination among different languages pairs was done in this phase. For instance, translating German to English samples to French would let us get German to French samples in the meantime.

3.2 Head-tail Combination

There is no doubt that different types of texts have different features. As for news, the title tends to be the most informative place in each article since the authors need to use as few concise words as possible to let the readers know what happened in the story. Besides, we believe the head and tail parts of a news article provide much information as well as similar to the introduction and conclusion parts in a research paper. As the XLM-R is capable of processing 512 tokens in each sequence (a pair of articles) at most and the large majority of articles in the training set are much longer than 256 tokens (see Figure 2), we tried different truncation strategies to further boost the model performance.

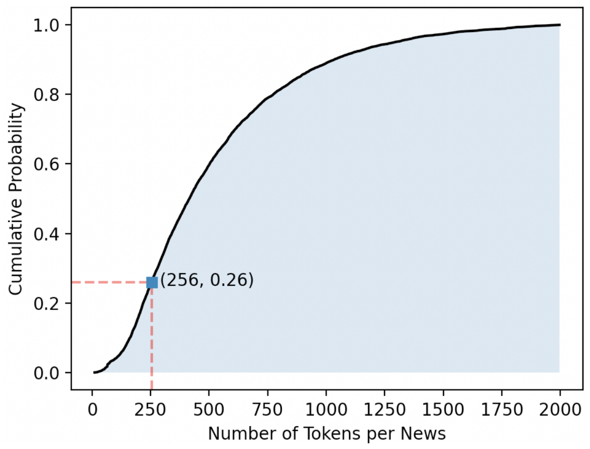


Figure 2: Cumulative probability distribution of article lengths in the training set.

3.3 Multi-label Loss

As introduced in Background, only the predictions for *Overall* will be used to evaluate, but the other 6 sub-dimensions are also probably helpful for assisting in building a better model. Consequently, we tried to assign various weights for *Overall* when calculating the loss while treating other sub-dimensions equally. For example, if the loss for *Overall* accounts for 40%, the percentages of the other six sub-dimensions are all 10% individually.

3.4 Adapted R-Drop

R-Drop is proved to be an effective regularization method based on dropout, by minimizing the KL-divergence of the output distributions of every two sub-models generated via dropout in model training (Liang et al., 2021). To better fit with this regression task, we replaced the KL-divergence loss with MSE loss (adapted R-Drop). Similarly, at each training step, we feed the samples through the forward pass of the network twice. Then, our adapted R-Drop method tries to regularize the model by minimizing the two predicted scores for the same sample, which is:

$$L_R^i = \text{MSE}(y_1^i, y_2^i)$$

where the y_1^i means the model output in the first forward pass for the i_{th} sample. With the basic MSE loss L_B of the two forward passes:

$$L_B^i = \frac{1}{2} \cdot (\text{MSE}(y_1^i, \hat{y}^i) + \text{MSE}(y_2^i, \hat{y}^i))$$

where the $\hat{y}^i$ is the label of the i_{th} sample, the final training target is minimizing L^i for i_{th} sample:

$$L^i = \alpha \cdot L_R^i + (1 - \alpha) \cdot L_B^i$$

where the α is the weight for the adapted R-Drop loss. Based on this introduction, it is easy to extend the formulas to those of forwarding three times.

3.5 Extra Linear Layers

In our baseline system, the prediction score is generated by passing the output of [CLS] token from XLM-R through a single linear layer with the size of (1024, 1). In other words, there are only 1024 parameters that are responsible for the regression from the sentence representation vector to the prediction score, which is probably beyond their power. Hence, we attempted to add a few more layers on top of the XLM-R.

3.6 Post-processing

Once getting the prediction scores, we further corrected some wrong numbers which were outside the expected range. As introduced in Section 2.1, the annotators annotated the similarity in the range (1, 4); consequently, we clipped the outliers.

4 Experimental setup

4.1 Dataset Split

Both the original training set and the training set with DA set were split into 10 subsets with no intersection by random sampling. All the experiments discussed in this paper were conducted with 10-fold cross-validation, and the results displayed are the averages. By using the cross-validation method (Browne, 2000), we could ensure the strategies applied will take a good effect on the final test set to the largest extent.

System	Pearson's CC
w/ data augmentation	
Baseline	83.49
+ DA	**85.86**
w/o data augmentation	
Baseline	84.94
+ Head-tail Combination	85.38
+ Multi-label Loss	85.33
+ Adapted R-Drop	**86.14**
+ Extra Linear Layers	85.50

Table 4: Best results with training methods we used.

4.2 Pre-processing

The news articles in all the data sets are released as URLs and the task organizers offer us a python script[3] which helps to download the pages. After downloading the original files in JSON format, we then extracted and combined "title" and "text" parts of each article and abandoned all other information like "description". Before starting training our model, apart from conducting data augmentation to the training set, we also cleaned the data and joined the head and tail parts of each article. During the process of cleaning, we mainly removed some dirty formatted data like URLs and file paths.

4.3 Evaluation Metrics

The evaluation metric for task 8 is the Pearson's Correlation Coefficient (Pearson's CC) which is a measure of linear correlation between two series of data with a range from -1 to 1 (from least to most correlated) (Stigler, 1989).

4.4 Others

Although hyper-parameters tuning is not a crucial point in our work, we tested a few values for several of them within a small range as they did have an influence on our decisions about how well a strategy worked (see Appendix). Additionally, to help readers replicate our experiments, the details of tools and libraries are provided (see Appendix).

5 Results

5.1 Overall Performance

Finally, our system got 0.818 on the evaluation set according to the official scoring system and

[3]https://github.com/euagendas/semeval_8_2022_ia_downloader

Head	Tail	Pearson's CC
256	0	84.94
200	56	**85.38**
128	128	85.21
56	200	84.53
0	256	78.85

Table 5: Results on different head-tail combinations.

ranked 1st. As results are shown in Table 4, all the strategies introduced in Section 3 turned out to have positive effects, and we will discuss the effect of the strategies mentioned individually in the following subsections. For convenience, all the results from our experiments are multiplied by 100.

5.2 Data Augmentation

To find out whether the augmented data was helpful or not, we trained our system on the original training set and the training set with DA respectively (samples used for testing were removed in both of them), then tested it on each fold of the DA set. In experiments on other strategies, we trained and tested our system on the original training set. And this is the difference between the two baselines in Table 4.

Without any surprise, an evident increase is observed from the results displayed in the top part of Table 4, based on which we could make a conclusion that a more abundant training set is definitely beneficial for building a strong system.

5.3 Head-tail Combination

As introduced in Section 3.2, we realized the importance of the head and tail parts of the news articles. However, we cannot determine which part should be paid more attention to heuristically. So, we tried on different ratios of head-tail combination and the results are enumerated in Table 5. Clearly, the head part plays a much more important role by comparing the first and last rows where only either of them are used. However, from the middle three rows where the head and tail parts are combined, it is indicated that the tail part also benefits the whole model performance.

5.4 Multi-label Loss

As discussed in Section 3.3, we used other 6 dimensions and assigned a few different values for the weight of *Overall* from 0% to 100%. It is explicitly

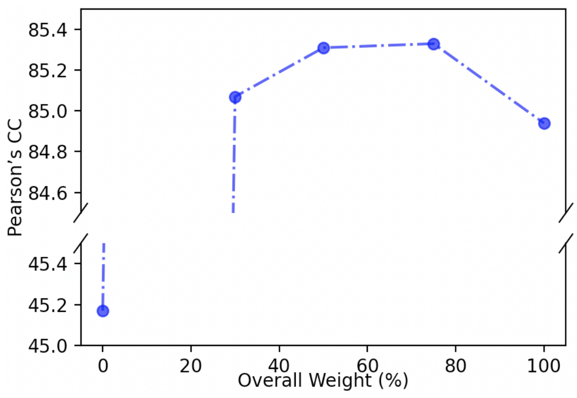


Figure 3: Results on training with multi-label loss.

observed from Figure 3 that there is an overwhelming increase followed by a slight drop while the weight of *Overall* rises gradually. Based on the experiment results, we believe that *Overall* is of the greatest importance to this task, yet the other 6 subdimensions also have a positive effect on achieving a better similarity assessment system.

5.5 Adapted R-drop

As described in Section 3.4, the training loss in our system is composed of both the loss between predictions and labels and the loss between the predictions from different forwarding processes. Here, we explored forwarding once to three times while changing the weight of adapted R-Drop loss.

Apparently, there is a phenomenon from Figure 4 that no matter how large the weight of R-Drop loss is, the more forwarding times are, the better results we will achieve. However, by comparing the results between forwarding once and twice and the results between forwarding twice and three times, we speculate that there is a marginal utility (Kauder, 2015) on this trick, which means the additional benefit from this method will decrease while simply continuing increasing the number of forwarding.

5.6 Extra Linear Layers

During the process of exploration in this direction, we attempted to add 2 or 3 extra linear layers to test if it worked. In the 2-layer setting, the sizes of the layers are (1024, 512) and (512, 1) while sizes composed of (1024, 768), (768, 256) and (256, 1) are prepared for the 3-layer setting. Two sets of experiments were conducted in both settings about whether to put an activation layer (we used GELU (Hendrycks and Gimpel, 2016) here) between adjacent linear layers or not.

It can be observed from Table 6 that there is only

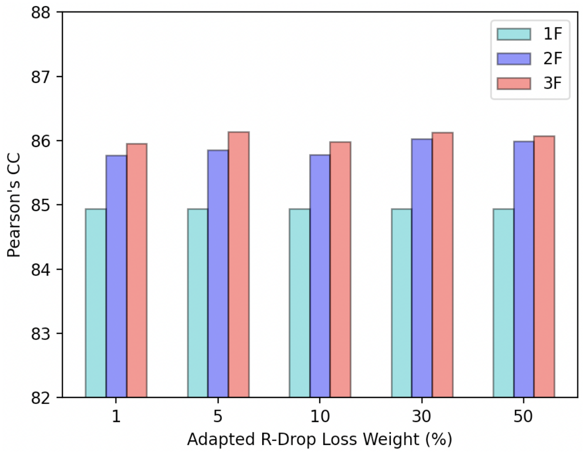


Figure 4: Results about adapted R-Drop (RD) in different settings. "2F" means forwarding twice.

System	Pearson's CC
1-layer	84.94
2-layer	85.46
+ activation	**85.50**
3-layer	85.32
+ activation	85.23

Table 6: Results on different extra layers.

a quite small difference that caused by activation layers in each setting and the effect of that is not always positive. In addition, by comparing the results from different settings, we could draw a conclusion that more parameters did help to boost the system performance even if the benefit does not show linear growth.

5.7 Negative Results

Aside from the strategies discussed above, several tricks that were attempted to deploy in our system as well turned out to be meaningless or had a bad effect on the model performance. For example, we tried to use a pooling vector (max or mean) or the fusion of [CLS] vectors from different layers in XLM-R as the article representation. We also tried to expand the length of sentences that XLM-R could process to 1024 tokens by modifying its position embedding matrix by means of adding a random shift vector after each vector or just randomly initializing the latter part of the learnable expanded matrix. Each negative strategy mentioned above brought approximately at least 2 points drop on the Pearson's CC. Furthermore, unsurprisingly, a two-tower architecture where each shared-parameter model processed each article in a pair led to scores of points decrease, which reflected the importance of semantic interaction via the attention mechanism inside the model.

en	de	es	pl	tr
87.19	84.96	86.64	75.29	83.54
ar	**ru**	**zh**	**fr**	**it**
79.42	78.47	76.78	86.53	86.17
es-en	**de-en**	**pl-en**	**zh-en**	
86.35	85.98	88.18	81.00	
es-it	**de-fr**	**de-pl**	**fr-pl**	
81.97	68.89	64.31	82.68	

Table 7: Individual results of all language pairs in our best submission.

5.8 Error Analysis

After the evaluation phase ended, the evaluation data with labels were provided and we also checked the system performance on different language pairs individually. The details of our best submission are presented in Table 7. It is obvious that the model tends to perform worse on the language pairs which are rare or absent from the training set like German to Polish (only 64.31). Interestingly, although having seen monolingual samples in Polish and related cross-lingual data, the system still behaves badly on Polish monolingual data (just slightly over 75), which is probably due to its complicated lexical variation and grammar rules (Smoczynska, 2017).

6 Conclusion

By deploying various optimization methods, including data augmentation, head-tail combination, multi-label loss, adapted R-Drop and adding extra linear layers, we built a relatively strong system for assessing the similarity between a pair of news articles in multilingual and cross-lingual settings and ranked 1st in the competition with a Pearson's CC of 0.818 on the official evaluation set.

In the future, apart from enriching the training data, we are also supposed to analyze the languages individually and try to leverage the exclusive rules or features of each language rather than relying too heavily on general pre-trained models to further boost the model performance, especially on those minority languages.

References

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Michael W Browne. 2000. Cross-validation methods. *Journal of mathematical psychology*, 44(1):108–132.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Benjamin W Fortson IV. 2011. *Indo-European language and culture: An introduction*. John Wiley & Sons.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Emil Kauder. 2015. *History of marginal utility theory*. Princeton University Press.

Xiaobo* Liang, Lijun* Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *NeurIPS*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-m: enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. *arXiv preprint arXiv:2012.15674*.

Jay M Ponte and W Bruce Croft. 2017. A language modeling approach to information retrieval. In *ACM SIGIR Forum*, volume 51, pages 202–208. ACM New York, NY, USA.

Faisal Rahutomo, Teruaki Kitasuka, and Masayoshi Aritsugi. 2012. Semantic cosine similarity. In *The 7th International Student Conference on Advanced Science and Technology ICAST*, volume 4, page 1.

Hyperparameter	**Range/Value**
Epoch	20 ~ 30
Batch Size	32
Weight Decay	1e-4
Warm-up Rate	0.1
Learning Rate	5e-6 ~ 3e-5
Overall Weight	0 ~ 1
Adapted R-Drop Weight	0.01 ~ 0.5

Table 8: Main hyper-parameters tuned in our system.

Tools & Libraries	**Version**
NumPy	1.21.2
pandas	1.2.4
Python	3.7.10
PyTorch	1.9.0
Transformers	4.5.1
semeval_8_2022_ia_downloader	0.1.7

Table 9: Main tools and libraries used in our system.

Eric Sven Ristad and Peter N Yianilos. 1998. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532.

Magdalena Smoczynska. 2017. The acquisition of polish. In *The crosslinguistic study of language acquisition*, pages 595–686. Psychology Press.

Stephen M Stigler. 1989. Francis galton's account of the invention of correlation. *Statistical Science*, pages 73–79.

Graham Thurgood and Randy J LaPolla. 2016. *The sino-tibetan languages*. Taylor & Francis.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of the 2014 Australasian Document Computing Symposium*, pages 58–65.

A Appendix

Table 8 and Table 9 provide the details about the corresponding hyper-parameters and libraries respectively, which are beneficial to help replicate our experiments.

GateNLP-UShef at SemEval-2022 Task 8: Entity-Enriched Siamese Transformer for Multilingual News Article Similarity

Iknoor Singh, Yue Li, Melissa Thong and **Carolina Scarton**
Department of Computer Science, University of Sheffield (UK)
{i.singh, yli381, mlthong1, c.scarton}@sheffield.ac.uk

Abstract

This paper describes the second-placed system on the leaderboard of SemEval-2022 Task 8: Multilingual News Article Similarity. We propose an entity-enriched Siamese Transformer which computes news article similarity based on different sub-dimensions, such as the shared narrative, entities, location and time of the event discussed in the news article. Our system exploits a Siamese network architecture using a Transformer encoder to learn document-level representations for the purpose of capturing the narrative together with the auxiliary entity-based features extracted from the news articles. The intuition behind using all these features together is to capture the similarity between news articles at different granularity levels and to assess the extent to which different news outlets write about "the same events". Our experimental results and detailed ablation study demonstrate the effectiveness and the validity of our proposed method.

1 Introduction

News article similarity measures could facilitate various important tasks such as the clustering of news (Montalvo et al., 2007; Azzopardi and Staff, 2012), duplicate news detection (Alonso et al., 2013; Gibson et al., 2008; Singh et al., 2021a; Theobald et al., 2008), fact-checking (Hassan et al., 2017; Jiang et al., 2021) and tracking of the spread of news (Zhai and Shah, 2005). SemEval-2022 task 8 (Chen et al., 2022) assesses the similarity between news articles in terms of the real world happenings. Therefore, it mainly considers the location, time, entities and narratives, instead of writing style, political spin, or tone of the article. The training set comprises of monolingual and cross-lingual news articles pairs in 10 different languages.

The main contributions in this paper are: 1) We propose an entity-enriched Siamese Transformer whose general idea is to enable the model to explicitly learn from entity features (geolocation, organization, date and quantity) that are crucial to determine the similarity between news events but difficult to extract directly from the language models during fine-tuning. 2) We explore different data augmentation approaches through semi-supervised learning in order to tackle the imbalanced data problem. 3) We compare our proposed model with strong baselines and conduct an ablation study to analyse the contribution of each component in our model. We also present an error analysis at the end of this paper. Our best system which exploits Language Agnostic BERT Sentence Representations (Feng et al., 2020) ranks 2nd in the competition.

2 Background

The goal of the task is to predict similarity scores ranging from 1 (most similar) to 4 (least similar) for a give news article pair. The training data consists of 4,964 article pairs in seven distinct languages (English, German, Spanish, Turkish, Polish, Arabic and French), with seven groups of monolingual pairs and only one group of cross-lingual pair (German and English). The 4,902 pairs in test data feature three more languages (Chinese, Russian and Italian) and seven groups of new cross-lingual pairs. Appendix A.1 (Figure 3) shows the distribution of the train and test data. Most of the news pairs exhibit low level of similarity. However, the test set contains more pairs with the same news stories. The similarity based on location, time, entities, narratives, style and tone is also annotated respectively and given for the training data.

Previous studies on textual similarity have investigated different approaches based on corpus, knowledge or deep neural network (Chandrasekaran and Mago, 2021; Gomaa et al., 2013; Devlin et al., 2019; Reimers and Gurevych, 2019; Thakur et al., 2021a; Singh et al., 2021b), but they mainly experiment with short text pairs (e.g., the Semantic Textual Similarity (STS) benchmark (Cer et al., 2017), which is a sentence-level task). Measurement of the document similarity is arguably more challenging than that of short text since the information in the document is sparse and the model is easier to be misled by non-essential content. This paper tries to tackle the above mentioned challenges to build robust models for document similarity (Section 3).

3 System Overview

We propose an entity-enriched Siamese Transformer model which exploits multiple multilingual pre-trained Transformers (MPT) for multilingual news document similarity. The main subdimensions of similarity as annotated by the annotators are geolocation, time, shared entities and the shared narratives between the news articles (Chen et al., 2022)[1]. All this information is encoded in our model to capture different dimensions of news articles. Figure 1 shows the architecture of our proposed model and section 3.1 discusses its details.

3.1 Model Details

Reimers and Gurevych (2019) propose a Siamese and triplet network training methodology for the BERT-based (Devlin et al., 2019) models to derive semantically meaningful sentence embeddings. Instead of just learning the sentence representations, we use the Siamese (or bi-encoder) network architecture to learn document-level representations for the purpose of capturing the narrative of the article.

In this work, we test multiple MPT models as the backbone Transformer (Vaswani et al., 2017) encoder for our Siamese network and train the model using a regression objective function (Reimers and Gurevych, 2019). In this, the model is trained such that it forces contextual representations from similar documents close to each other in contrast to dissimilar documents by reducing the mean squared error loss between the overall similarity score and the cosine similarity between document representations. First, we concatenate title and news article text together to generate a single document. For the documents that lie beyond the maximum input sequence length for the model, we truncate their length and use the initial part of the document as a proxy for its fundamental narrative that is usually stated in the title and the lead paragraphs of news articles.

In an effort to explicitly capture other dimensions of the news article such as mentions of geolocation, event date, organisations and other named entities, we use SpaCy NER[2] to extract entities from the documents. For the non-English documents, we use their machine translated English versions to extract the entities (Fan et al., 2021).[3] All the extracted entities are aggregated into four different types of entities. The entity types along with their SpaCy labels (in brackets) are:

- **Geolocation (GEO)**: Location (LOC) and geopolitical entities (GPE) which includes mentions of countries, cities, states etc.
- **Organization (ORG)**: Name of organization (ORG), person (PERSON) and mentions of other important named entities (FAC, EVENT, NORP, PRODUCT, WORK_OF_ART) which can include names of airports, highways, sports events, religious or political groups etc.
- **Date (DATE)**: Date of publish from training set and date (DATE) and time (TIME) extracted from the news articles.
- **Quantity (QTY)**: Mentions of numerical quantity (QUANTITY) which can include both ordinal and cardinal measurements (ORDINAL, CARDINAL).

After extracting the list of all the above mentioned entities for both the documents, we lowercase all the entities and compute the cosine similarity to get a single similarity score which we use as feature for further training. The complete algorithm to compute similarity between entities is presented at Algorithm 1. We use the cosine scores of these four additional entity features along with the shared narrative score from Siamese Transformer to train a multilayer perceptron (MLP) with sigmoid activation to get the final news article similarity score. Overall, only the Siamese Transformer model is differentiable and all other are non-differentiable static features before the MLP layer. It is worth noting that both the Siamese Transformer and the MLP layer are trained separately and not jointly. The intuition behind using all these features together is to capture the similarity between news articles with different granularity and to assess the extent to which outlets write about same events.

[1]https://competitions.codalab.org/competitions/33835

[2]Pre-trained model en_core_web_trf is used for our experiments. Ref. https://spacy.io/models

[3]Pre-trained model m2m_100_418M is used for our experiments. Ref. https://github.com/pytorch/fairseq

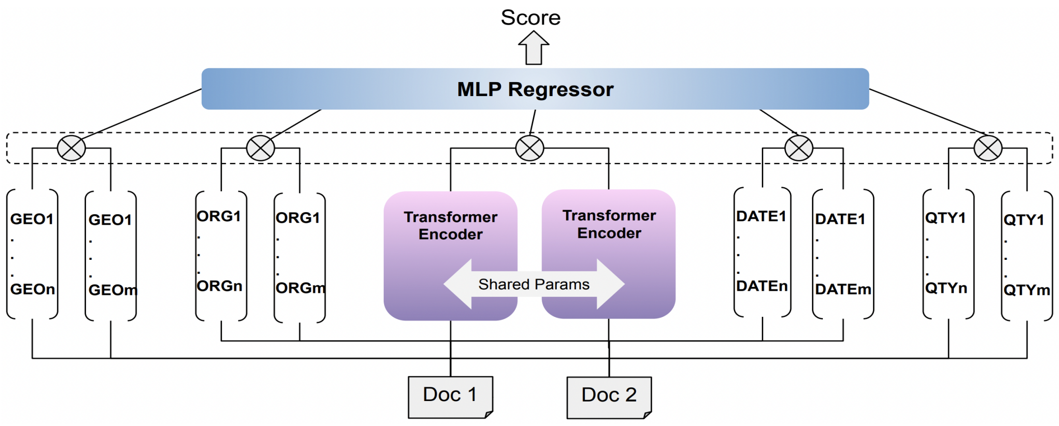


Figure 1: Entity-enriched Siamese Transformer model

Algorithm 1 Similarity between list of entities

```
function SIMILARITY($A, B$)
                    ▷ where A and B are list of entities
    Entities ⟵ Union (A, B)
    Product ⟵ Sum(A.count(ent)*B.count(ent) for ent in Entities)
    MagnitudeA ⟵ Sqrt(Sum(A.count(ent)$^2$ for ent in Entities))
    MagnitudeB ⟵ Sqrt(Sum(B.count(ent)$^2$ for ent in Entities))
    Score ⟵ Product / ( MagnitudeA * MagnitudeB )
    return Score
end function
```

3.2 Semi-Supervised Learning

One of the issues with the training dataset is its imbalanced nature, i.e. there are considerably more document pairs with low similarity scores as compared to pairs with high similarity scores (Figure 3). In order to upsample the instances with high similarity, we first augment the training set and then adopt a semi-supervised training methodology to train the model. For this, we first explore randomly sampling document pairs and label them using our previous best model trained on the training set. However, we found that random sampling does not generate good quality augmented training data as most of the generated pairs still lie in the similarity range of [0.0,0.2]. Therefore, we employ the augmentation strategy similar to one proposed by Thakur et al. (2021b), i.e. BM25 sampling, for which we use the Elasticsearch[4] implementation of BM25 to generate augmented article pairs. However, our method is slightly different where we use the title of a news article as a query for retrieving other news article to get the document pairs. This helped the generation of documents pairs with similarity score that are more evenly distributed. A total of 59,943 additional document pairs were generated using the BM25 sampling and 2,845 document pairs were generated using machine translation. The complete details of the generation of augmented data is mentioned in Appendix A.2.

Semi-supervised learning (Zhu and Goldberg, 2009) is a widely known training paradigm where a model is first trained on a human labelled dataset and the model is further used to extend the training set by automatically annotating the unlabelled dataset. Following previous studies (Thakur et al., 2021b; Jurkiewicz et al., 2020), we initially start with training on the original training set and then for all the generated unlabelled document pairs, we use the previously trained model for inference to get the similarity scores for the new synthetic document pairs. Finally, we train our entity-enriched Siamese Transformer in a semi-supervised fashion on both the complete augmented training set.

4 Experimental Setup

The training set contain 4,964 article pairs, however, after removing pairs with empty documents or if one of the document has less than ten tokens, we get a total of 4,544 article pairs for training. For the experiments, the training set is split into train and development sets using a 80:20 split ratio. The data is split in a stratified fashion to keep equal proportion of all languages, except Arabic document pairs that only appear in the development set, keeping it as a unseen language for the training set We get a total of 3,635 document pairs for the training

[4]https://www.elastic.co/elasticsearch/

set and 909 document pairs for the development set. In the end, we train models on the complete train and development set for final evaluation on the test set consisting of 4,953 document pairs.

The training set has different types of labels which take into account different aspects of similarity among news articles (e.g., geolocation, narrative) (Chen et al., 2022). For our experiments, we use the "Overall" similarity as the true labels. However, the overall similarity label is in a range [1,4] where 1 signifies highest similarity and 4 means lowest similarity. In order to normalise these overall similarity label, we subtract all values from 4 to bring values in range [0,3], followed by min-max normalisation. The formula for this is as follows,

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} = \frac{4 - x}{3} \quad (1)$$

where x denotes the overall similarity value and x_{norm} is the normalised value in range [0,1] from least similar to most similar. We use these as labels for training our models.

Since there are no baselines provided by task organisers, we use strong baseline methods for comparison. First, we use separate entity-based cosine similarity scores (Section 3.1) to find the Pearson correlation with the official test set labels. Second, we use Siamese multilingual BERT (Devlin et al., 2019), XLM-RoBERTa base variant (Conneau et al., 2020), Universal Sentence Encoder (USE)[5] (Yang et al., 2020) and LaBSE (Feng et al., 2020). Third, we also use the pre-trained MPNet[6] (Song et al., 2020) and fine-tuned it on the machine translated English training set. Please refer to Appendix A.3 for training details and the hyper-parameters used in our experiments.

5 Results and Discussion

Table 1 shows the Pearson correlation coefficient scores on the official test data. The first part of the Table 1 presents the results of the baseline methods and the second part presents the results of our proposed model. For baselines, we find that if we just consider the entity-based features without training, organisation (ORG) achieves the highest Pearson correlation with the official test set. Furthermore, for the trained Siamese Transformer models, the LaBSE works best and the XLM-RoBERTa achieves the lowest score of around 0.70.

The second part of Table 1 shows the results of our entity-enriched Siamese Transformer trained on the augmented data using the semi-supervised learning paradigm. Training the Siamese Transformer on the augmented data brings significant improvements for all the models except USE when compared with the ones trained on the training set without augmentation. A potential explanation for LaBSE's impressive performance is the MLM and TLM pre-training objective on a dataset of 109 different languages, followed by training on a translation ranking task (Feng et al., 2020). We also test the Siamese Transformer both with and without the entity-enrichment in order to study how significant the improvements are statistically with entity features. Furthermore, we use Williams test (Graham and Baldwin, 2014) to test the statistical significance of increased correlation with the added auxiliary entity features. As shown in the Table 1, for all Siamese Transformers trained on the augmented data, entity-enrichment brings improvements in results to a statistically significant degree (p-value$<$0.01). We also find that the results of entity-enriched Siamese Transformers are statistically significant (p-value$<$0.01) when compared with the baseline Siamese Transformers for all the models. Overall, our entity-enriched Siamese LaBSE model trained on augmented data achieves the highest Pearson correlation of 0.80164.

Figure 2 presents the detailed analysis of the results for the entity-enriched Siamese LaBSE trained on augmented data model (best model). We observe that model performance varies in different language settings. The model performs the worst over German-French pairs (0.619), while it achieves the highest score on French-Polish articles (0.866). Overall, the model performs better on mono-lingual pairs than on cross-lingual pairs.

We further analyse the "serious mistakes" that our best model tends to make: giving high/low similarity scores for dissimilar/similar article pairs. We identify these instances based on the difference between the true similarity score and the predicted similarity score from the model. We examine the cases when the absolute values of the differences are larger than 2.0 (total 46 samples found), and observe that: 1) News pairs that cover different

[5]We employ a custom variant of USE model which supports 50+ languages. Ref. https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2

[6]We employ a custom MPNet model which already fine-tuned on a 1 billion sentence pairs dataset with contrastive training objective. Ref. https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Experiments	Test Results	P-value
Baselines		
Geolocation (GEO)	0.26696	
Organisation (ORG)	0.45503	
Date (DATE)	0.43482	
Quantity (QTY)	0.35791	
Siamese mBERT	0.73069	
Siamese XLM-RoBERTa	0.70775	
Siamese USE	0.73324	
Siamese MPNet (English MT)	0.74536	
Siamese LaBSE	0.79187	
Proposed Method		
Siamese mBERT (Augmented data)	0.76649	
Entity-enriched Siamese mBERT (Augmented data)	0.76771	<0.0001
Siamese XLM-RoBERTa (Augmented data)	0.77669	
Entity-enriched Siamese XLM-RoBERTa (Augmented data)	0.77781	<0.0001
Siamese USE (Augmented data)	0.73320	
Entity-enriched Siamese USE (Augmented data)	0.73590	<0.0001
Siamese LaBSE (Augmented data)	0.80089	
Entity-enriched Siamese LaBSE (Augmented data)	**0.80164**	0.0022

Table 1: Pearson correlation coefficient score on the official test data. The best performance is in bold.

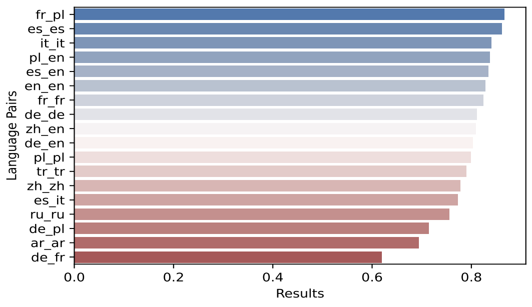


Figure 2: Performance over different language pairs in official test data. Average score of mono-lingual and cross-lingual pairs is 0.79833 and 0.78205 respectively.

stories around the same entities or topics are more challenging for the model. For example, the model outputs a similarity score of 1.77 for the following dissimilar news pair: an article about COVID-19 quarantine policy in Azerbaijan and another one regarding sanitary rules for preventing COVID-19 in Azerbaijan. 2) Web scraping could introduce noises in the model evaluation and obscure the true performance. We find that the scraping tool provided by the organizer fails to extract the real news content for all the Arabic news from Ahewar news website. The banner of the news website is returned instead. If we ignore the 25 pairs involving news from this website in the official test set, the Pearson correlation coefficient score over the Arabic data increases significantly from 0.69425 (2nd worst in Figure 2) to 0.84067 (4th best). 3) The model tends to overestimate the degree of similarity. Among the 46 "serious mistakes", only 15 of them are similar news pairs with predictions indicating dissimilarity. And these wrong predictions are all caused by scraping-related issues (the real content of news is not returned). However, we argue that our entity features may potentially increase the robustness and decrease the level of overestimation. We compare the performance between models with and without entity features. The results show that the differences between true label and prediction decrease for 22 of the 31 dissimilar news pairs.

6 Conclusion

We introduce an entity-enriched Siamese Transformer for SemEval-2022 Task 8: Multilingual News Article Similarity. The error analysis shows that the entity-enrichment leads to statistically significant improvement and make models more robust for computing similarity between news articles in both monolingual and cross-lingual setting. Our entity-enriched LaBSE model achieves a Pearson Correlation of 0.802, ranking 2nd in the competition. The code is available at `https://github.com/iknoorjobs/semeval-code`.

7 Acknowledgement

This work has been partially supported by the WeVerify (EU H2020, grant no. 825297), SoBigData++ projects (EU H2020, grant no. 871042) and CDT in Speech and Language Technologies and their Applications funded by UKRI (grant no. EP/S023062/1).

References

Omar Alonso, Dennis Fetterly, and Mark Manasse. 2013. Duplicate news story detection revisited. In *Asia Information Retrieval Symposium*, pages 203–214. Springer.

Joel Azzopardi and Christopher Staff. 2012. Incremental clustering of news reports. *Algorithms*, 5(3):364–378.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Dhivya Chandrasekaran and Vijay Mago. 2021. Evolution of semantic similarity—a survey. *ACM Computing Surveys (CSUR)*, 54(2):1–37.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

John Gibson, Ben Wellner, and Susan Lubar. 2008. Identification of duplicate news stories in web pages. Technical report, MITRE CORP BEDFORD MA.

Wael H Gomaa, Aly A Fahmy, et al. 2013. A survey of text similarity approaches. *international journal of Computer Applications*, 68(13):13–18.

Yvette Graham and Timothy Baldwin. 2014. Testing for significance of increased correlation with human judgment. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 172–176, Doha, Qatar. Association for Computational Linguistics.

Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, et al. 2017. Claimbuster: The first-ever end-to-end fact-checking system. *Proceedings of the VLDB Endowment*, 10(12):1945–1948.

Ye Jiang, Xingyi Song, Carolina Scarton, Ahmet Aker, and Kalina Bontcheva. 2021. Categorising fine-to-coarse grained misinformation: An empirical study of covid-19 infodemic. *arXiv preprint arXiv:2106.11702*.

Dawid Jurkiewicz, Łukasz Borchmann, Izabela Kosmala, and Filip Graliński. 2020. ApplicaAI at SemEval-2020 task 11: On RoBERTa-CRF, span CLS and whether self-training helps them. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1415–1424, Barcelona (online). International Committee for Computational Linguistics.

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

Soto Montalvo, Raquel Martínez, Arantza Casillas, and Víctor Fresno. 2007. Bilingual news clustering using named entities and fuzzy similarity. In *International Conference on Text, Speech and Dialogue*, pages 107–114. Springer.

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

Iknoor Singh, Kalina Bontcheva, and Carolina Scarton. 2021a. The false covid-19 narratives that keep being debunked: A spatiotemporal analysis. *arXiv preprint arXiv:2107.12303*.

Iknoor Singh, Carolina Scarton, and Kalina Bontcheva. 2021b. Multistage bicross encoder for multilingual access to covid-19 health information. *PloS one*, 16(9):e0256874.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021a. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021b. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Martin Theobald, Jonathan Siddharth, and Andreas Paepcke. 2008. Spotsigs: robust and efficient near duplicate detection in large web collections. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 563–570.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Yun Zhai and Mubarak Shah. 2005. Tracking news stories across different sources. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 2–10.

Xiaojin Zhu and Andrew B Goldberg. 2009. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130.

A Appendix

A.1 Data Distribution

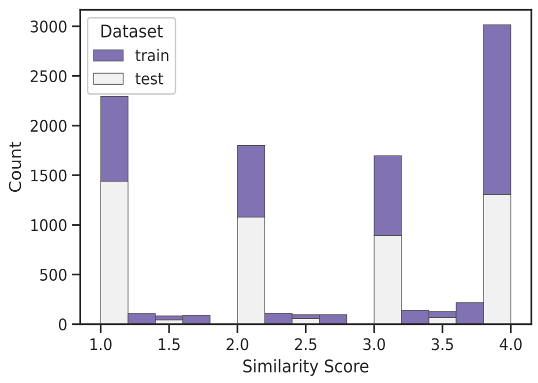


Figure 3: Distribution of training and test data

A.2 Augmented Data Generation

The details of augmented data used in our experiments is as follows,

Augment Data 1. In this, we used the training set provided by organisers where the title of the article is used as a query to all the other articles in the training set that are indexed in Elasticsearch. If title or article are not in English, then the English machine translated version is used to perform the retrieval. Also, we exclude the cases of document pairs that are already present in the training set. We generated 25,125 by using the top 5 retrieved documents for each case.

Augment Data 2. We machine translated the English document pairs with overall similarity score in range of [0.5-1] to other languages. We only choose to translate document pairs that are English to get a good quality machine translated augmented training data. For this we randomly sampled five different language pairs from the list of language pairs that follow the same distribution as that of language pairs found in the test set and machine translated the documents[7] (Fan et al., 2021). We generated 2845 additional document pairs using this augmentation strategy.

[7]Pre-trained model m2m_100_418M is used for our experiments. Ref. `https://github.com/pytorch/fairseq`

Augment Data 3. Here, we utilised "All the news" dataset available on Kaggle[8] which contains 143,000 articles from 15 American publications. Here, title from the SemEval training set is used as query to all the articles in Kaggle new datasets. We generated 34,818 document pairs using the top 5 retrieved documents.

In our experiments, we use augment data 1, 2 and 3 which constitute a total of 62,788 additional document pairs and use them all together along with the training set to train the models.

A.3 Hyperparameters

The Transformer encoder of our Siamese Transformer (shared parameters) model is trained for 4 epochs with a batch size of 8, learning rate of 2e-5 and maximal input sequence length of 512. The output of model along with entity features is passed to the MLP layer which is composed of 32 hidden units trained with learning rate of 1e-3. The output of this MLP layer is used to get the final similarity score for the news articles. Adam (Kingma and Ba, 2015) is used to optimize the model parameters. For all the experiments, the model which works the best on the dev set is submitted and evaluated on the test set provided by the task organisers. All experiments are conducted on a machine with NVIDIA GeForce RTX 3090.

[8]https://www.kaggle.com/snapcrack/all-the-news

SemEval-2022 Task 8: Multi-lingual News Article Similarity

Nikhil Goel
CS Department
Stony Brook University
Long Island, NY
nigoel@cs.stonybrook.edu

Ranjith Reddy
CS Department
Stony Brook University
Long Island, NY
rbommidi@cs.stonybrook.edu

Abstract

This work is about finding the similarity between a pair of news articles. There are seven different objective similarity metrics provided in the dataset for each pair and the news articles are in multiple different languages. On top of the pre-trained embedding model, we calculated cosine similarity for baseline results and feed-forward neural network was then trained on top of it to improve the results. We also built separate pipelines for each similarity metric for feature extraction. We could see significant improvement from baseline results using feature extraction and feed-forward neural network.

1 Introduction

For finding similarity between two documents, the general approach is to get the embeddings for the documents and use some similarity metric like cosine similarity to get similarity measures. Such work has been done in multiple domains like research papers, semantic similarity (Olizarenko and Radchenko, 2021) (Boom et al., 2015) document similarity (Ostendorff et al., 2020) (Rushkin, 2020) and for news articles (Watters and Wang, 2000) (Singh and Singh, 2020) (Blokh and Alexandrov, 2017). There have been other works to find similarity between documents of multiple languages (Potthast et al., 2008) as well. The most significant difference here is that authors have used subjective similarity measures like writing style and authorship for these works.

We do feature extraction corresponding to each similarity metric via a unique feature extraction pipeline. For example, for geolocation, we extract the locations from the text. To make it a domain-specific task, we train a network on top of the generic similarity pre-trained embedding model.

To test our approach, we use a pre-trained model for sentence similarity based on S-BERT (Reimers and Gurevych, 2019a) (Devlin et al., 2019). We calculate cosine similarity (Rahutomo et al., 2012) as our baseline on the embedding of the documents. We train a feed-forward network on top of the pre-trained embedding to learn about our domain-specific task. We also use a large pre-trained model, which can take more words as input and compare the results. Our approaches result in an almost 60% MSE improvement over the baseline model.

Our contributions are mentioned below:

- We implement a strong paraphrase-multilingual-MiniLM baseline and show significant gains in MSE by adding a feed-forward neural network on top to learn domain specific knowledge.
- We do domain-specific feature extraction for each similarity metric to prevent model from learning incorrect information. This feature extraction helps improve the final metrics for the task.
- We do qualitative result analysis and show that the baseline model does not learn relevant information for location/time/entity for the corresponding metrics. We observe that after adding feature extraction, the model learns metric specific information.

2 System Description

2.1 Feature Extraction

We extract the entities from the text using Stanza (Qi et al., 2020). Stanza is a python natural language analysis library that contains tools for parsing sentences, recognize named entities etc. Through our experiments we observe that for similarity measures like geography, entities and time, we just need to extract those relevant named entities from the text. To do this, we build seven stanza pipelines for the seven similarity measures. Feature extraction for each of the metric is described below.

1. **Geography:** We extract the location from the article for each pair. If Stanza cannot extract the location, we use the entire text as the embedding. We linearize the location and feed them to the pre-trained model if there are multiple locations. For example, if there are four locations in the article, the output of the pre-trained model is [4, embedding size]. To further linearize the input, we take the mean of all the rows, so the linearized input dims is [1, embedding size]. We repeat this process for the other news articles and calculate the cosine similarity between the two articles.
2. **Entities:** We extract all the named entities from the article for each pair. We linearize the input in the same way as geography, calculating the cosine similarity. There were no cases in this dataset where stanza was unable to extract entities, but in case there are, they will be handled similarly to location.
3. **Time:** We extracted time and date entities from the articles for each pair. If Stanza cannot extract the time entity, we create an embedding for the entire text instead of the time. The rest of the processing is similar to Geography and entities.
4. For **Narrative**, **Style**, **Tone** and **Overall**, we give the entire text as input to the pre-trained embedding and calculate the similarity based on that. In general, for this slightly subjective measure, the neural network learns the representations on its own.

Using the feature extraction defined above, we build our baseline model. Initially, we were giving entire text as input for location class but were getting similar results (less overfitted but worse on test data). We hypothesized that it might be because the model is not learning about specific location class entities but looking at the overall text. To confirm this hypothesis, we took two news articles that were outputting highly similar scores for location and manually changed the location of one article to some random place. The result was that the model was still outputting the two articles as quite similar, which showed that the model was not learning correctly. We extracted just the entities for an objective metric like location, time, and entities and gave that as input to the model. We are making sure that the model is not learning the wrong information to get correct results.

In the future, we can to try to extract the dependency parse of the sentence and give it as an input to the pre-trained embedding as another step in the feature extraction task. We hypothesize that for the style and tone metric giving the dependency parse of sentence as input to the neural network will improve the results. This is because style and tone of writing is author specific and depends on the writing style of the author which a dependency parse of the sentence can capture, but we have not used that approach for this paper.

2.2 Baseline Model

For the baseline model, we used the en-en articles as our dataset. We then do feature extraction to extract relevant features from the article corresponding to the similarity metric. After that, we got the embeddings for these pairs from the pre-trained model for sentence similarity (miniLM-v6) (Reimers and Gurevych, 2019a) and calculated the cosine similarity to find similarity between the articles and calculated the loss for each similarity metric separately.

Limitation The issue with the approaches done historically is that the similarity metrics were subjective, like comparing two documents to see if the same author has written them or not. For such tasks, semantic and syntactic features of the document play a vital role. For our task, we need to focus more on the article's content. For the baseline model that we create, there are two significant issues. First, we use en-en articles to train the model. To test, we translate other language articles to English using Google translate. This approach is error-prone as it compounds the error. Second, if we directly use the cosine similarity of two embedding from a generic document similarity model, we do not take advantage of the domain-specific task. For example, the document similarity in the case of medical documents will be much different in the case of research documents.

3 Proposed approaches

3.1 Feed-forward network

The first idea is to take advantage of the domain-specific task in this work. To do this, we train a feedforward model on top of the baseline to learn document similarity for the news articles domain. To implement this idea, we concatenated the embedding that we got using the pipeline in Figure 1. This input is fed through a three-layer neural network with a ReLU activation function after each

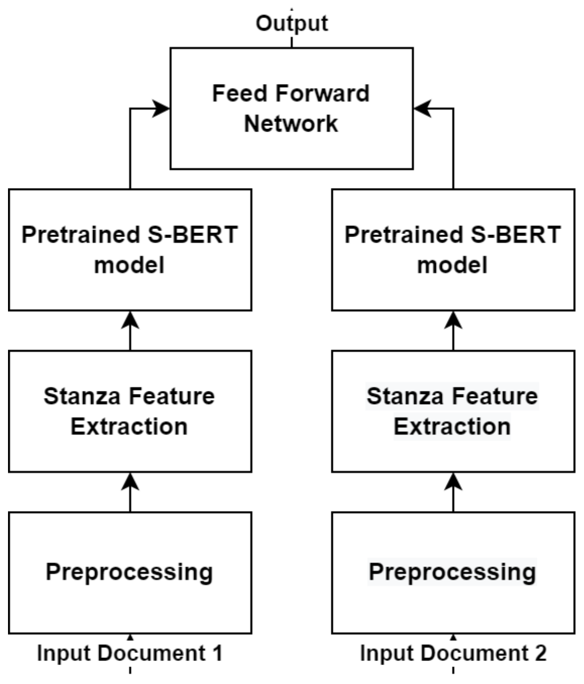


Figure 1: Pipeline for the task. The figure depicts the pipeline for similarity generation containing pre-processing, stanza feature extraction, and pre-trained S-BERT model steps. If we remove the feed-forward and directly calculate the similarity score, that is our baseline model.

layer. The final layer is passed through a softmax layer to get the similarity score between 0 to 1. This model uses MSE loss between the output and predicted value as the loss function, and the optimizer used to train this network is SGD.

3.2 Doc S-BERT

Doc S-BERT is a BERT-base document similarity model that is also called bert-base-nli-mean-tokens. The amount of resources needed to use a base pre-trained model for BERT is much more, but the idea behind using the large doc similarity model is due to a shortcoming in the miniLM model. The miniLM model takes a maximum of 256 words of a document, after which it truncates the rest of the document. The input text length went up to 1000 words, so we believe there is a loss of important information if we use the miniLM model. Both Doc S-BERT and MiniLM models belong to the family of Siamese BERT (Figure 2) (Reimers and Gurevych, 2019b) models.

3.3 Multilingual S-BERT

The idea is to use a multilingual pre-trained model to generate embedding instead of converting other

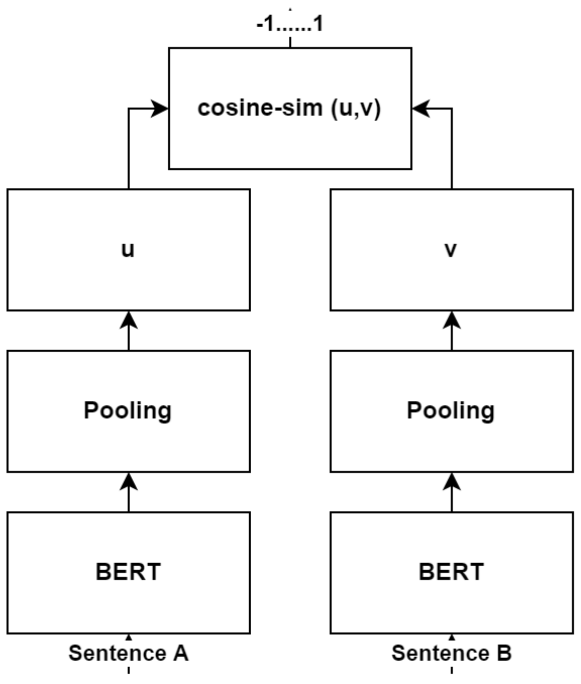


Figure 2: Architecture of S-BERT (Reimers and Gurevych, 2019b)

languages to English to prevent the compounding of errors. We use a multilingual pre-trained similarity embedding generating model. Multilingual miniLM is a single model that is pre-trained for multiple languages. The model is called 'paraphrase-multilingual-MiniLM-L12-v2'. The output embedding dimension from this is similar to our previous model, and the constraints are also similar. We hypothesize that if we can remove the compounding of errors, the multilingual model should give better results than the baseline.

4 Experimental Setup

4.1 Evaluation Measures

The dataset rates similarity metric between 1 to 4, where 1 means most similar, and 4 means most dissimilar. To convert this score to a more usable metric, we subtracted it from the max (4) and divided it by max-min (4-1). The score gets normalized between 0 to 1, with 1 being most similar and 0 being most dissimilar. This score is treated as a regression problem as the scores are rated continuously between 1 to 4 with small buckets.

To evaluate our model, we use MSE loss as the metric. Since we treat this as a regression problem, MSE loss is a good way to judge how the model performance.

$$\text{Mean squared error MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$$

where e_t is the difference between predicted and actual value.

We have also set a tolerance value to get the accuracy. We have tried different values of tolerance to see how our results vary.

If the predicted and actual value difference is less than tolerance, we predict it as a true label; otherwise, we predict it as a false label. We take the mean over the dataset to get the accuracy.

4.2 Preprocessing

We download news articles as HTML files from the URLs given in the dataset. We use beautiful soup to extract the headings and body of the text in the news article and store the metadata in JSON files for each article id. There was some junk data appended at the end of the news article that we manually removed in many articles. We removed the stop words from these articles because our pre-trained model has a word limit in the input document.

4.3 Experimental design

We used the dataset provided by the task organizers [1]. The train dataset has 4964 pairs of news articles with different language articles. En-en:1800, de-de:857, de-en:577, es-es: 570, tr-tr: 465, pl-pl: 349, ar-ar: 274, fr-fr: 72. We split the dataset into train and test sets with the division as 67:33. We did not use a validation set since the dataset was tiny.

We used a variation of S-BERT for all four approaches as the embeddings generation pre-trained model. For baseline and first approach, we use a miniLM model. We use a large miniLM model for the second approach. For the third approach, we used a small multilingual miniLM model. The input truncates after 256 words in the first approach (miniLM FFN) and the third approach(multilingual miniLM). For the second approach (Doc-S-BERT), the input truncates after 256 words.

The output of the embedding size from the small pre-trained model is (384,1). The input to the neural network will be two * embedding size since we concatenate the two article embeddings in our approach, it will be (768,1). We use a three layer feedforward network with layer sizes as (120, 84, 1). After each layer, we feed the input through a ReLU activation function. The final layer is passed through a softmax layer to get the similarity score between 0 to 1. There are seven models for each similarity metric as the input is different for each.

We train the model using SGD with the learning rate at 0.01 and momentum at 0.9. The loss function we chose to train this model is MSE. We trained the model for eight epochs (Early stopping) as the input size was small, and the model started overfitting on the data.

5 Result and analysis

5.1 Baseline

For **location**, the MSE loss that we got is around 0.15 (Table 1). The result is as expected as we hypothesized that just extracting the location entities should give us a good match for similarity. If we allow the threshold to be high (0.5), the accuracy, in this case, is around 0.82. Since its regression, we need to decide the threshold to calculate a metric like an accuracy. With a threshold of 0.33, accuracy was around 0.67. A low threshold like 0.2 gives an accuracy of 0.47.

Similarly, for **time** similarity, we got the MSE as 0.132 (Table 1) and the accuracy with 0.5 tolerance as 0.834. Comparatively, for time, the performance of cosine similarity is good, which is the expected behavior from our initial thoughts since it is a simple task of extracting time from an article and checking its similarity.

For **entities** similarity, we got the MSE as 0.288 (Table 1) and the accuracy with 0.5 tolerance as 0.507. The results are much worse for entities as compared to other results. Our initial thought was that cosine similarity should work with simple tasks like entity similarity. We observed that each article has more than 20 entities at the least, so doing mean of all entities makes it lose all necessary information. We might need to think of a more innovative way of using cosine similarity in this case or use the neural model.

As can be seen from the table, **Overall** class and **Narrative** class show promising results, which is intuitive as the text-similarity of the entire article captures the narrative class and the overall class similarity. As expected, **Style** class and **Tone** class do not show that great accuracy as both style and tone require hidden features like semantic and syn-

[1]URL of the dataset `https://competitions.codalab.org/competitions/33835#learn_the_details-timetable`.

Metric	baseline (cosine-sim) MSE	miniLM (approach1) MSE	doc-SBERT MSE	M-miniLM-cosine-sim MSE	M-miniLM-MSE
Geography	0.15	**0.135**	0.164	0.175	0.134
Time	0.132	**0.074**	0.079	0.171	0.109
Entity	0.288	0.106	**0.098**	0.327	0.133
Narrative	0.129	**0.119**	0.131	0.258	0.127
Style	0.192	**0.075**	0.085	0.136	0.081
Tone	0.187	0.077	0.077	0.131	**0.075**
Overall	0.131	**0.122**	0.132	0.248	0.128

Table 1: Results of different approaches

tactic parse of the sentence, which text embedding similarity cannot capture. These results show the need for neural models to find these hidden features with little help from feature extraction if required.

5.2 miniLM

As we can see clearly from the table (Table 1), using a feed-forward network on top of the pre-trained embedding to make it a domain-specific task decreases the loss and improves the results for almost all similarity metrics. These results confirm our hypothesis for domain-specific knowledge.

Our hypothesis for location class not performing well is that the pre-trained embedding is for sentence similarity, all location entities should be nearby in the embedding space for such models, and overall it will be hard for such models to learn the differences between location classes. If we use a location-specific pre-trained model, we should get good results.

Also, we can see that the improvement is the most for entities, which is intuitive. The baseline model was poorly performing because embedding all entities averaged did not give any helpful information. Training a model on top of it allows the neural model to learn the differences in entities and output much better results. The logic is similar for the time similarity metric as well.

We see significant improvement for style and tone classes as the similarity score loss decreased from 0.192 and 0.187 to 0.075 and 0.077 (Table 1), respectively. As mentioned previously, these require semantic and syntactic information of the text, which a cosine similarity cannot find. The neural model can learn these differences better and show significant improvement.

Narrative class and Overall class gave good results with baseline as they depend on the embedding of the entire text, but the neural model can improve on those results slightly. These results are intuitive as the model does not learn meaningful additional information to improve those metrics.

5.3 Doc S-BERT

Using a large BERT model for sentence similarity did not lead to many significant improvements in test data (Table 1). The shortcoming of the initial model was that it truncates the text after 256 words. The reason for these poor results is that most of the information is present in the headlines and the first paragraph of news articles, and the rest of the article is a detailed explanation of these texts. This logic makes sense as to why the larger model is not showing significant improvement. The objective similarity metric can be calculated using these words, generally less than 256 words, and gain insightful results.

5.4 Multilingual miniLM

For the multilingual model, we first look at the baseline results, which are the cosine similarity between the embedding.

In Table 1, the most significant difference we see is between the narrative class and the overall class. For both these measures, our single language model performed much better than the multilingual model. We hypothesize that because of multiple languages, the embedding of this model learns less about the text itself as much as it learns the semantic and syntactic knowledge of the words.

This hypothesis can be confirmed by looking at the cosine similarity of style class and tone class, which depends more on these hidden features. As we can see, the multilingual model is performing better for these metrics because of the hypothesis stated above. We were unable to test this hypothesis in great detail but looking through online resources; intuitively, it makes sense.

Even the narrative and overall classes, which gave terrible results (Table 1) for cosine similarity, have almost the same accuracy as our English model, which contained only the English data. Using this model for different language types and still getting the same results strengthens our idea of using a multilingual model instead of compounding the errors.

The test loss for geography in multilingual miniLM and miniLM is precisely the same, which shows that the models have been trained similarly for this metric (Table 1). Our hypothesis in miniLM for location should also hold for this model.

The multilingual model is giving poor results for time and entity compared to our previous model (Table 1), which might be because of the feature extraction problem for other languages. We used Stanza to extract the features, but the overall accuracy is much lower for other languages than English. To confirm this hypothesis, we looked at the extracted entities from Other languages, and the number of entities was much lower than they were in English.

5.5 Best model final results

We experiment with multiple approaches using different parameters. Overall, the multilingual model gave us the most consistent results across all languages. We choose this to be our best model. Both models performed well in a few languages but poorly in others. Hence, we chose the most consistent model across all languages as there can be unseen instances of language in the test dataset.

We calculated the Pearson correlation coefficient on the test data to further test this model, which is the official task metric. On the test data [2] that the task organizers have provided, the coefficient was 0.288 for the multilingual model. The eval data has 4902 pairs of news articles and ten languages.

[2]URL of the dataset `https://competitions.codalab.org/competitions/33835#learn_the_details-timetable`.

One of the biggest reasons for the low coefficient was that both the stanza and multilingual model did not support pl and tr languages from the dataset. Further, the multilingual model did not support a few other languages from the eval dataset, which were absent in the training dataset. Due to these two significant shortcomings, our model performed poorly on a few dataset instances while performing well on other languages. In the future, we can try to find other parsers for Named Entity Recognition and different multilingual models which supports all languages.

5.6 Code

The google drive link for the code for this project is given here: link for google drive The link contains all the generated files, readme for instructions, and code.

6 Conclusions

We created an end-to-end pipeline to find similarities between two news articles in this work and used multiple approaches to find the most optimal way to calculate similarity. We were able to deal with the problem of multilingualism best using a multilingual S-BERT model. We were able to identify the shortcomings of the pre-trained model for similarity in the location metric. We also explored using a large model, which did not significantly improve the results. On the eval data, the multilingual model showed decent results even on unseen languages, which shows that we can extend the model to other languages with minor changes.

7 Acknowledgments

We did this work as part of the NLP course's final project, and we would like to thank prof. Niranjan and the TAs for guiding us in this work.

References

Ilya Blokh and Vassil Alexandrov. 2017. News clustering based on similarity analysis. *Procedia Computer Science*, 122:715–719. 5th International Conference on Information Technology and Quantitative Management, ITQM 2017.

Cedric De Boom, Steven Van Canneyt, Steven Bohez, Thomas Demeester, and Bart Dhoedt. 2015. Learning semantic similarity for very short texts. *CoRR*, abs/1512.00765.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Serhii Olizarenko and Viacheslav Radchenko. 2021. Method for determining the semantic similarity of arbitrary length texts using the transformers models.

Malte Ostendorff, Terry Ruas, Till Blume, Bela Gipp, and Georg Rehm. 2020. Aspect-based document similarity for research papers. *arXiv preprint arXiv:2010.06395*.

Martin Potthast, Benno Stein, and Maik Anderka. 2008. A wikipedia-based multilingual retrieval model. In *European conference on information retrieval*, pages 522–530. Springer.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *CoRR*, abs/2003.07082.

Faisal Rahutomo, Teruaki Kitasuka, and Masayoshi Aritsugi. 2012. Semantic cosine similarity. In *The 7th International Student Conference on Advanced Science and Technology ICAST*, volume 4, page 1.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ilia Rushkin. 2020. Document similarity from vector space densities. *CoRR*, abs/2009.00672.

Ritika Singh and Satwinder Singh. 2020. Text similarity measures in news articles by vector space model using nlp. *Journal of The Institution of Engineers (India): Series B*, 102:329–338.

Carolyn Watters and Hong Wang. 2000. Rating news documents for similarity. *Journal of the American Society for Information Science*, 51(9):793–804.

SkoltechNLP at SemEval-2022 Task 8: Multilingual News Article Similarity via Exploration of News Texts to Vector Representations

Mikhail Kuimov **Daryna Dementieva** **Alexander Panchenko**
Skolkovo Institute of Science and Technology
{Mikhail.Kuimov, Daryna.Dementieva, A.Panchenko}@skoltech.ru

Abstract

This paper describes our contribution to SemEval 2022 Task 8 on Multilingual News Article Similarity. The aim was to test completely different approaches and distinguish the best performing. That is why we've considered systems based on Transformer-based encoders, NER-based, and NLI-based methods (and their combination with SVO dependency triplets representation). The results prove that Transformer models produce the best scores. However, there is space for research and approaches that give not yet comparable but more interpretable results.

1 Introduction

The SemEval 2022 Task 8 competition (Chen et al., 2022) aims to develop systems that identify multilingual news articles that provide similar information. This is a document-level similarity task in the applied domain of news articles, rating them pairwise on a 4-point scale from most to least similar. The alikeness of news is measured in such a sense: how similar are them in geography, time, shared entities, and shared narratives. The developed approaches for solving the task can be applied to several different real-world tasks. The first one is the clustering of news.

A lot of news-providing companies want thousands of articles from different publishers on one topic to be combined in a single page showing the news.

Another task is Fake News Detection. This task has become extremely important to solve lately. Different articles on the same news story can be compared to find contradictions in facts and details. This can be an indicator that the story is fake like it is shown in (Dementieva and Panchenko, 2021).

2 Related Work

In (Montalvo et al., 2007), authors extract `PERSON`, `ORGANIZATION` and `LOCATION` named entities (NE) and compose a vector for each of the category with the help of Levenshtein distance function and TF-IDF weighting function, which combines Term Frequency (TF) and Inverse Document Frequency (IDF). These 3 vectors are compared to 3 corresponding vectors of the second news with cosine distance. Obtained scores are combined with the set of IF-THEN rules. In (Rahimi et al., 2019) the approach for cross-lingual transfer is proposed which is evaluated on the Named Entity Recognition task. Dialogue competition (Gusev and Smurov, 2021) on Russian news clustering has produced many promising methods which could be adopted to the multilingual case. Most of them, like (Sergei et al., 2021; Glazkova, 2021), are the variations of fine-tuning the transformer models and making ensembles.

In (Martín et al., 2021), the authors developed the pipeline for checking the news on veracity. They compare embeddings of the news under consideration with ones from the database, using cosine distance. Then they take the most similar news found and apply Natural Language Inference (NLI) model to obtain the probability that two texts contradict each other. This probability is used to decide whether the input news is fake. However, NLI scores can be used to find the similarity between articles.

3 Methodology

To solve the task we have tried several approaches. In the subsection 3.1 we will give an overview on methods exploiting pre-trained transformer models as the foundation. In the next subsection we will explain how the Natural Language Inference (NLI) problem can be reduced to the task of News Article Similarity. Block 3.3 is dedicated to approaches based on the Named Entities extracted from the news texts. In the last section, the approaches which were tested to improve the quality of prediction during the post-evaluation period will be described.

3.1 Transformer-based Pre-trained Encoders

Pre-trained neural masked language models like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have shown superior performance on a wide range of NLP tasks both in monolingual and multilingual settings. During the work on the task of the competition, the approach for fine-tuning Transformers was developed. The following multilingual models were tested: DistilBERT[1], BERT[2] RoBERTa[3], XLM[4]. All these models support all the languages included in the competition dataset. Two different architectures were chosen for fine-tuning the language models. The first one is based on the approach for the BERT Next Sentence Prediction problem described in the original article (Devlin et al., 2019). We will call it **TransformerEncoder-CLS**. The second approach is inspired by the articles (Reimers and Gurevych, 2019; Sergei et al., 2021). It will be labeled as **TransformerEncoder-CosSim** from now on.

3.1.1 TransformerEncoderCLS

The general scheme of the approach is shown in Fig. 1. The Transformer model takes as input 2 tokenized news texts separated by [SEP] token, which is needed for the model to distinguish words from different texts. Also, this sequence of tokens has a special [CLS] token in the beginning. Passing through the layers of the model, each token results in the embedding vector. All the information from the sequence is aggregated in the [CLS] token embedding. That is why we use it as the input to the regression head, which is the combination of fully-connected layer and Sigmoid nonlinearity. The linear layer dimensions are $emb\_len \times 2$, where emb_len is the dimension of the hidden layer. We use the output probability of the first class as the similarity score. Togeth er with mapped to $[0, 1]$ range ground true similarity scores, the predicted scores are passed to the MSE loss function. Transformers weights are not frozen while training and initialized from the aforementioned pre-trained multilingual models. The models were trained on GPU NVIDIA GeForce RTX 3090 for 10 epochs with a learning rate of 10^{-5} and batch size equal to 8.

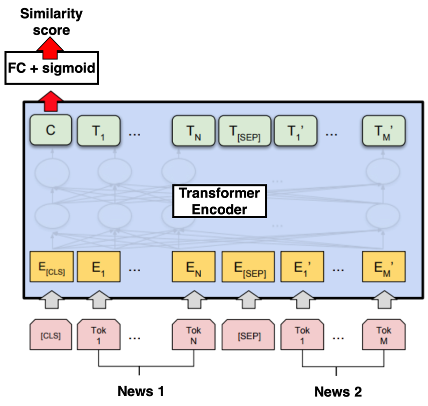


Figure 1: TransformersEncoderCLS architecture, depicted from the original paper (Devlin et al., 2019).

3.1.2 TransformerEncoderCosSim

The general scheme of the approach is shown in Fig. 2. The pre-trained Transformer model takes as input the tokenized news text. Then, Transformer output embeddings are passed through the average pooling followed by a fully-connected layer[5] and L2 normalization layer. This procedure is applied for both compared news. Then, the resulting text embeddings are passed in the cosine distance function which is computed with equation bellow to produce a distance score:

$$cosine\_dist = 1 - |cosine\_sim|.$$

We use absolute value of cosine similarity function because it takes values from -1 to 1. Together with mapped to $[0, 1]$ range ground true scores, the predicted scores are passed to MSE loss function. Transformers weights are not frozen while training and initialized from the aforementioned pre-trained multilingual models. The models were trained on GPU NVIDIA GeForce RTX 3090 for 10 epochs with a learning rate of 10^{-6} and batch size equal to 4.

3.2 Natural Language Inference

The task of estimation similarity between news contents can be reformulated as Natural Language Inference task, which is the main hypothesis tested in

[1] https://huggingface.co/distilbert-base-multilingual-cased

[2] https://huggingface.co/bert-base-multilingual-cased and https://huggingface.co/bert-base-multilingual-uncased

[3] https://huggingface.co/xlm-roberta-base and https://huggingface.co/xlm-roberta-large

[4] https://huggingface.co/xlm-mlm-17-1280

[5] The linear layer dimensions are $emb\_len \times emb\_len$, where emb_len is the dimension of the hidden layer

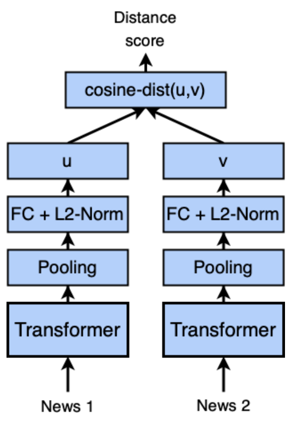
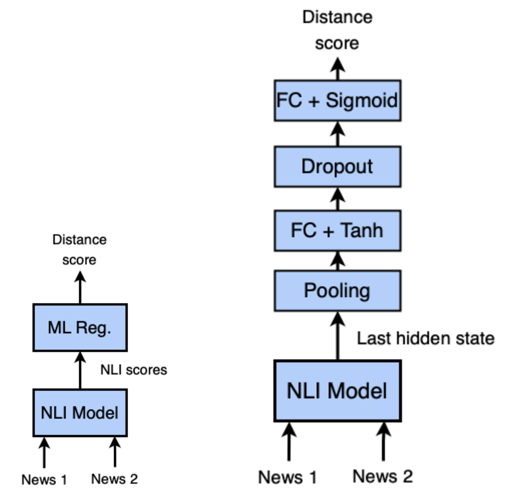


Figure 2: TransformerEncoderCosSim architecture.

this work. *Natural Language Inference* (NLI) is the problem of determining whether a natural language hypothesis h can reasonably be inferred from a natural language premise p (MacCartney and Manning, 2008). The relations between hypothesis and premise can be *entailment*, *contradiction*, and *neutral*. The release of the large NLI dataset (Bowman et al., 2015) and later multilingual XNLI dataset (Conneau et al., 2018) made possible the development of different deep learning systems to solve this task. That is why the pre-trained NLI models like XLM-RoBERTa-large appeared. We use this model pre-trained on multilingual XNLI dataset[6] to obtain NLI scores for pairs "the first news as premise $p \leftrightarrow$ the second one as hypothesis h". The size N of the used content is a hyperparameter of this NLI based approach for the news content similarity computation. NLI model outputs the probabilities of news pair to be classified as entailment, contradiction, or neutral. Hence, it's 3 real numbers from the $[0, 1]$ range. These extracted NLI features are passed as input to the Machine Learning model, which predicts the similarity score for the pair of news under consideration. In our work, we've compared the performance of several regression models: Linear Regression, Support Vector Machine for regression, Decision Trees, Random Forest, Gradient Boosting. The last one gave the best results. The general scheme of the approach is shown in Fig. 3. Also, several improvements to this pipeline were tested:

1. **Both pairs**. Each piece of news is used as a premise and hypothesis. As a result, we get twice more features for training.



(a) Basic architecture. (b) Fine-tuning architecture.

Figure 3: NLI approach

2. **Subject-Verb-Object triplets**. We extract syntactic dependencies from the sentences of a text to make triplets consisting of subjects, verbs, and objects. These triplets are passed to the model. Such an approach shortens the input data, which makes the process of extracting NLI features faster and doesn't have the significant influence in quality of the method. We extracted syntactic dependencies with Spacy library (Honnibal et al., 2020).

3. **Fine-tune**. We fine-tune the NLI model on the data of the competition. The approach is based on the one proposed by (Martín et al., 2021). We add the regression head to the NLI model, which has global average pooling of the last hidden state of the transformer model, linear layer with 768 neurons and $\tanh$ activation, a 10% dropout for training, and a classifier linear layer with sigmoid. The output probability is treated as a similarity score, and MSE loss is used. This regression head is trained, freezing the XLM-RoBERTa-large weights to preserve the previous pre-training. This is optimized using Adam optimizer (Kingma and Ba, 2015) with 10^{-3} learning rate. The general scheme of the approach is shown in Fig. 3.

3.3 Named Entity Recognition

Transformers have great performance but almost no interpretability. In search of interpretability,

[6] https://huggingface.co/joeddav/xlm-roberta-large-xnli

the Named Entity Recognition-based approach has been developed. The general scheme of the approach is shown in Fig. 4. News texts are preprocessed and forwarded to the NER extractor to extract locations (LOC), organizations (ORG), and person entities (PER). For this task we've tested and compared several tools:

1. **Transformer for named entities tagging.** We used BERT[7] pre-trained model from the Hugging Face repository. It is a Named Entity Recognition model for 10 high-resource languages (Arabic, German, English, Spanish, French, Italian, Latvian, Dutch, Portuguese, and Chinese) based on a fine-tuned mBERT base model.

2. **Polyglot for Named Entity Extraction.** The models from this package (Al-Rfou et al., 2015) were trained on datasets extracted automatically from Wikipedia. Polyglot currently supports 40 major languages, including all presented in the dataset of the competition.

3. **Spacy.** Spacy library (Honnibal et al., 2020) provides huge variety of NLP tools, including NER extractor. We used multi-language model,[8] trained on Wikipedia.

In the next step, we vectorize extracted entities with Bag of Words, Tf-Idf, Fasttext (Bojanowski et al., 2017), Bert embeddings[9] for comparison. Then we average all the word vectors. As a result, we obtain 3 vectors (one for each of LOC, PER, ORG entities) for each text. Corresponding vectors for LOC, ORG, PER for two texts are compared with cosine distance to get 3 distance scores for every pair of news under consideration. Then, these scores are passed in the Machine Learning model to get the final distance score. We test several regression models: Linear Regression, Support Vector Machine for regression, Decision Trees, Random Forest, Gradient Boosting.

3.4 Additional study

To improve the quality of the prediction the following two techniques were tested:

1. **Augmentation.** Testing part of the dataset has a lot of language pairs[10] which are not presented in the training part of the dataset. To test the influence of unseen language pairs on the results, we added pairs of news for the missing language pairs. Such augmentation was performed with the help of the Google Translator, which was accessed with the help of Deep Translator python library. The pairs of news were selected randomly from the pairs written in English and then translated to the target languages. Samples were added to the training part of dataset in the same proportion they are presented in the testing part of the dataset. As a result, training dataset was extended to 7505 samples.

2. **Stacking.** Ensembling different models is a common way to improve the scores. To aggregate the dependencies caught by several models, we exploited the technique called stacking. To form the ensemble, we used TransformerEncoderCosSim, TransformerEncoderCLS, fine-tuned NLI model and NER model[11] which has shown the best results in the experiments described bellow. All the models were trained on three quarters of the training dataset. And one quarter of the dataset was used to train the aggregation model. We used Linear Regression model with L_2 regularization as aggregation model.

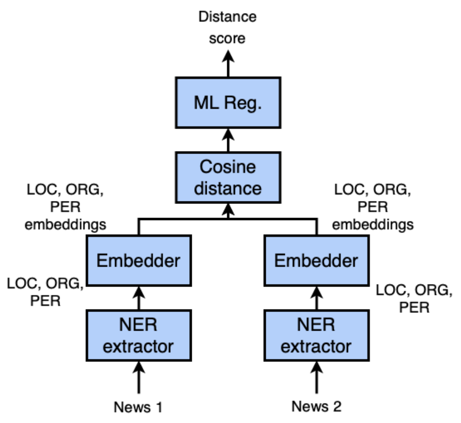


Figure 4: NER approach architecture.

4 Results

Our team has taken the 14th place among 32 participants. The best result for separate model, 0, 734 correlation, was reached by the TransformerEn-

[7] https://huggingface.co/Davlan/bert-base-multilingual-cased-ner-hrl

[8] xx_ent_wiki_sm

[9] bert-base-multilingual-uncased pre-trained model was used

[10] A pair of languages, in which a pair of news is written.

[11] We used the following combination: Huggingface NER tagger, Huggingface embeddings, Gradient Boosting ML model.

coderCosSim model. Final results for all separate methods, as well as the best competition score, are provided in Table 1. Also, we provide the results for ensembles of models in the Table 3. The application of ensembling and augmentation techniques improved the best result to $0,763$ correlation. In addition to the test set, which was provided by organisers during the evaluation period, the performance of the developed systems was evaluated on the validation set. Validation set was randomly sampled from the training data[12] in case of TransformerEncoder methods, including fine-tuned NLI model. For other methods the results on validation are the results obtained with 5-fold cross-validation.

	Validation	Evaluation
TransformerEncoderCLS	**0.813**	0.706
TransformerEncoderCosSim	0.793	**0.734**
NLI	0.478	0.477
NLI fine-tuned	0.670	0.632
NER	0.496	0.395
NLI + NER	0.615	0.546
Best SemEval result	—	0.818

Table 1: Overall results. Pearson correlation.

Transformer models. As it has already been said TransformerEncoderCosSim model has shown the best result. It was the one with XLM[13] pre-trained model. The worst score was given by the DistilBert model. We provide the comparison of different encoders from Transformers for 2 proposed models in the Table 6 in the appendix. As for the TransformerEncoderCLS model, its performance has dropped by 12% on the evaluation part of the dataset in comparison to validation part. And it's become worse than the TransformerEncoderCosSim model, although it showed better results on the cross-validation.[14] In general, the transformer-based models have a lower correlation on the evaluation data. You can see a similar behavior for the NLI fine-tuning approach.

NLI. The comparison of the results for NLI-based models is provided in Table 2. The best score for the NLI approach was given by the Gradient Boosting model. (We provide the comparison of results for different Machine Learning models for «NLI pairs - titles» in the Table 7 in appendix.) The fine-tuning approach has given the best correlation here. Also, there is a tendency for smaller input text to have better scores. The highest correlation was achieved when only titles were given as input. The reason for that could be that the NLI model was trained on the XNLI dataset, composed of short phrases. That is why it was decided to try to shorten the news with the extraction of SVO triplets from them. The extracted triplets were joined to form a text which was forwarded to the input of the NLI model. As you can see from Table 2 the quality of both methods (with fine-tuning and without) has dropped significantly. Hence, the conclusion is that despite SVO triplets give a good summary of the given text, they are not applicable, at least without any complex processing, for the task of comparing the news. Also, it could mean that the source of similarity of articles is not contained in Subjects, Verbs, and Objects. Last, it is worth mentioning that the resulting summary for big texts still has quite a large size in comparison to titles.
The idea to extract NLI scores from both pairs, as it was described in devoted subsection, gave an improvement. Also, it can be noticed that the NLI approach without fine-tuning is quite robust to adding new languages. The score for "NLI pairs - titles" has only a slight decrease on the evaluation dataset. Although the correlation for single NLI features is low, it becomes significantly better in combination with features with the NER-based method. This approach is described in more details in the devoted paragraph bellow.

	Validation	Evaluation
NLI tiltes	0.453	0.438
NLI pairs - titles	0.478	0.477
NLI pairs - titles + text	0.354	0.310
NLI pairs - SVO	0.154	0.107
NLI fine-tuned - titles	**0.670**	**0.632**
NLI fine-tuned - titles + text	0.627	0.589
NLI fine-tuned - SVO	0.495	0.422

Table 2: Comparison of NLI approaches. Pearson correlation.

NER. One can find the comprehensive comparison of different NER taggers, various vectorizing techniques and different Machine Learning models for prediction of distance score in the Table 5 in the appendix. You can see that the best correlation was shown by combination: Huggingface NER tagger, Huggingface embeddings, Gradient

[12] The size of validation set was 0.25 from the size of the training dataset.
[13] https://huggingface.co/xlm-mlm-17-1280
[14] Model which has shown the best result: https://huggingface.co/xlm-roberta-large

Boosting ML model. In general, Gradient Boosting has shown superior scores for all combinations of NER taggers and vectorizers. Also, Huggingface embeddings in combination with this model have shown the highest results for all vectorizing methods listed in the Methodology section. However, in comparison to NLI and Transformers approaches, the results for NER models are significantly lower. Looking at the outputs of the model (You can find the examples in Table 4 in appendix), the following behaviors can be noticed. In our method in cases when no named entities were found for the `PER`, `ORG` or `LOC` classes, the distance score was set to 0.5, because it is not clear whether the absence of named entities is an indicator of similarity or not. These 0.5 scores confuse the model, increasing its generalization error. The second problem is that when there is no overlap of named entities in one of the classes, it could lead to two bad outcomes. When the other two distance scores correctly reflect the ground true similarity, like in the second example in Table 4, the one with no overlap could be large, which spoils the overall prediction. The second behavior happens when the extracted entities have no straight overlap but happen to be similar in vector space. For example, two different news about the close locations. In this case, the model can output a small distance, which is not correct. Also, the errors of the NER tagger makes the model performance worse. As a result, the model tends to predict values from the middle of the $[1, 4]$ range, avoiding its edges. In addition, the problems described make the results even worse on unseen evaluation data. We provide the comparison of the best results for different NER extractors for validation and evaluation in the Table 8.

NER + NLI. As you can conclude from Table 1, NER features, having poor single performance, add significant improvement in correlation being combined with NLI features. To obtain this result we have taken the features used in best-scored NLI and NER models. For classification Gradient Boosting ML model was used as it had given the highest results for both approaches.

Additional study. The application of augmentation to the training part of the dataset improved the result of the best performing model from 0.734 to 0.746, which is a slight improvement. It can be concluded that the performance of this model is not highly effected by unseen language pairs. The increase in score may be caused just by the increase of the number of training samples.

	Correlation
TrEncCLS, TrEncCosSim	0.752
TrEncCLS, TrEncCosSim, NLI	**0.763**
TrEncCLS, TrEncCosSim, NLI, NER	**0.763**

Table 3: Comparison of the results for different ensembles on the evaluation dataset. Pearson correlation. The names of TransformerEncoders models were shortened.

The results for stacking of the models can be found in the Table 3. In this experiment stacking technique was combined with augmentation, which showed a slight improvement in score. You can see that the addition of the predictions obtained with the NER model gives no increase in score. Overall, the augmentation together with stacking gave the 4% improvement to the result of TransformerEncoderCosSim model.

5 Conclusion

We have tested several approaches, including two systems based on Transformer-based encoders, two NLI approaches (with fine-tuning and without), NER-based pipeline and the ensemble of these models. The best result was achieved by the ensemble of TransformerEncoderCosSim, TransformerEncoderCLS and fine-tuned NLI models. To improve the scores the following things can be done. For models based on Transformer-based encoders, sentence Transformers can be tested. To improve the NER-based method additional features can be added (addition of NLI features improved the correlation), and also we can apply the binary mask for the feature matrix not to take into account 0.5 values while calculating the loss during the training process can be applied.

Source code of our solutions is available online[15]. Also, the hyper-parameters of the models can be found there.

Acknowledgements

This work was supported by MTS-Skoltech laboratory on AI.

References

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. 2015. POLYGLOT-NER: massive

[15] `https://github.com/skoltech-nlp/multilingual_news_similarity`

multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, pages 586–594. SIAM.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–146.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2475–2485. Association for Computational Linguistics.

Daryna Dementieva and Alexander Panchenko. 2021. Cross-lingual evidence improves monolingual fake news detection. In *Proceedings of the ACL-IJCNLP 2021 Student Research Workshop, ACL 2021, Online, JUli 5-10, 2021*, pages 310–320. Association for Computational Linguistics.

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

A. Glazkova. 2021. Towards news aggregation in russian: a bert-based approach to news article similarity detection. In *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue 2021". Moscow, Russia (Online)*.

Ilya Gusev and Ivan Smurov. 2021. Russian news clustering and headline selection shared task. *CoRR*, abs/2105.00981.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python.

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

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

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK*, pages 521–528.

Alejandro Martín, Javier Huertas-Tato, Álvaro Huertas-García, Guillermo Villar-Rodríguez, and David Camacho. 2021. Facter-check: Semi-automated fact-checking through semantic similarity and natural language inference. *CoRR*, abs/2110.14532.

Soto Montalvo, Raquel Martínez-Unanue, Arantza Casillas, and Víctor Fresno. 2007. Bilingual news clustering using named entities and fuzzy similarity. In *Text, Speech and Dialogue, 10th International Conference, TSD 2007, Pilsen, Czech Republic, September 3-7, 2007, Proceedings*, volume 4629 of *Lecture Notes in Computer Science*, pages 107–114. Springer.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 151–164. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Khaustov Sergei, Kabaev Andrey, Gorlova Nadezda, and Kalmykov Andrey. 2021. Bert for russian news clustering. In *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue 2021". Moscow, Russia (Online)*.

Appendix

pair_id	NER 1	NER 2	dist. LOC	dist. PER	dist. ORG	Predict.	Ground true
1484012638 1483801741	**LOC**: Baku, Azerbaijan, Shamakhi, Ismayilli, Aghsu **PER**: Ilham Aliyev **ORG**: _	**LOC**: Azerbaijan, Baku **PER**: Ilham Aliyev **ORG**: _	0.148	0.000	0.500	2.959	2.500
1483806302 1483770632	**LOC**: Atlanta, GA, Washington, D. C., Capitol Hill, BarackO, America, Georgia, New Jersey **PER**: John Lewis, Lewis, RepJohnLewis, Barack Obama, God, Stacey Abrams, Cory Booker, Jim Crow, Mark Hamill **ORG**: Ku Klux Klan	**LOC**: America, Georgia, Mississippi Delta, Edmund Pettus Bridge **PER**: John Lewis, Peniel Joseph, Jim Crow, Barbara Jordan, Peniel Joseph, Lewis, Crow, Donald Trump, **ORG**: Center for the Study of Race and Democracy, LBJ School of Public Affairs, CNN, University of Texas at Austin	0.078	0.071	0.971	2.492	1.000
1546012672 1488866568	**LOC**: Dresden, Chemnitz **PER**: _ **ORG**: Staatsanwaltschaft	**LOC**: Dresden **PER**: Carolyn, Carolyn Anne Cavender **ORG**: Jackson Madison, General Hospital	0.471	0.500	0.998	3.360	4.000

Table 4: Example of performance of the best NER model. (Huggingface NER extractor, Huggingface vectorizer, Gradient Boosting model).

In this section, we provide some further comments on Table 4. In the Table example output of the best-performing NER approach can be found. You can notice several patterns, which could be the reason for the low quality of prediction of NER approaches. First of all, the errors of the NER tagger makes the model performance worse. You can see several wrong detections. For example, «BarackO» definitely should be tagged as person, not location, in the second example.

Also, in our method, in cases when no named entities were found for the `PER`, `ORG` or `LOC` classes, the distance score was set to 0.5, because it is not clear whether the absence of named entities is an indicator of similarity or not. These 0.5 scores confuse the model, increasing its generalization error. The second problem is that when there is no overlap of named entities in one of the classes, it could lead to two bad outcomes. When the other two distance scores correctly reflect the ground true similarity, like in the second example in Table 4, the one with no overlap could be large, which spoils the overall prediction.

The second behavior happens when the extracted entities have no straight overlap but happen to be similar in vector space. For example, two different news about the close locations. In this case, the model can output a small distance, which is not correct. As a result, the model tends to predict values from the middle of the $[1, 4]$ range, avoiding its edges. However, there are examples which show good performance. In the third example, there is an overlap in one word for `LOC`, which gives the distance from the middle of the range. There is no overlap in organizations. As a result, we get a score quite similar to ground true, taking into account the peculiarities discussed above.

Tagger	Vectorizer	Linear Regression	SVR	Decision Tree	Random Forest	Gradient Boosting
Huggingface	BOW	0.202	0.200	0.154	0.244	0.246
	Tf-Idf	0.195	0.191	0.135	0.229	0.239
	Fasttext	0.194	0.194	0.157	0.320	0.326
	Huggingface	0.250	0.250	0.200	0.385	**0.395**
Polyglot	BOW	0.228	0.227	0.146	0.240	0.244
	Tf-Idf	0.220	0.218	0.143	0.227	0.226
	Fasttext	0.206	0.205	0.151	0.309	0.310
	Huggingface	0.211	0.211	0.180	0.334	**0.342**
Spacy	BOW	0.227	0.227	0.147	0.230	0.235
	Tf-Idf	0.223	0.223	0.154	0.224	0.231
	Fasttext	0.184	0.183	0.146	0.254	0.259
	Huggingface	0.219	0.220	0.152	0.278	**0.279**

Table 5: Comparison of different NER taggers, vectorizers and ML models for evaluation dataset. Pearson correlation.

	Transformer-EncoderCLS	Transformer-EncoderCosSim
distilbert	0.591	0.679
bert-base-cased	0.644	0.704
bert-base-uncased	0.678	0.714
xlm-roberta-base	0.656	0.643
xlm-roberta-large	**0.706**	0.718
xlm-mlm-17-1280	0.650	**0.734**

Table 6: Comparison of performance of different pre-trained encoders from Transformers on evaluation dataset. Pearson correlation.

	Validation	Evaluation
LinearRegression	0.290	0.364
SVR	0.288	0.356
DecisionTreeRegressor	0.228	0.273
RandomForestRegressor	0.477	0.469
GradientBoostingRegressor	**0.478**	**0.477**

Table 7: Comparison of performance of different pre-trained encoders from Transformers for «NLI pairs - titles» approach. Pearson correlation.

	Validation	Evaluation
Polyglot	0.461	0.342
Spacy	0.426	0.279
Huggingface	**0.496**	**0.395**

Table 8: Comparison of the results on cross-validation and evaluation dataset. Pearson correlation.

IIIT-MLNS at SemEval-2022 Task 8: Siamese Architecture for Modeling Multilingual News Similarity

Sagar Joshi[*], **Dhaval Taunk**[*], **Vasudeva Varma**
IIIT Hyderabad, India
{sagar.joshi, dhaval.taunk}@research.iiit.ac.in
vv@iiit.ac.in

Abstract

The task of multilingual news article similarity entails determining the degree of similarity of a given pair of news articles in a language-agnostic setting. This task aims to determine the extent to which the articles deal with the entities and events in question without much consideration of the subjective aspects of the discourse. Considering the superior representations being given by these models as validated on other tasks in NLP across an array of high and low-resource languages and this task not having any restricted set of languages to focus on, we adopted using the encoder representations from these models as our choice throughout our experiments. For modeling the similarity task by using the representations given by these models, a Siamese architecture was used as the underlying architecture. In experimentation, we investigated on several fronts including features passed to the encoder model, data augmentation and ensembling among our major experiments. We found data augmentation to be the most effective working strategy among our experiments.

1 Introduction

News articles from the web covering the same event tend to differ on regional and political biases, style of writing, conciseness and preciseness of coverage and the intended audience of the news outlet. Misleading and confusing articles might lead to unnecessary confusion, chaos and tensions potentially exacerbating or even creating non-existent issues. Often, news articles covering the same event in different languages from varied regional sources are readily available. This can apply for happenings of local, regional as well as international significance. In such a scenario, it is pertinent to able to identify or cluster together news articles covering the same story in different languages and different view points while also being able to distinguish between articles similar in style but covering different happenings.

The significance of the problem of multilingual news article similarity (Chen et al., 2022) lies towards the applicability in this area by scoring a pair of news articles based on their similarity in coverage of the event without focusing on the subjectivity of the content.

In our approach, we model the task as a regression problem by making use of a Siamese neural network (Bromley et al., 1993) as the base architecture. We perform experiments in feature engineering by making use of metadata such as title, description, keywords and tags in addition to the news text. We also experimented with artificially augmenting the data by document permutation. Also, we tried with ensembling of two models - one trained using only textual features, the other using metadata information.

While data augmentation strategy did give a decent improvement in performance, considering the length of the news articles and the coverage of a variety of entities or events throughout the article, globally modeling the text representation might not be the best way to model for this task. In the subsequent sections, we elaborate on our experiments performed and the results obtained and analyze the same. We have open-sourced our code[1] for ease of replicability and improvement over our techniques.

2 Task Definition

The task of multilingual news similarity consists of predicting the similarity score between two news articles on a scale of 1 to 4, with a higher score indicating a higher degree of similarity. The news articles may belong to either the same or different

[*]These authors contributed equally to this work.

[1]https://github.com/sagarsj42/multilingual-news-article-similarity

Lang	#Train
ar_ar	274
de_de	857
de_en	577
en_en	1800
es_es	570
fr_fr	72
pl_pl	349
tr_tr	465

Table 1: Stats of different languages available in train set

languages, with there being no specified set of languages in which the articles might be written in. Most of the pair of news articles were annotated by 1-3 annotators, with the maximum no. of annotations per data sample being 8. In case of multiple annotations per sample, the scores were averaged.

Statistics of the data created for the task are shown in table 1 and 2. The training and evaluation (test) splits were prepared such that the languages appearing in either of the sets might not occur in the other, which shows the necessity of a good multilingual representation for modeling a solution. The news articles were downloaded from the provided URLs following which the dataset was prepared for the task. There were 18 and 22 articles not accessible by the provided URLs in the train and test sets respectively, which were replaced by a placeholder dummy text. 10 % of the training data was used as the validation set. The model with the best performance on the validation split was evaluated on the test data.

3 System Description

Figure 1 shows the architecture we adopted for modeling the task. In the Siamese architecture (Bromley et al., 1993), an encoder representation is taken for each of the news articles following which a linear layer is used for reducing the dimensionality of the representation. An aggregation of these representations is then performed for having a unified representation of the two articles before passing them through fully connected layers that finally output the similarity score from 1 to 4. The entire architecture was trained end-to-end by minimizing the mean squared error (MSE) loss between the actual scores and model predictions. We experimented with various strategies in this generalized architecture which we elaborate in the remainder of this section.

Lang	#Test	Lang	#Test
ar_ar	78	es_it	247
ar_en	11	fr_fr	98
de_de	494	fr_pl	10
de_en	152	hu_hu	1
de_fr	85	it_en	1
de_pl	27	it_it	371
de_ru	1	ja_en	2
el_el	1	ja_ja	5
en_ar	11	ja_zh	2
en_de	9	nl_fr	2
en_en	268	pl_en	58
en_es	6	pl_pl	179
en_fr	3	ru_de	2
en_it	17	ru_en	3
en_pl	5	ru_ru	196
en_ru	1	tr_tr	240
en_zh	10	zh_en	63
es_da	1	zh_nb	1
es_en	380	zh_zh	164
es_es	194		

Table 2: Stats of different languages available in test set

1. **Features passed to the encoder.**
   (a) **News text.** The plain text content of the news article is used as the input. The distribution of the news content lengths is shown in Figure 2. The articles have a mean length of 589 tokens and a standard deviation of 954, with about 60.5% of the documents having length greater than 512 tokens, which is the maximum tokenization output size for the encoders used. Considering, however, that most of the salient information of a news article is contained in its initial section, the token lengths were capped to 512 for all news texts before feeding to the encoder.
   (b) **Metadata.** Information such as title, description, keywords and tags was present for the many of the news articles, the statistics of which are shown in table 3. We ran experiments using only these features as well as concatenating them before the news text, capping the overall length to 512 tokens.
   (c) **NER-extracted features.** Since the task entails determining the similarity based on objective features such as entities, date/time

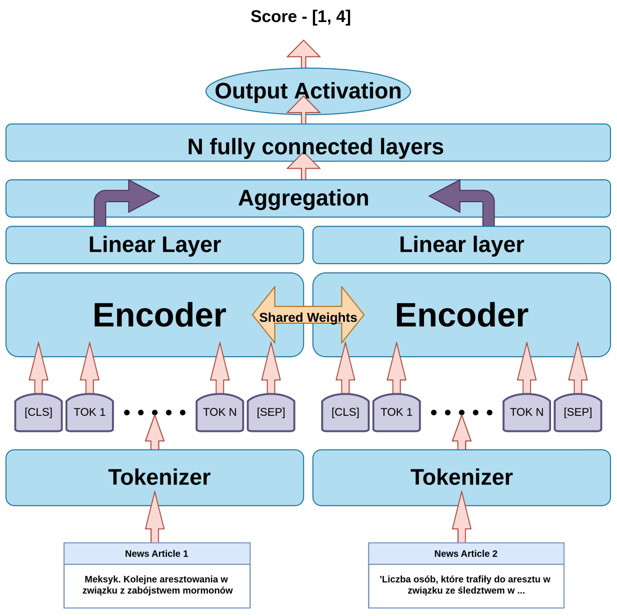


Figure 1: Underlying architecture used across the experiments performed

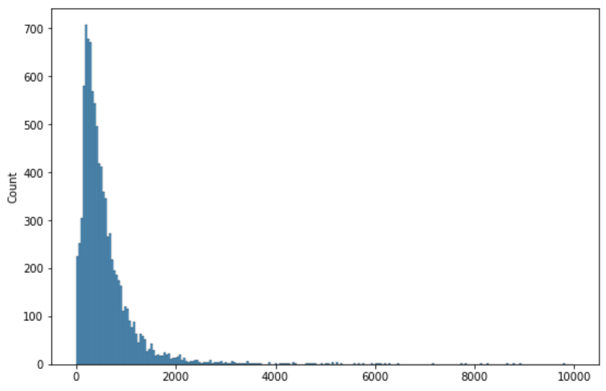


Figure 2: Distribution of number of tokens per news article

values, places, NER features for all the available tags were extracted from text using the Multilingual BERT model from DeepPavlov (Burtsev et al., 2018).

2. **Base encoder model.**
   (a) **XLM-RoBERTa.** (Conneau et al., 2020) This transformer-encoder (Vaswani et al., 2017) based model used was pretrained on 100 languages using masked language modeling objective, achieving remarkable improvements on various cross-lingual understanding tasks.
   (b) **Multilingual DistilBERT.** (Sanh et al., 2019) A distilled version of the multilingual cased BERT-base (Devlin et al., 2018) model trained on Wikipedia data from 104 languages was also used as the base encoder. While the performance of the distilled version was slightly lesser as compared to the original model, this version was chosen on account of it having relatively less no. of paramaters which usually suits well for low-data settings.

Attribute	Count
meta_keywords	4430
tags	4430
title	4421
meta_description	4121

Table 3: Count of the non-null metadata attributes present in the data for the ones used as additional features.

3. **Concatenating encoder representations.** The following concatenation strategies were tried out for building an aggregated representation of the two news articles:
   (a) $[\,|\,x_1 - x_2\,|\,;\,(x_1 + x_2)/2\,]$
   (b) $[\,x_1\,;\,x_2\,;\,|\,x_1 - x_2\,|\,]$

   Here, $x1$ and $x2$ are the encoder representations of the two news articles after passing through the linear layer for reduced dimensionality.

4. **Data augmentation.** Synthetic data samples were created by randomly permuting the sentence order in the news articles to create an augmented data of size 3, 4 and 5 times the original size.

5. **Output activation.** We tried using Sigmoid and ReLU activation functions at the output along with also trying out simple linear output without any activation to determine the best one.

6. **Ensembling.** An ensemble of two models was created by combining the prediction scores of the models by simple average as well as weighted average, in the latter case of which the scores were determined by first training a linear regression model based on the prediction scores on the train data.

4 Experimentation & Results

4.1 Experiments

In this section, we describe the set of experiments performed based on the strategies described in section 3. Unless otherwise specified, sigmoid activation was used at the output and the final score was scaled in the range of 1 to 4 as $3*sigmoid(.) + 1$.

1. **XLM-TXT**: News text feature passed as input to XLM-RoBERTa (XLMR).
2. **DB-TXT**: News text feature passed as input to the DistilBERT model.
3. **XLM-MTD**: Concatenated metadata features passed as input to XLMR.
4. **XLM-MTD-TXT**: Concatenation of metadata features with text passed to XLMR.
5. **XLM-NER-MTD**: Concatenation of extracted NER features and metadata passed as input to XLMR.
6. **DB-NER-MTD**: Concatenation of extracted NER features and metadata passed as input to DistilBERT.
7. **DB-CAT3**: The second concatenation strategy for feature aggregation as described in section 3 used in the same setting as DB-TXT.
8. **XLM-REL**: ReLU activation used at the output in the same setting as XLM-TXT.
9. **DB-CAT3-LIN**: A simple linear output without any activation kept in the same setting as DB-CAT3.
10. **DB-DA3**: Data augmented to thrice the original size, fed to setting same as DB-TXT.
11. **DB-DA4**: Data augmented to four times the original size, fed to setting same as DB-TXT.
12. **XLM-DA5**: Data augmented to five times the original size, fed to setting same as XLM-TXT.
13. **SA**: Simple averaging of the predictions of XLM-MTD-TXT and XLM-DA5.
14. **WA**: Weighted average of the predictions of XLM-MTD-TXT and XLM-DA5.

4.2 Training Setup

The training was done with the number of epochs ranging from 5 to 10. The batch size for train set was kept to be 4 and gradients were accumulated over 8 steps giving an effective batch size of 32. Adam (Kingma and Ba, 2014) optimizer with a weight decay (Loshchilov and Hutter, 2019) of 0.01 was used and the learning rate was kept to a constant value of 5e-6. Validation was performed four times per epoch and the model performance was evaluated using Pearson's Correlation Coefficient (PCC) and Mean Absolute Percentage Error (MAPE) along with the MSE loss value. The best checkpoint was saved based on the PCC score on validation set, following which the predictions on test set were sent for evaluation.

4.3 Results

Results for all the experiments performed on the validation set are shown in table 4 and the PCC reported on some of the experiments on test set are in table 5.

Experiment	Validation set		
	PCC	MAPE	MSE
XLM-TXT	0.53	0.39	0.98
DB-TXT	0.55	0.41	0.93
XLM-MTD	0.46	0.47	1.03
XLM-MTD-TXT	0.52	0.41	0.94
XLM-NER-MTD	0.45	0.43	1.05
DB-NER-MTD	0.47	0.43	1.04
DB-CAT3	0.42	0.47	1.06
XLM-REL	0.45	0.43	1.05
DB-CAT3-LIN	0.42	0.46	1.10
DB-DA3	0.52	0.38	0.99
DB-DA4	**0.58**	**0.41**	**0.94**
XLM-DA5	0.54	0.37	0.99
SA	0.49	0.42	1.02
WA	0.51	0.53	0.95

Table 4: Results of all the experiments performed on validation set.

Experiment	Test PCC
DB-DA3	0.436
DB-DA4	**0.441**
SA	0.43

Table 5: PCC for three of the experiments performed on test set.

4.4 Analysis

The major insights that can be derived out of the results on the validation set are:

- **Metadata as features.** Using plain text (XLM-TXT, DB-TXT) turned out to be the best way to capture the representation among the experiments we tried. The metadata information in itself (XLM-MTD) was insufficient to provide the representation. Even concatenating the metadata information with text (XLM-MTD-TXT) resulted in a suboptimal solution.

- **NER output as features.** The NER output concatenated with metadata features (XLM-NER-MTD, DB-NER-MTD) did not result in a great feature modeling, with the metadata in combination with text and only text input performing better.

- **Encoder model.** DistilBERT-based multilingual model performed consistently better than its XLMR counterpart across similar experiments (DB-TXT v/s XLM-TXT, DB-NER-MTD v/s XLM-NER-MTD). This might be due to DistilBERT having lesser no. of parameters, thus suiting better against overfitting.

- **Concatenation strategy.** The aggregation strategy of concatenating the absolute difference and average of the two news article representations (DB-TXT) worked better than the other strategy (DB-CAT3) tried out.

- **Output activation.** Sigmoid activation[2] achieved the best training trajectory and results as compared to ReLU (XLM-REL) and linear (DB-CAT3-LIN). During training, the loss from ReLU and linear activations started off with very high values before achieving convergence at values subpotimal as compared to that on sigmoid output.

- **Ensembling.** Simple (SA) and weighted averaging (WA) for ensembling performed competitively, with the validation PCC on simple averaging having an edge over the weighted averaging one.

- **Effectiveness of data augmentation.** Data augmentation turned out to be the best strategy so far, as obvious from the results on validation and test sets. Augmenting the data to 4 times the original size turned out to the best among the tried values $\epsilon \{3, 4, 5\}$ (DB-DA3, DB-DA4, XLM-DA5).

As an overall insight, it seems the method of globally representing the news article by a single representation after capping the length to a smaller, fixed size is not the best way in modeling a solution for this problem. A solution exploiting the information present in the articles at a more granular level - either by explicit feature extraction or implicit detection of these features through techniques suitable across multiple languages can be experimented with in pursuit of better results.

Another possible conjecture we speculate based on the poorer performance on the test dataset is that finetuning the underlying multilingual transformer encoder models might have hampered the effectiveness of the multilingual representation for languages that were not present in the training or validation sets, but were present at the time of evaluation. It is to be noted that since the validation dataset was a split taken from the original training data itself, there was not much of a difference between these two distributions. Hence, if this conjecture holds, a solution that trains the model parameters for the task without deranging their multilinguality aspect should provide a scalable solution across multiple languages.

5 Conclusion

We presented our base architecture adopted for the task of multilingual news article similarity and explained the various experiments performed on the same. An analysis of the results gave insights on what strategies worked best on our underlying Siamese architecture, of which we determined data augmentation to be the most effective one. Finally, we provided some insights on the modeling efficiency of the architecture adopted along with directions for possible improvement. This problem being very relevant in the face of diverse, multilingual news content being continually generated from diverse sources, improvements achieved in this task would be of value in industry and social benefit.

References

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, page 737–744, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras

[2]Apart from XLM-REL and DB-CAT3-LIN, all the experiments used sigmoid activation at the output.

Khakhulin, Yuri Kuratov, Denis Kuznetsov, Alexey Litinsky, Varvara Logacheva, Alexey Lymar, Valentin Malykh, Maxim Petrov, Vadim Polulyakh, Leonid Pugachev, Alexey Sorokin, Maria Vikhreva, and Marat Zaynutdinov. 2018. DeepPavlov: Open-source library for dialogue systems. In *Proceedings of ACL 2018, System Demonstrations*, pages 122–127, Melbourne, Australia. Association for Computational Linguistics.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

HuaAMS at SemEval-2022 Task 8: Combining Translation and Domain Pre-training for Cross-lingual News Article Similarity

Sai Sandeep Sharma Chittilla and **Talaat Khalil**
Huawei Technologies R&D
Amsterdam, Netherlands
{sandeep.chittilla, talaat.khalil}@huawei.com

Abstract

This paper describes our submission to SemEval-2022 Multilingual News Article Similarity task. We experiment with different approaches that utilize a pre-trained language model fitted with a regression head to predict similarity scores for a given pair of news articles. Our best performing systems include 2 key steps: 1) pre-training with in-domain data 2) training data enrichment through machine translation. Our final submission is an ensemble of predictions from our top systems. While we show the significance of pre-training and augmentation, we believe the issue of language coverage calls for more attention.

1 Introduction

In the recent few years, there has been a growing interest towards automating news understanding tasks thanks to the continuous demand by downstream applications. One of the most important aspects of news understanding is identifying news articles that cover the same stories. Grouping such similar articles can be useful in multiple application scenarios including news recommendation, news stories analysis, news retrieval and ranking amongst others.

Task 8 in SemEval 2022 (Chen et al., 2022) provides an experimental setup to address and identify the challenges of assessing the similarity between two news articles. Unlike standard document similarity tasks (Agirre et al., 2015), this task is focused on a more challenging multilingual setting where the systems are not only expected to evaluate pairs of long text articles in the same language but in different languages as well. Moreover, the task assumes that accurately identifying articles that share the same story cannot be solely captured by textual similarity since the underlying similarity function is hypothetically a combination of a set of other features like time, geo-location, mentions of named entities and narratives.

Participating teams are required to provide a similarity score for each pair of news articles. The scores should range between 1 and 4 where 1 indicates that two stories are almost identical and 4 indicates no similarity at all. The systems are evaluated based on the Pearson's correlation with ground truth scores which are provided in a test set annotated by human evaluators.

In our submission, we hypothesize that the sub-dimensions (that are assumed to be story similarity predictors by the task organizers) do not have to be independently modeled and present an approach that assumes that such sub-dimensions can be represented in the model's latent space while directly optimizing to learn article similarity scores. Formally speaking, we model the similarity task as a supervised regression problem to optimize the similarity score between two news articles. To test our hypothesis, we experiment with different pre-trained language models and evaluate multiple methods of boosting the model's performance via domain pre-training and data augmentation. Our final submission ranked 8 out of a total of 32 teams in the official rankings with a Pearson correlation of 0.771 - a score difference of 0.047 compared to the top ranked submission.

This paper is organized as follows: In section 2 we describe the official datasets as well as the external datastes that were used. In section 3 we present our baselines and systems' setups. Section 4 presents our data splits and model training specifics. We show and analyse our results in section 5, then we conclude and discuss future work in section 6.

2 Datasets

The used datasets are categorized into two categories: 1) Task Datasets which are provided by the task organizers and 2) External datasets that are used as additional assets for model training.

Pair	Training	Evaluation
ar-ar	263	298
de-de	832	611
de-en	532	190
de-fr	-	116
de-pl	-	35
en-en	1689	235
es-en	-	498
es-es	506	243
es-it	-	320
fr-fr	69	111
fr-pl	-	11
it-it	-	442
pl-en	-	64
pl-pl	333	224
ru-ru	-	287
tr-tr	419	275
zh-en	-	223
zh-zh	-	769
Total	4643	4953

Table 1: Count of training and evaluation samples by language-pair. "-" means not present in training data.

2.1 Task Dataset

The task organizers provided a number of 4,964[1] news article pairs along with their "Overall Similarity" scores to be used for training purposes. Multiple trained human evaluators were asked to evaluate pairs of news articles and provide similarity scores for different sub-dimensions as well as an overall score; all in 1-4 range. The final scores are calculated by averaging the individual scores across all the evaluators. The training set contains same language pairs as well as cross-lingual pairs. The evaluation dataset was created in the same way as the training data however it contains new languages that are not present in the training set and the language pair distribution does not match the training set distribution as shown in Table 1.

2.2 External Datasets

We made use of monolingual datasets from multiple news sources to fine-tune the pre-training phase of the language models that we experimented with (see section 3 for the details). NADiA dataset (Al-Debsi et al., 2019) is used for Arabic, CCNEWS dataset (Hamborg et al., 2017) is used for English, Global Voices news data (Tiedemann, 2012) is used for Polish and MLSM (Scialom et al., 2020) dataset's input text is used for German, Spanish, French and Turkish.

[1]Task organizers published article URLs but only 4643 pairs were retrievable by the time we scrapped them

3 System Overview

All of our systems are solely trained on the text descriptions of input articles as features and their similarity score as the target variable. All the presented systems except for the baseline system 3.1, are based on XLM-RoBERTa (XLM-R). This choice is made based on the fact that it's a multilingual model that is pre-trained on large amounts of data spanning 100 languages and performs competitively on several cross-lingual transfer tasks (Conneau et al., 2019). The general architecture of our systems itself is kept relatively simple i.e. a Language Model with a regression head on top.

3.1 Baseline

Our baseline system is a multi-variate linear regression model that uses 2 independent variables: a) count of the named entities[2] that are shared between the two news articles and b) cosine similarity between the sentence embeddings of the news article pair. Formally speaking; We model this as $Y = A + B_1X_1 + B_2X_2$ where X_1 and X_2 are the aforementioned variables and Y is the similarity score. The model is trained to minimize the Mean Square Error (MSE). We also experimented with a Gamma Function for regression on the non-negative similarity values, however no difference was perceived in Pearson's scores. We evaluated sentence embeddings generated using LaBSE (Feng et al., 2020), MPNET (Song et al., 2020) and SBERT (Reimers and Gurevych, 2019) and reported baseline results using LaBSE embeddings since it resulted in the least MSE.

3.2 XLMR

Our first system dubbed *XLMR* is an XLM-R model with a regression head on top which is trained to minimize the MSE on the task dataset 2.1. The input is formed by concatenating the text of the two input articles and placing a special token in between.

3.3 XLMR-Pre

Our second system dubbed *XLMR-Pre* follows the same architecture as *XLMR* however, we continue model pre-training using the Masked Language

[2]Exact lexical matching

Pair	Original	After
ar-ar	263	1607
de-de	832	2176
de-en	532	532
en-en	1689	1689
es-en	-	1344
es-es	506	1850
es-it	-	1344
fr-fr	69	1413
it-it	-	1344
tr-tr	419	1763
pl-pl	333	1677
ru-ru	-	1344
zh-zh	-	1344
zh-en	-	1344
Total	4643	12707

Table 2: Count of training samples by language-pair before and after data augmentation by translation. Please note that only samples from the training split were translated to avoid any *potential* data leakage. "-" means not present in training data.

Modelling (MLM) objective (Devlin et al., Liu et al.) for languages in the training set using the collected external datasets 2.2.

3.4 XLMR-Aug

Our third system dubbed *XLMR-Aug* follows the same setup of *XLMR* however the training data is supplemented with data augmentation. Synthetic data is created in two different ways namely: pair switching and Machine Translation (MT). Pair switching is achieved by switching the text concatenation order of the input pairs to act against pair order bias. Machine Translation is leveraged to address the fact that the majority of the training set pairs are in the same language (4500 pairs) however the systems are evaluated on their ability to score cross-lingual pairs as well. A number of 1344 English pairs are sampled and translated to different languages. Additionally a number of 667 German language examples are translated into English to encourage improvements in the English language pairs. Translated pairs' statistics are reported in Table 2. The used MT system is an in-house general purpose Transformer Big model (Vaswani et al., 2017) that is not adapted to any specific domain.

3.5 Ensemble Systems

Ensemble systems are developed by averaging the individual scores of different combinations of our three systems: *XLMR*, *XLMR-Pre* and *XLMR-Aug*.

4 Experimental Setup

4.1 Data Setup

To be able to run our model selection experiments and validate our hyper-parameter settings, a standard split of 80:10:10 is applied to the task dataset 2.1 to split it into train, validation and held-out sets. This setting resulted in a training set size of 3719 samples, validation set size of 464 and a held-out set size of 460 samples. The validation set was used for hyper-parameter tuning and the held-out set was used for system comparisons since there were no datasets provided for such purposes by the task organizers. Our experiments that leveraged data augmentation techniques made use of a total of 12707 training samples. To avoid any data leakage, we only used samples from the train set to augment the data.

4.2 Hyper-parameters

Model hyper-parameters were initially setup with the recommended values for XLM-R model fine-tuning (Conneau et al., 2019) and were manually fine-tuned based on the correlation scores and the loss on the validation set. *XLMR* and *XLMR-Pre* systems were trained for 4 $epochs$ and *XLMR-Aug* system was trained for 10 $epochs$. In all the experiments, an $AdamW$ optimizer (Kingma and Ba, Loshchilov and Hutter) was used with a linear schedule, a $learning\_rate = 2e - 5$, $epsilon = 1e - 8$ and training and validation $batch\_size = 8$.

4.3 Training

The Huggingface transformers library[3] was used to conduct all our model training experiments and all our models were initialized using xlm-roberta-large[4] weights. All models were trained using 8 Nvidia Tesla-V100 GPUs.

4.4 Evaluation

We evaluated our models using Pearson's correlations score on the overall test set. Additionally, we conducted a per-language correlation scoring for better reasoning and model development.

[3]https://github.com/huggingface/transformers

[4]https://huggingface.co/xlm-roberta-large

System	Score
Baseline	0.677
XLMR	0.808
XLMR-Pre	0.804
XLMR-Aug	0.790

Table 3: Pearson's scores for different systems during model development on the held-out set

5 Results and Analysis

During the development phase, we used the fixed held-out set to compare the performance of different systems. When the evaluation phase ended, the gold labels were made available for the evaluation set and thus we re-evaluated all our systems using that set as well. The results on the held-out set are shown in Table 3 and the results on the evaluation set are shown in Table 4. A per language breakdown scoring is also provided on both the held-out and the evaluation sets in Table 5 and Table 6 respectively.

Our submitted system is ES_2S_3 (table 4) which is an ensemble of *XLMR-Pre* and *XLMR-Aug*, however our post-evaluation analysis showed that $ES_1S_2S_3$ which an ensemble of our three systems performs slightly better than our official submission with a marginal increase of 0.004 in the correlation score. We attribute this increase to the power of ensembling given that the three systems were competitive to each other in terms of the aggregate performance scores however each system has it's own strengths when it comes to language specific performances as shown in Table 6. A little inconsistency between the scores of the different systems on the held-out and evaluation sets is attributed to the fact that the evaluation set has unseen language pairs and a radically different language distribution compared to the training and held-out distributions.

Our per language evaluation (Table 6) reveals explainable patterns. *XLMR-Pre* performs the best on fr_pl and fr_fr language pairs due to abundance of French pre-training data in this model. *XLMR-Aug* performs the best in 12 out of 18 language pairs due to it's MT augmentation that boosts it's performance on unseen pairs. An ensemble of *XLMR* and *XLMR-Pre* performs the best for en_en pairs due to the bias of the original XLM-R model towards English, the news domain fine-tuning and the lack of translation noise or parameter sharing competition with other languages.

System	Score
Baseline	0.615
XLMR (S_1)	0.752
XLMR-Pre (S_2)	0.755
XLMR-Aug (S_3)	0.753
ES_1S_3	0.768
ES_1S_2	0.767
ES_2S_3	**0.771***
$ES_1S_2S_3$	**0.775**

Table 4: Pearson's scores for different systems on the evaluation set. ES_iS_j is an ensemble of S_i and S_j. * *indicates our best submitted system*

Pair	XLMR	XLMR-Pre	XLMR-Aug
ar_ar	0.603	**0.717**	0.606
de_de	0.810	0.788	**0.838**
de_en	0.862	0.862	**0.899**
en_en	0.818	**0.827**	0.764
es_es	**0.914**	0.861	0.906
fr_fr	**0.812**	0.762	0.682
pl_pl	**0.709**	0.622	0.579
tr_tr	0.823	0.782	**0.841**

Table 5: Language pair wise Pearson's scores for different systems on the held-out set

6 Discussion

In this paper we described our submissions to the news similarity task in SemEval 2022. Our models showed competitive performance by leveraging pretrained language models and showed that further improvements can be gained by the use of domain pre-training and data augmentation using machine translation. Due to the competition time limits such domain pre-training and translation experiments were conducted on relatively small datasets and we did not manage to experiment with a model that combines both additions. We believe that scaling these approaches by using huge amounts of monolingual data across different languages is potentially a direction that is worth exploring.

We see improvement posibilities when it comes to modeling as well. In the early stages of our experimentation we tried a contrastive learning based approach similar to the works done by (Chopra et al., 2005) though, initial results were not promising and we decided to discard this direction, we believe that further efforts can be fruitful. We've also experimented with explicit modeling of Named Entities within our models without a positive outcome

Pair	XLMR (S_1)	XLMR-Pre (S_2)	XLMR-Aug (S_3)	ES_1S_3	ES_1S_2	ES_2S_3	$ES_1S_2S_3$
ar_ar	0.784	0.790	0.774	0.797	0.805	0.805	**0.809**
de_de	0.752	0.753	0.730	0.757	0.766	0.763	**0.768**
de_en	0.795	0.797	0.741	0.785	**0.809**	0.793	0.802
de_fr	0.559	0.528	0.583	**0.592**	0.564	0.586	0.590
de_pl	0.673	0.713	0.667	0.701	0.721	0.720	**0.725**
en_en	0.780	0.791	0.756	0.779	**0.795**	0.786	0.791
es_en	0.807	0.810	0.794	0.816	0.821	0.819	**0.824**
es_es	0.819	0.813	0.813	0.828	0.826	0.829	**0.833**
es_it	0.718	0.717	0.744	0.752	0.738	0.750	**0.754**
fr_fr	0.834	0.847	0.818	0.837	**0.848**	0.845	0.847
fr_pl	0.853	**0.943**	0.846	0.862	0.911	0.908	0.898
it_it	0.788	0.766	0.763	0.786	0.790	0.781	**0.790**
pl_en	0.632	0.615	**0.712**	0.709	0.659	0.703	0.705
pl_pl	**0.679**	0.655	0.643	0.672	0.678	0.663	0.675
ru_ru	0.704	0.678	0.718	**0.724**	0.703	0.717	0.719
tr_tr	0.810	0.814	0.804	0.824	0.827	0.830	**0.833**
zh_en	0.684	0.689	0.758	**0.763**	0.715	0.763	0.762
zh_zh	0.729	0.725	0.739	0.748	0.741	0.750	**0.752**

Table 6: Language pair wise Pearson's scores for different systems on the evaluation set. ES_iS_j is an ensemble of S_i and S_j

however this could be due to the fact that we used a very simple string matching approach for named entities identification. Another modeling aspect is the train/evaluation language distribution modeling. Given that the distribution of evaluation language pairs are available, one could leverage this to improve the model optimization process.

Finally, in this exploratory work we haven't made use of any available article related meta-data which can have strong predictive power of article similarity. Examples include URL normalization to identify parallel articles in different languages, domain and country information among other features. We leave out these territories to be explored in future works.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Ridhwan Al-Debsi, Ashraf Elnagar, and Omar Einea. 2019. Nadia: News articles dataset in arabic for multi-label text categorization.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

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

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT sentence embedding. *CoRR*, abs/2007.01852.

Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. 2017. news-please: A generic news crawler and extractor. In *Proceedings of the 15th International Symposium of Information Science*, pages 218–223.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. MLSUM: The multilingual summarization corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8051–8067, Online. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *CoRR*, abs/2004.09297.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

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

DartmouthCS at SemEval-2022 Task 8: Predicting Multilingual News Article Similarity with Meta-Information and Translation

Joseph Hajjar[1], Weicheng Ma[2], and Soroush Vosoughi[3]

[1,2,3]Department of Computer Science, Dartmouth College
[1]joseph.h.hajjar.gr@dartmouth.edu
[2]weicheng.ma.gr@dartmouth.edu
[3]soroush.vosoughi@dartmouth.edu

Abstract

This paper presents our approach for tackling SemEval-2022 Task 8: Multilingual News Article Similarity. Our experiments show that even by using multi-lingual pre-trained language models (LMs), translating the text into the same language yields the best evaluation performance. We also find that stylometric features of the text and meta-information of the news articles can be predicted based on the text with low error rates, and these predictions could be used to improve the predictions of the overall similarity scores. These findings suggest substantial correlations between authorship information and topical similarity estimation, which sheds light on future stylometric and topic modeling research.

1 Introduction

Given a pair of news articles in arbitrary languages, the objective of SemEval-2022 Task 8 is to predict whether the two articles cover the same story (Chen et al., 2022). While the task falls in the category of topical similarity estimation, traditional statistical topic models may not be appropriate due to the large vocabulary size and the difficulty of matching representative "topic words" across languages. For example, determining whether two articles are about the same news story requires answering questions such as what the article is talking about, who is involved, and where did it occur. We demonstrate this by applying an LDA model (Blei et al., 2003) on the original dataset, and using the LDA representations of news articles in a logistic regression (LR) model yields a Pearson correlation of 0.193 on the evaluation dataset.

Due to the difficulty of obtaining gold standard topical similarity annotations, prior research on this topic is mostly conducted under an unsupervised setting. For example, Bisandu et al. (2018) model text using n-gram features and cluster articles using an improved square root cosine similarity measure. Singh and Singh (2021) use the similarity of news headlines to approximate similarities of news articles. However, their approaches rely primarily on simple statistical features on the word level, which is not proper for our task. Rupnik et al. (2016) use an external event list as an alias of "topic words" with greater granularity to solve the topical similarity estimation task. While this idea could be well adapted to our task, engineering the list of events is effort-consuming and lacks generalizability to unseen data, since the news articles may not always talk about a singular, distinguishable event.

Our models are based on the pre-trained multilingual BERT (mBERT) model (Devlin et al., 2018), one of the most competitive models on multi-lingual natural language processing (NLP) tasks. Though mBERT could handle all the languages in the challenge dataset, we find that translating both news articles in each instance into the same language improves the performance of the model by 0.024 in Pearson correlation coefficients. This is counter-intuitive, since translating the text into another language unavoidably harms its syntactic quality. We hypothesize that the improvements result from the match of named entities across documents when they are translated into the same language.

In addition to text translation, we find that the intermediate scores of each instance that are related to stylometric features of the text and meta-information of the news articles could be predicted by the an mBERT model with high Pearson correlation scores (0.71 to 0.88). With a simple LR model, the six predicted intermediate scores (i.e., geography, entities, time, narrative, styles, and tone) can be aggregated to make more accurate predictions of the overall similarity scores. We speculate from this result that stylometric features are important for aligning the events and named entities across news articles, while the meta-information of these articles provides hints about whether the two articles are talking about completely irrelevant events. We additionally try injecting the predicted intermediate scores into the text encodings produced by mBERT. However, the performance of the ensemble model drops below the pure mBERT-based approach. This is expected since the predicted scores are not dimensionally compatible with text encodings. We leave for future research the problem of efficiently boosting the text encodings with stylometric features and meta information predicted from the text.

Our approach ranked the 10th in the all-language data challenge and the 6th most successful results in the English-only challenge. The main areas which impacted our model's performance were the languages and the lengths of the articles. Shorter articles and certain languages proved to confuse the model.

2 Dataset

Each sample in the training data for the task included a url and language code for each article and a score for the pair's overall similarity, along with a score for each of six stylistic features and news meta-information (i.e., geography, time, shared entities, shared narratives, style, and tone). The feature scores and overall similarity were calculated by averaging several annotators' scores and were rated on a scale of 1 to 4, where 1 indicated low similarity and 4 implied high similarity. Each pair of news articles is annotated by 1 to 8 annotators, with the majority of instances annotated by 1 to 3 annotators. In carrying out experiments, we split the released training data into two sets, 80% for training and the remaining 20% for validation. The evaluation dataset is left out for testing purposes only.

2.1 Scraping Article Contents

Since the dataset provides links to news articles, we scrape the text using the newspaper3k library [1]. Two links were provided for each article, one pointing to the article's most up to date link and another pointing to an Internet Archive site containing the earliest available version of the article. The analysis was meant to be carried out on the content of the earliest version of the article, however if that link was unable to be scraped, then the most up to date version of the article was used as the article's content. It is noteworthy that in 854 training instances, at least one news article cannot be retrieved from either link. Removing these data which were not usable resulted in a training dataset containing 7049 instances. Similarly, in the evaluation data, there were 31 instances where at least one news article could not be retrieved from either link.

[1]https://github.com/codelucas/newspaper

2.2 Training Data and Evaluation Data

Training data were released in two batches, where the first batch contained 2,939 pairs of news articles and the second batch contained 4,964 pairs of news articles.

A key difference between the first and second training datasets was that the second dataset included 577 pairs with articles in different languages (we refer to these as bilingual pairs in the rest of the paper) whereas in the first dataset, articles were paired with other articles in the same language. Additionally, the evaluation data had 1440 bilingual pairs, and the bilingual pairs in question had more variety in the languages appearing in the pairs.

The languages used in the training data were, in order of most to least prevalent: English, German, Spanish, Polish, Turkish, French, and Arabic. The only bilingual pairs in the training data were a set of article pairs where one article was in German and the other was in English. The evaluation data had the same languages as the training data, along with Russian, Italian, and Chinese (zh: ISO 639-1).

3 Experimental Setup

All our models are implemented based on the Multi-Task Deep Neural Networks for Natural Language Understanding (MT-DNN) (Liu et al., 2019) model. Specifically, we use the pre-trained mBERT model with 12 layers and 12 attention heads on each layer. For data pre-processing, we separately truncate or pad the two news articles in each instance to 512 words, join them with a "[SEP]" token, and tokenize the text with the mBERT tokenizer. The regression head of MT-DNN is applied on top of the last-layer output of mBERT to generate predictions.

For each approach we present, we fine-tune the mBERT model for 5 epochs with early stopping. A batch size of 8 and a learning rate of 5×10^{-5} are used for all the settings. These are default hyperparameters from the MT-DNN library. Additionally, we implement a bidirectional LSTM (BiLSTM)

(Hochreiter and Schmidhuber, 1997) model as our baseline, using exclusively the text of the articles as input. We set the dimension of hidden states to 100 for this model, and we train the model for 20 epochs with a learning rate of 0.001 and a batch size of 512 in all the experiments. Our BiLSTM model achieves a Pearson Correlation of 0.277 on the validation set.

We repeat all the evaluations three times with different random seeds (i.e., 0, 1, and 2), and we report the mean Pearson correlation coefficient and the standard deviation of scores. All the experiments are run on a single RTX-6000 graphics card.

3.1 Translating Articles

As mentioned in Section 2, there were a substantial number of bilingual article pairs in the training and particularly in the evaluation data. Since some of our approaches require translating both articles in each instance into the same language, we adopt googletrans[2] to carry out the translations. We experimented with settings where the articles are translated (1) into English, (2) into the language of the first article, or (3) into the language of the second article, and in the evaluations we used the setting under which each model performs the best on the validation set.

3.2 Approaches

Figure 1 visualizes the three approaches we tried in the challenge.

3.2.1 LR-Based Feature Ensemble Model

In our best-performing model, the mBERT model is used to encode the text and predict six intermediate scores provided in the training dataset. An LR model with the intermediate scores as input is then used to predict the overall similarity scores. The scikit-learn (Pedregosa et al., 2011) implementation of the LR model is used in our model.

3.2.2 Text-Based Regression Model

For this approach, we rely purely on the text encoding ability of mBERT to make topical similarity predictions. Under this setting, we remove all the meta-information or stylistic clues of the documents from the input and directly fine-tune the vanilla mBERT model to predict the overall similarity scores.

[2]https://github.com/ssut/py-googletrans

3.2.3 Feature-Injected mBERT Model

Since the meta-information and stylistic features of the news articles have proven to be useful in Section 3.2.1, we attempt to combine these intermediate predictions with text encodings produced by mBERT. Specifically, in each instance, we predict the six intermediate scores and concatenate these to the last-layer hidden state of mBERT to make predictions on feature-enhanced text representations. Other parts of the mBERT model remain unchanged. Different from the ensemble model, the feature-injected model is end-to-end.

4 Results and Analysis

The efficacy of each approach was judged on the Pearson Correlation between the predicted similarity and the ground truths (Freedman et al., 2007). With this metric, scores closer to 1 and -1 imply the strongest positive and negative correlations possible, and scores close to 0 imply poor to no correlation.

A summary of the scores can be found in Figure 2. The three approaches worked most successfully when used in tandem with translations. Here we focus on the best performing translation setting for each of the three approaches described in Section 3.2 and compare it to the same approach applied to non-translated data.

4.1 Impact of Language Unification

For our LR-based model, the best performing setting was translating the bilingual pairs in the training data and the evaluation data to the first language of the pair. As discussed in Section 3, we ran this setting with three different seeds, and the approach yielded a mean of 0.746 with a standard deviation of 0.00210. This represented an increase in the correlation by 0.024 when compared to the same setting with no translation.

The next best performing approach was our text-based regression model, with translations to the first language in the bilingual pair. The vanilla mBERT model yielded a mean correlation of 0.737 and a standard deviation of 0.00398. Translation in this setting proved to increase the correlation of the scores by 0.020.

Lastly, the injected features approach performed best when the bilingual pairs were translated to English, and yielded a mean score of 0.737 with a standard deviation of 0.00642. Although this approach scored the lowest among the three, trans-

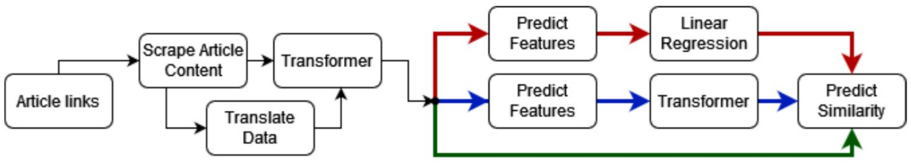


Figure 1: A visualization of three different models we explore in the challenge. The red line shows the approach outlined in Section 3.2.1, the green line shows the approach outlined in Section 3.2.2 and the blue line shows the approach outlined in Section 3.2.3

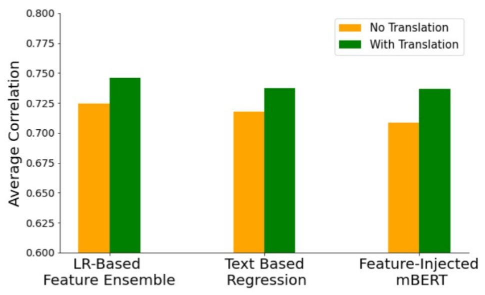


Figure 2: Pearson correlation coefficients achieved by our models on the evaluation dataset. For each model, the green bar represents the score achieved when both documents in each instance are translated to the same language.

lation proved to be the most useful when applied in this setting, with an increase of 0.29 when compared to injecting the features without translation.

Due to the nature of the task, i.e., determining whether articles speak about the same event, it is reasonable to infer that translating articles into the same language boosted performance because of the alignment of named entities in the same language. Although mBERT can manage all of the languages used in the dataset, named entities may be harder to represent between languages, and given their significance in determining article similarity, it is likely that their translation into the same language was the operative factor in enhancing performance.

4.2 Error Analysis for the LR-Based Feature Ensemble Model

When examining the samples where our LR-based model's predictions were furthest from the ground truth annotations, the length in the pair of articles proved a significant factor. The shorter the length of the articles, the more difficult it was for the model to produce accurate predictions, and conversely, samples with longer article lengths tended to yield higher correlation. The shorter articles on which the model was failing tended to be articles where

Langs	Overall (%)	Outliers (%)	Short (%)
ar - ar	6.08	15.85	18.07
zh - zh	15.69	22.95	48.27
ru - ru	5.85	11.47	4.70
pl - pl	4.57	7.10	1.65

Table 1: Percentage of same-language instances in the entire evaluation data (Overall), in the subset of evaluation data where our best model performs at least 1.5 points off the gold standard labels (Outliers), and in the subset of evaluation data where at least one article in a pair is at most 80 characters in length (Short).

we had only been able to scrape the headline of the article.

For example, our model incorrectly predicted that the following two articles were dissimilar: (1) “释放激发制造业活力稳定经济增长-商会频道-长城网” and (2) “稳外贸工作座谈会释放利好一揽子新举措待发–财经–人民网” (translated: “Release to stimulate the vitality of the manufacturing industry to stabilize economic growth - Chamber of Commerce Channel - Great Wall Network”, “Symposium on Stabilizing Foreign Trade Unleashes Favorable Package of New Measures to Be Launched”). In this case, the articles are lacking in substance to investigate their similarity. Instead, the model has to rely on the articles' headlines which contain far less information.

Another likely influence on the model's prediction was the language on which it was predicting. Although our approach leveraged multilingual models, there were certain languages which tended to yield poor outcomes. The most notable were Chinese-Chinese, Arabic-Arabic, Polish-Polish, and Russian-Russian pairs. Some of these could be explained by the prevalence of certain language pairs in the subset of evaluation data where article lengths were particularly short. This is detailed in Table 1. Thus, it is difficult to determine whether the length of the article or the language is the operative factor in the poor predictions. It

Instances	LEN-A1	LEN-A2
Good preds	1427	1354
Poor preds	394	507

Table 2: The median number of characters in Article 1 (LEN-A1) and Article 2 (LEN-A2) for each instance of the evaluation datasets where the error margin achieved by our best model is small (Good preds) or large (Poor preds). We regard the instances where the predictions are within an error margin of 0.2 points as good predictions and those whose error margins are above 1.5 points as poor predictions.

is important to note, however, that in the case of articles written in Chinese, each character encodes substantially more information than a character in the Latin, Arabic, or Cyrillic alphabets. Nevertheless, it is noteworthy that two of the four languages which confused the model were those which it had not seen in the training data.

4.3 Inter-Approach Error Analysis

Our LR-Based model tended to outperform the text-based regression and feature-injected mBERT models when the article pairs spoke about the same news story but with different entities as subjects. For example, in one pair of articles, both articles referred to tableau proposals for India's Republic Day celebration being rejected, however one speaks about the proposal from the Maharashtra and the other about the West Bengal government proposal. Due to the articles being about a similar event, they contain similar phrases, such as "Twenty-two proposals, 16 from states and union territories and six from central ministries — out of a total 56 have been short-listed for this Republic Day parade" and "the Ministry of Defence has selected 22 tableaux out of 56 proposals for the Republic Day parade." This pairing was given a similarity score of 1.5; our LR-based model predicted 1.48, our text-based regression model 2.42, and our feature-injected mBERT 2.67. Because the latter two models rely more heavily on textual data than the former, the articles' similar phrases likely inflated their scores while the LR-based model was more greatly influenced by the pair's features. In this case, both articles were written in English.

Conversely, our LR-based model underperformed when the textual similarity correlated more with the score than the features did. In one case, one article refers to Austrian MPs who are sick with COVID-19 and the other compares Austria and the EU's approach to dealing with MPs who are sick with COVID-19. Similar phrases and words appear in both articles, such as the pair "That will depend above all on whether the government submits corresponding requests to Parliament", "Now the President wants to discuss Parliament's timetable", and the pair "how MPs who have tested positive could also take part in votes", "Ten MPs of the ÖVP ... are in self-quarantine." This pairing was scored 3.0 for overall similarity, however the feature scores for this pair are 1.0, 2.0, 1.0, 4.0, 1.0, 1.0 for geography, entities, time, narrative, style, and tone, respectively. Our LR-based model predicted 1.95, our text-based regression 2.63, and our feature-injected mBERT 2.87. It is reasonable to conclude that the overwhelmingly low features scores contributed to the large difference in our LR-based model's prediction and the annotated score, while the textual similarity of the pair improved the scores for the latter two models.

5 Conclusion and Future Work

This paper describes the model we use for tackling SemEval-2022 Task 8: Multilingual News Article Similarity. Our work shows that while the vanilla mBERT model could greatly outperform shallower baseline models (e.g., BiLSTM and LDA) on this task, introducing intermediate prediction objectives (e.g., stylometric features and meta-information of news articles) helps improve the performance of mBERT noticeably. Additionally, we find that unifying the languages of news articles in each training and evaluation instances has a positive effect on the ensemble model's performance. We speculate that translating both articles in an instance into the same language helps the model align similar named entities across articles, which is important for assessing whether a pair of articles focus on the same set of events or entities.

Since it is shown by our experiments that stylometric features are contributive for topical similarity estimations, future work can extend our ensemble model by involving a richer list of writing-style-related linguistic features.

References

Desmond Bala Bisandu, Rajesh Prasad, and Musa Muhammad Liman. 2018. Clustering news articles using efficient similarity measure and n-grams. *International Journal of Knowledge Engineering and Data Mining*, 5(4):333–348.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

David Freedman, Robert Pisani, and Roger Purves. 2007. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*.

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

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jan Rupnik, Andrej Muhic, Gregor Leban, Primoz Skraba, Blaz Fortuna, and Marko Grobelnik. 2016. News across languages-cross-lingual document similarity and event tracking. *Journal of Artificial Intelligence Research*, 55:283–316.

Ritika Singh and Satwinder Singh. 2021. Text similarity measures in news articles by vector space model using nlp. *Journal of The Institution of Engineers (India): Series B*, 102(2):329–338.

Team Innovators at SemEval-2022 for Task 8: Multi-Task Training with Hyperpartisan and Semantic Relation for Multi-Lingual News Article Similarity

Nidhir Bhavsar*
Navrachana University
Vadodara, India
18103488@nuv.ac.in

Rishikesh Devanathan*
IIT Patna
Patna, India
rishikesh_2001cs85@iitp.ac.in

Aakash Bhatnagar*
Navrachana University
Vadodara, India
18124526@nuv.ac.in

Muskaan Singh, Petr Motlicek
Speech and Audio Processing Group,
IDIAP Research Institute, Switzerland
(msingh,petr.motlicek)@idiap.ch

Tirthankar Ghosal
ÚFAL, MFF
Charles University, Czech Republic
ghosal@ufal.mff.cuni.cz

Abstract

This work represents the system proposed by team Innovators for SemEval 2022 Task 8: Multilingual News Article Similarity (Chen et al., 2022). Similar multilingual news articles should match irrespective of the style of writing, the language of conveyance, and subjective decisions and biases induced by medium/outlet. The proposed architecture includes a machine translation system that translates multilingual news articles into English and presents a multitask learning model trained simultaneously on three distinct datasets. The system leverages the PageRank algorithm for Long-form text alignment. Multitask learning approach allows simultaneous training of multiple tasks while sharing the same encoder during training, facilitating knowledge transfer between tasks. Our best model is ranked 16 with a Pearson score of 0.733. We make our code accessible here[1]

1 Introduction

Over the last decade, English has been one of the most dominant languages on the internet. However, the number of non-English websites is rapidly increasing. From 2001-to 11, online use of English increased at a slower rate than that of Spanish, Chinese, etc. Approximately 4 billion people connect to the internet every day, but only half of them access web pages written in English (Pimienta, 2009). This creates a severe problem regarding verifying the integrity of the documentation because most systems in use are enhanced with English as the medium of information delivery.

SemEval has conducted similar tasks on sentence-level semantic textual similarity in the past. The SemEval 2017 Task 1 (Cer et al., 2017) dealt with finding the similarity between sentence pairs of both monolingual and cross-lingual nature. ECNU (Tian et al., 2017) was the best model that used feature engineering and deep averaging network (DAN) (Iyyer et al., 2015).

Furthermore, language-agnostic representation for sentence similarity described in Tiyajamorn et al. (2021) uses meaning embedding to estimate the cross-lingual sentence similarity without using human annotations. It improves the performance of any pre-trained multilingual sentence encoder, even in low-resource languages, with a few thousand parallel sentence pairs.

Inspired by Ham and Kim (2021), we explored the concept of semantically aligned multilingual sentence embedding. It covers the biases induced by monolingual similarity evaluation and multilingual sentence retrieval to generate language-aware embeddings. This method aligns semantic structures across different languages and uses a teacher network to distill the knowledge of pivot languages, thus achieving state-of-the-art STS 2017 multilingual corpora.

Our model is inspired by a multitask training approach using a BERT-based encoder. We use multiple subtasks to improve the overall accuracy score. Our model uses a machine translation model to cope with the articles' linguistic diversity. Additionally, since the average size of each article in the training data is close to 512 tokens, we use a text ranking algorithm that can capture the long-range sentence-level similarity between two documents. The next sections provide a more in-depth exposition.

[1]https://github.com/rdev12/Multilingual-News-Article-Similarity
* First three authors have equal contribution

2 Task Description

The shared task emphasizes finding the similarity of multi-lingual news articles irrespective of the style of writing, political spin, tone, or any other more subjective "design decision" imposed by a medium/outlet. It gives participants access to a cross/multi-lingual dataset that spans over ten languages, including English, German, French, Italian, Polish, Russian, Chinese, Turkish, Arabic, and Spanish. The dataset includes an overall matching score between two different news articles. Additionally, the dataset consists of other dimensionality scores, such as Geo-location, Time, Shared Entities, and Shared Narratives (see table 1). These scores are based on a four-point scale ranging from the most to the least similar.

Table 1 indicates two examples: the first pair of articles shows extreme similarity with a Pearson score of 1.25, and the second pair shows non-similar articles with a Pearson score of 4. The second pair of articles is a cross-lingual pair where one article is in English and another in German. The next section outlines the overview of the model proposed by our team.

3 System Overview

Our system pipeline can be decomposed into four modules i.e., extraction, translation, text ranking, and multitask training.

3.1 Extraction

We extract title, descriptions, meta-description, and text from the JSON files obtained by scraping the news articles from the URLs given in the dataset. In most instances, the description and meta-description are the same, so we merge them by creating an additional field in the dataset called "extra text." The intuition behind this is to provide more context, as the title and descriptions tend to convey the overall message of the news article. This led to an increase in Pearson score by 0.07.

3.2 Machine Translation

Our translation module is based on the OPUS-MT (Tiedemann and Thottingal, 2020), a transformer-based neural machine translation model. This model uses Marian-NMT (Junczys-Dowmunt et al., 2018), a stable production-ready neural machine translation toolbox with efficient training and decoding capabilities. It is pre-trained on freely available parallel corpora collected in the large bitext repository OPUS (Tiedemann, 2012). The pre-trained version of the OPUS-MT model has six self-attentive layers in both the encoder and decoder networks and eight attention heads in each layer. Also, to handle the long-form nature of text articles, we use the chunking technique to segment texts into various chunks and then concatenate these derived chunks with other characteristics.

3.3 Text Ranking

There are many instances where the combined token length for a pair of articles exceeded the length of 512 tokens. Often, there are many irrelevant sentences with no semantic significance. So, these sentences are eliminated since they contribute much less to the overall context of the article. To achieve this, we adopt sentence-level noise filtering approach similar to Pang et al. (2021) & Mihalcea and Tarau (2004). In this, we first concatenate the pair of texts d_a and d_b obtained in the previous sections and split them into their component sentences s_i as shown in equations 1 & 2.

$$d_a = \{s_1^1, s_2^1, s_3^1, ..., s_n^1\} \tag{1}$$

$$d_b = \{s_1^2, s_2^2, s_3^2, ..., s_m^2\} \tag{2}$$

Then we concatenate d_a and d_b into S as shown in equation 3 later we derive representation matrix by taking the mean of the component word embeddings.

$$S = (s_1^1, ..., s_n^1, s_1^2, ..., s_m^2) \tag{3}$$

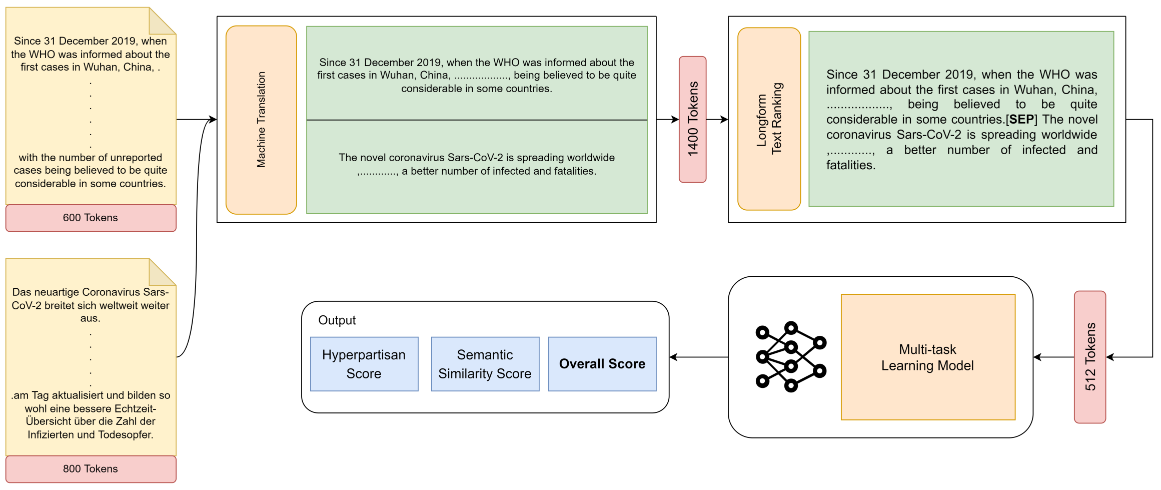
We use fastText embeddings (Bojanowski et al., 2017) which construct a better node similarity matrix than traditional methods and generate the embeddings by capturing transitive relationships and utilizing very sparse random projections. To generate the sentence similarity graph, we calculate the pairwise similarity of the sentence embeddings. The sentence similarity is defined as the same as TextRank (Page et al., 1999) to measure the overlapping word ratio between two sentences:

$$Sim\left(s_i, s_j\right) = \frac{\left|\{w_k \mid w_k \in s_i, w_k \in s_j\}\right|}{\log\left(|s_i|\right) + \log\left(|s_j|\right)}, \quad s_i, s_j \in \mathcal{S} \tag{4}$$

Then, we apply the Page rank algorithm (Page et al., 1998) to calculate the sentence importance score of each s_i in S and sort in decreasing order of importance. Finally, we extract the top λ sentences from d_a and d_b separately such that it is less than 512 tokens when combined. If the two articles

		Geo	Entities	Time	Narrative	Overall	Style	Tone
Pair 1	India approves third moon mission, months after landing failure (link)							
	India targets the new moon mission in 2020 (link)	1	1.25	1	1.25	1.25	1	1
Pair 2	Hong Kong exam question on China and Japan sparked outrage (link)							
	Staatsanwalt wirft Reeder "inszenierte Machenschaften" vor (link)	4	4	3	4	4	1	4

Table 1: Two examples from the SemEval dataset. Pair 1 shows extreme similarity and pair 2 non similairty



Since 31 December 2019, when the WHO was informed about the first cases in Wuhan, China, .
with the number of unreported cases being believed to be quite considerable in some countries.
600 Tokens
Das neuartige Coronavirus Sars-CoV-2 breitet sich weltweit weiter aus.
.am Tag aktualisiert und bilden so wohl eine bessere Echtzeit-Übersicht über die Zahl der Infizierten und Todesopfer.
800 Tokens
Machine Translation
Since 31 December 2019, when the WHO was informed about the first cases in Wuhan, China, ..., being believed to be quite considerable in some countries.
The novel coronavirus Sars-CoV-2 is spreading worldwide ,..........., a better number of infected and fatalities.
1400 Tokens
Longform Text Ranking
Since 31 December 2019, when the WHO was informed about the first cases in Wuhan, China, ..., being believed to be quite considerable in some countries.[SEP] The novel coronavirus Sars-CoV-2 is spreading worldwide ,..........., a better number of infected and fatalities.
512 Tokens
Multi-task Learning Model
Output
Hyperpartisan Score
Semantic Similarity Score
Overall Score

Figure 1: Main model pipeline

are similar, the sentences on the top of the ranked corpora reflect the same. As hypothesized, the title and *extra text* often appear on the top since they are rich in information.

3.4 Multi-task Training

Our multitask approach is based on the architecture proposed by Pruksachatkun et al. (2020). As shown in 2, the model consists of separate task-specific heads with their preprocessing methods but a shared encoder where the weights of all the tasks are updated simultaneously. This approach of introducing multiple subtasks supplements the main task. As evident from table 3, when more relevant tasks are added as task heads, the performance improves. In our best-performing model, the task head consists of an auxiliary semantic similarity task, a hyperpartisan identification task, and the main task. If required, the provided dataset and the hyperpartisan dataset are preprocessed by translation and text ranking modules. The sentences in the two ranked documents are combined, and a separator token separates the documents. This output is then fed into our multitasking model based on DeBERTa. The Loss during training was calculated using Mean Square Error (MSE) Loss function.

4 Experimental Setup

This section describes various hyper-parameters we use in data preprocessing and training. After scraping all the valid URLs, we could access 3651 pairs of articles for training, 408 for validation, and 4902 for a test. For training, we implement simple transformer models. We use DeBERTa (He et al., 2020) as the pre-trained language model with a batch size of 4. We found that the model performs better when the initial learning rate is 10^{-6} to 10^{-5}. We use the AdamW optimizer. We use Google Colab with a Tesla V100 GPU for various experiments. We apply various Python libraries like Pytorch, Transformers, Numpy, and Pandas to implement our multitask learning model.

4.1 Selecting Loss Function

We tried various approaches for selecting the loss function for our model. Initially, we experimented with many different loss functions such as Mean Square Error (MSE), weighted MSE, and dice loss. Furthermore, we experimented with a multi-objective weighted loss function as the data had multiple features. This loss is calculated as the weighted loss of L_E, L_N, and L_O, which are the

https://huggingface.co/microsoft/deberta-base

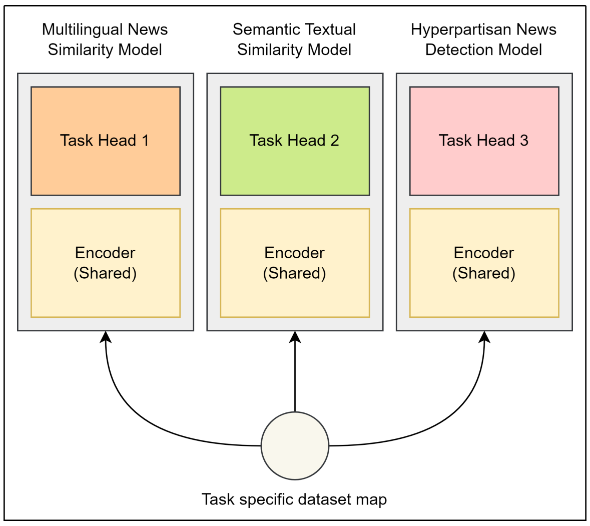

Multilingual News Similarity Model
Semantic Textual Similarity Model
Hyperpartisan News Detection Model
Task Head 1
Task Head 2
Task Head 3
Encoder (Shared)
Encoder (Shared)
Encoder (Shared)
Task specific dataset map

Figure 2: Multi-task Training Model

individual MSE Loss of entity, narrative, and overall similarity prediction. Multi-objective loss can be represented by the equation 5, where α, β, γ are the hyper-parameters decided while experimentation.

$$L = \alpha L_O + \beta L_E + \gamma L_N \tag{5}$$

5 Performance Analysis

In this section, we analyze the performance of several components of our system and compare different schemes for representing the document and the techniques used. Since the commencement of this task, Our team leaned toward using a multi-task learning strategy because it seemed to be the ideal fit for this problem. To do so, we ideated with several subtasks and experimented with our base encoder architecture. Table 3 shows all the different combinations of subtasks and base encoders used. Since the task prompts the participants to use multilingualism as a feature, our initial models utilized multilingual transformers such as XLM-Roberta (Conneau et al., 2019), and RemBERT (Chung et al., 2020), both of which are efficient with multiple languages. However, the primary issue we faced was the cross-linguality of the data; since the delivery style of information varies within every language, it becomes challenging to surface common information among pairs of text articles.

Thus, we emphasize more on translating the articles into English. We utilize a state-of-the-art publicly available model OPUS-MT. This translation module helped all articles to be represented in the same language. Doing so opens up the opportunity for us to choose from multiple encoders instead of our previous approach, where the number of encoders is limited.

After experimentation with various combinations of subtasks, as mentioned in table 3, we see that each of the subtasks has a unique attribute that can contribute to the overall performance of our model. After extensive experimentation, we chose two subtasks along with one main task.

5.1 Maintask - SemEval2022

As described earlier, our approach for generating the overall similarity is quite simple yet effective. The model uses an aggregate of title, text, and descriptions, which is then ranked based on the importance of each sentence in the article. The SemEval 2022 task-8 has various features to consider alongside the overall similarity score, rated between 1- and 4. During the task, the system pipeline supplies a pair of translated and ranked articles to the model, using the Deberta encoder to generate optimally aware document embedding. These embeddings are then supplemented with a set of linear layers to generate a final similarity score.

5.2 Subtask 1 - Hyperpartisan News Detection

As shown in Table 2 we used the Hyperpartisan News Detection dataset for detecting extreme sentences that may be biased towards a political group or a cause. The intuition behind this subtask was distinguishing news articles supporting extreme causes from more general articles. In sporadic cases, these two types of articles show similarities. This task also neglects political biases induced by media outlets and encourages us to treat each pair of articles impartially.

5.3 Subtask 2 - Semantic Textual Similarity

The semantic textual similarity closely relates to our main task since this involves finding the semantic proximity between pairs of texts. Because the STS-b dataset uses small sentence pairs with a max word limit of 40, it improved sentence-level similarity for the main task. The STS-b dataset covers a broad spectrum of sentences from news articles, image captions, and forums, which helps the model diversify across varied sentence-type situations.

5.4 Additional Models

We experimented with different combinations of Transformer based models and datasets. Table 4

Subtask	***Description***	***Dataset***
Semantic Textual Similarity	Determine how semantically similar two pieces of text are.	STS benchmark
Hyperpartisan detection	Given a news article, decide whether it follows a hyperpartisan argumentation, i.e., whether it exhibits blind, prejudiced, or unreasoning allegiance to one party, faction, cause, or person.	Hyperpartisan News Detection (Kiesel et al., 2019)
Stance detection	It involves estimating the relative perspective (or stance) of two pieces of text respective to a topic, claim or issue.	Fake News Challenge - 1 (Hanselowski et al., 2018)
Fake news inference detection	Fake news Detection using the Natural Language Inference. This entails categorizing a piece of text into categories such as "pants-on-fire", "false", "barely true", "half-true", "mostly true", and "true."	Fake news inference dataset
Language Inference	Determine whether a "hypothesis" is true (entailment), false (contradiction), or undetermined (neutral) given a "premise"	DocNLI Dataset (Yin et al., 2021)
Emotion Detection	Our Intuition behind this subtask was that if two articles are similar, they will exhibit same type of emotion. However, after experimentation we found this false as this subtask did not contribute towards the main task.	Go-Emotions Dataset (Demszky et al., 2020)
Paraphrase detection	Determine whether a particular sentence is a paraphrase of the original text.	Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005)

Table 2: Brief description of all different subtasks used for experimentation

Model subtasks	Model Type	Pearson Score (on validation set)
Stance detection	Translated text with RoBERTa	0.800
Stance detection	Untranslated text with XLM-RoBERTa	0.737
No Subtask (Multi-Objective Loss)	Weighted loss on Entity (0.15), Narrative (0.15) and Overall (0.7) Similarity (Sener and Koltun, 2018)	0.815
No Subtask (Multi-Objective Loss)	Weighted loss on Entity (0.2) and Overall (0.8) Similarity	0.811
Stance detection + Hyperpartisan detection	Translated text with RoBERTa	0.809
Hyperpartisan detection + Semantic Textual Similarity	**DeBERTa**	**0.835**

Table 3: Combination of all the subtasks and model types we used in experimentation. The results here are calculated on the validation set.

represents the score we achieved while developing the model. These scores are on the validation dataset that comprises 408 article pairs. Our model performed well during the developing stage. For instance, our best-performing model achieved a Pearson score of 0.835, and subsequently, we achieved a Pearson score greater than 0.8 in two other approaches. Table 3, however, shows the Pearson scores evaluated on the test dataset. We believe that the decline in performance was caused by the new languages (Chinese, Italian, Russian) introduced in the test dataset. These languages were not present in the training or validation set. This could be linked to the shift in writing style imposed on new languages, regardless of translation. According to our validation results, the model could not interpret the new languages and the unique cross-lingual pairs in the dataset.

6 Results

We use the Pearson score to evaluate SemEval 2022 task 8, which measures the linear relationship between the predicted and ground truth values. Like other correlation coefficients, the Pearson score varies between -1 and +1, with 0 implying no correlation. Correlations of -1 or +1 imply an exact linear relationship. The dataset available in the evaluation phase of the task lacked features like Geography, Entity, etc., which were present in the training phase. Furthermore, the test dataset has an additional set of languages not present in the training dataset. This test dataset ensured that the model trained by the participants was an accurate multilingual model. Table 4 shows the models submitted by our team for the SemEval 2022 task-8; our best model achieves a Pearson score of 0.733, placing us amongst the top 15 of all participating teams.

As stated in the second row of table 3 we also experiment with a singleton multitask learning model separate from a machine translation system. The model, however, was unable to generalize adequately for each language in the dataset, and there appeared to be considerable sparsity between the predicted scores, particularly for low-resource languages like Turkish and Polish. Also, the lack of invariant multilingual data, specifically for each of the languages listed previously, was crucial in our decision to switch to a translation-based approach.

Next, table 5 showcases two instances where our model is successful and two instances where it fails to predict the correct value. We believe that this behavior is due to the machine translation module in our pipeline. Our model performs poorly for the following language pairs: de-en, ar-en, pl-en, and zh-en. This can be closely connected to the OPUS-MT model's BLEU score values, which are much lower for the language mentioned above pairs. This

Model Type	Model Subtasks	Pearson Score (on test set)
RoBERTa	Semantic Textual Similarity + Stance Detection	0.724
	Phrase Detection + Stance Detection	0.730
DeBERTa	**Semantic Textual Similarity + Hyperpartisan Detection**	**0.733**

Table 4: Models submitted by our team for the SemEval 2022 Task-8

Article 1	Article 2	Predicted Similarity Score	Actual Similarity Score	Language Pairs
Paraguay: Presidente promulga ley contra el "dinero sucio" en campañas (link)	Paraguay: Deputies approve the law on "dirty money" in political campaigns (link)	1.045	1	de-de
Conductores chocan por detenerse a ver accidente (link)	Paraguay: Deputies approve law on "dirty money" in political campaigns (link)	3.8	4	de-de
Schlag gegen den rechtsnationalen Flügel (link)	AfD-Rechtsaußen unter Druck (link)	1.02	3	es-es
Neue Debatte um Steuern: Millionen Arbeitnehmer zahlen Höchstsatz (link)	Norbert Walter-Borjans wants a higher top tax rate from 76,000 euros (link)	3.8	1	es-es

Table 5: First two rows of this table represents the instances where our model performed up to the mark and the last two rows represent the cases where our model failed to predict the right values

	Pearson Score
Ours	**0.733**
Average	0.624
Best in all teams	0.818

Table 6: Comparison of ours result with the best and the average pearson score

indicates that the model cannot comprehend these languages adequately, resulting in unclear results. Additionally, our system performs exceptionally well for data in French and Spanish, where the BLEU score values were 59.66 and 57.5, respectively. Furthermore, the results on en-en pairs are very accurate since they are not translated, thus retaining the writing style of the editor/outlet. Also, We encountered numerous challenges while scraping the data; not all websites were fully accessible, others only included photos or titles, and a handful swapped the URL with that of the main website's landing page. Next, the scrapper made numerous mistakes in discovering and assigning relevant tags to the articles' various subsections. A common erratum was combining description and title and attributing the title of the continuing piece to some random advertisement or supporting material.

7 Conclusion & Future Work

Our model performs well on English, Spanish, and French data while falling short on German. Even though the German data is the second largest, the biggest problem is that there is a lot of data to consider. The model could not correctly address each attribute of the article, resulting in underperformance. Some of the future work we anticipate to do which can increase the performance of our model are stated below:

1. Better Neural Machine Translation system which can effectively produce English sentences. We do suggest using AWS Translate since it makes more accurate predictions.

2. Using a finely trained multilingual model, enhanced explicitly for dealing with documents. DocMT5 (Lee et al., 2021) is one considered model; however, due to its public unavailability at the time of writing this paper, made us unable to use it in our approach.

Acknowledgements

This work was supported by the European Union's Horizon 2020 research and innovation program under grant agreement No. 833635 (project ROXANNE: Real-time network, text, and speaker analytics for combating organized crime, 2019-2022).

References

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

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity - multilingual and cross-lingual focused evaluation. *CoRR*, abs/1708.00055.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Hyung Won Chung, Thibault Févry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2020. Rethinking embedding coupling in pre-trained language models. *CoRR*, abs/2010.12821.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan S. Cowen, Gaurav Nemade, and Sujith Ravi. 2020. Goemotions: A dataset of fine-grained emotions. *CoRR*, abs/2005.00547.

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

Jiyeon Ham and Eun-Sol Kim. 2021. Semantic alignment with calibrated similarity for multilingual sentence embedding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1781–1791, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Andreas Hanselowski, Avinesh PVS, Benjamin Schiller, Felix Caspelherr, Debanjan Chaudhuri, Christian M. Meyer, and Iryna Gurevych. 2018. A retrospective analysis of the fake news challenge stance-detection task. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1859–1874, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced BERT with disentangled attention. *CoRR*, abs/2006.03654.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Chia-Hsuan Lee, Aditya Siddhant, Viresh Ratnakar, and Melvin Johnson. 2021. Docmt5: Document-level pretraining of multilingual language models. *CoRR*, abs/2112.08709.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Larry Page, Sergey Brin, R. Motwani, and T. Winograd. 1998. The pagerank citation ranking: Bringing order to the web.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

Liang Pang, Yanyan Lan, and Xueqi Cheng. 2021. Match-ignition: Plugging pagerank into transformer for long-form text matching. *CoRR*, abs/2101.06423.

Daniel Pimienta. 2009. Twelve years of measuring linguistic diversity in the internet: balance and perspectives.

Yada Pruksachatkun, Philip Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R. Bowman. 2020. jiant: A software toolkit for research on general-purpose text understanding models. *CoRR*, abs/2003.02249.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. *CoRR*, abs/1810.04650.

Junfeng Tian, Zhiheng Zhou, Man Lan, and Yuanbin Wu. 2017. ECNU at SemEval-2017 task 1: Leverage kernel-based traditional NLP features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 191–197, Vancouver, Canada. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of* *the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal. European Association for Machine Translation.

Nattapong Tiyajamorn, Tomoyuki Kajiwara, Yuki Arase, and Makoto Onizuka. 2021. Language-agnostic representation from multilingual sentence encoders for cross-lingual similarity estimation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7764–7774, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenpeng Yin, Dragomir R. Radev, and Caiming Xiong. 2021. Docnli: A large-scale dataset for document-level natural language inference. *CoRR*, abs/2106.09449.

OversampledML at SemEval-2022 Task 8: When multilingual news similarity met Zero-shot approaches

Mayank Jobanputra [*] **Lorena Martín Rodríguez** [*]

Department of Linguistics, University of Tübingen, Tübingen, Germany
{mayank.jobanputra, lorena.martin-rodriguez} @uni-tuebingen.de

Abstract

We investigate the capabilities of pre-trained models without any fine-tuning, for a document-level multilingual news similarity task of SemEval-2022. We utilize title and news content with appropriate pre-processing techniques. Our system derives 14 different similarity features using a combination of pre-trained `MPNet` with well-known statistical methods (i.e. TF-IDF, Word Mover's distance). We formulate the multilingual news similarity task as a regression task and approximate the overall similarity between two news articles using these features. Our best performing system achieved a correlation score of 70.1% and was ranked 20^{th} among the 34 participating teams. In this paper, in addition to a system description, we also provide further analysis of our results and an ablation study highlighting the strengths and limitations of our features. We make our code publicly available at https://github.com/cicl-iscl/multinewssimilarity.

1 Introduction

Assessing semantic similarity between two given content pieces has become one of the important natural language processing (NLP) tasks. This task can help researchers estimate the quality of their models for many other tasks such as: machine translation (MT), summarization, question answering (QA), semantic search, dialog, and conversational systems. Extending semantic similarity task to a cross-lingual setup can extend the evaluation benefits to cross-lingual tasks as well. Previous works (Agirre et al., 2016; Cer et al., 2017) focus on sentence level cross-lingual semantic similarity. The presented shared task, Chen et al., 2022, proposes a novel problem that focuses on document-level semantic similarity based on news articles.

The multilingual news similarity task contains monolingual as well as cross-lingual pairs of news reports. This setup enables researchers to test their multilingual models on a document-level semantic similarity task. There can be multiple extensions to this, including the clustering of news articles and tracking similarity of news coverage between different outlets or regions.

In this work, we investigate the capabilities of pre-trained multilingual language models (LMs) as well as word-embedding models in this task, without fine-tuning them on the task data. Our solution pipeline combines pre-trained `MPNet` (Song et al., 2020) with well-known statistical methods with well-known statistical methods (i.e. TF-IDF, Word Mover's distance) to derive the semantic similarity features between articles using their *title* and *textual content*. We use these similarity features to approximate overall similarity between article pairs.

Our system performance is encouraging for this task, albeit with room for improvement. In the following sections, we describe our approach and provide a detailed study of the errors made by the system. We also report the results of an ablation study highlighting strengths and limitations of our derived features.

2 Task Setup

2.1 Dataset

The shared task introduced a new dataset consisting of 4964 article pairs in the training set and 4953 article pairs in the hidden test set. The participants were provided with these news articles' URLs and a Python script to scrape the texts. The training data contained an annotated overall similarity score for each pair and other similarity scores corresponding to features such as geography, entities, time, narrative, style, and tone.

The released training data consisted of monolingual pairs in English, German, Spanish, Turkish, Polish, Arabic, and French, and one cross-lingual

[*] Both authors contributed equally.

pair: German-English. The evaluation data contained 4,953 news article pairs. To the languages included in the training data, there were added monolingual pairs in Italian, Russian, and Chinese, and the cross-lingual pairs: German-French, German-Polish, Spanish-English, Spanish-Italian, French-Polish, Polish-English, and Chinese-English. For more details, please refer to table 5 and 6 in the Appendix. These monolingual and cross-lingual news article pairs are annotated on a 4-point scale from most to least similar.

2.2 Evaluation

The overall similarity between the two news stories is the only score used to evaluate system performance. The online scoring system calculates Pearson's correlation between system-generated overall similarity ratings and the gold standard ratings.

3 System Overview

In our approach, we formulate the multilingual news article similarity as a regression task, relying on different similarity features to approximate the overall similarity between two news reports. We choose this approach as the language pair distribution differs significantly between training set and the test set. This setup enables us to investigate the capabilities of pre-trained multilingual language models (LMs) as well as word-embedding models on document-level similarity task without fine-tuning.

We divide the news similarity task into a pipeline of five subtasks: Article Scraping, Preprocessing, Embedding Creation, Feature Calculation, and Inference. Figure 1 illustrates the architecture of our system.

3.1 Article Scraping

We used the script[1] provided by the organizers to download the article content from the web. After multiple tries, we were able to retrieve news content for 4940 out of 4964 training article pairs (see Table 5 in the Appendix). Similarly for evaluation data, we were able to retrieve news content for 4903 out of 4953 article pairs (see Table 6 in the Appendix).

3.2 Preprocessing

Our preprocessing step takes an article as an input and generates a json object containing only the information which is relevant for our approach. As we do not fine-tune any model based on the textual content, data cleaning is an important step for our system. We remove irrelevant content from the data in the following manner:

Copyright text: The article texts sometimes contains information regarding the copyright policy of the news websites. We observed a few cases where the scraper only downloaded the copyright notice instead of the news content. In such cases, copyright content increased or decreased the similarity significantly. To avoid errors in similarity feature calculation, we remove such copyright lines.

URLs: Generally, news reports also link other relevant references in their articles but parsing them to make them useful can be tricky. Moreover, sometimes these can contain unrelated advertisement links. Hence, we remove all kinds of links from the text for our purposes.

Cookies text: Similar to the copyright content, the scraper also ends up downloading text with information regarding the usage of cookies from the website. As this text is irrelevant for measuring content similarity, we remove this information from the article text.

Image captions: News stories can also contain images referring to some event or place that is covered in the news article. Such images are generally captioned with the details of the photographer credits. We find such credits irrelevant for our task and clean them from the article text.

3.3 Embedding Creation:

We use vector representation of article title and text to compute similarity features. We obtain these representations using `MPNet` and `FastText`. We calculate these vector representations once and store it on the disk. This way, we do not have to calculate these representations every time we need them. We utilize a multilingual version (Reimers and Gurevych, 2020) of `MPNet` (Song et al., 2020), fine-tuned on a paraphrasing task using parallel data for 50+ languages for calculating vector representations. We also experimented with language-agnostic BERT (Feng et al., 2020), but we dropped it after initial results.

For the longer text, multilingual `MPNet` model simply truncates the text and returns the representation of the first 512 tokens. We need a way

[1]https://github.com/euagendas/semeval_8_2022_ia_downloader

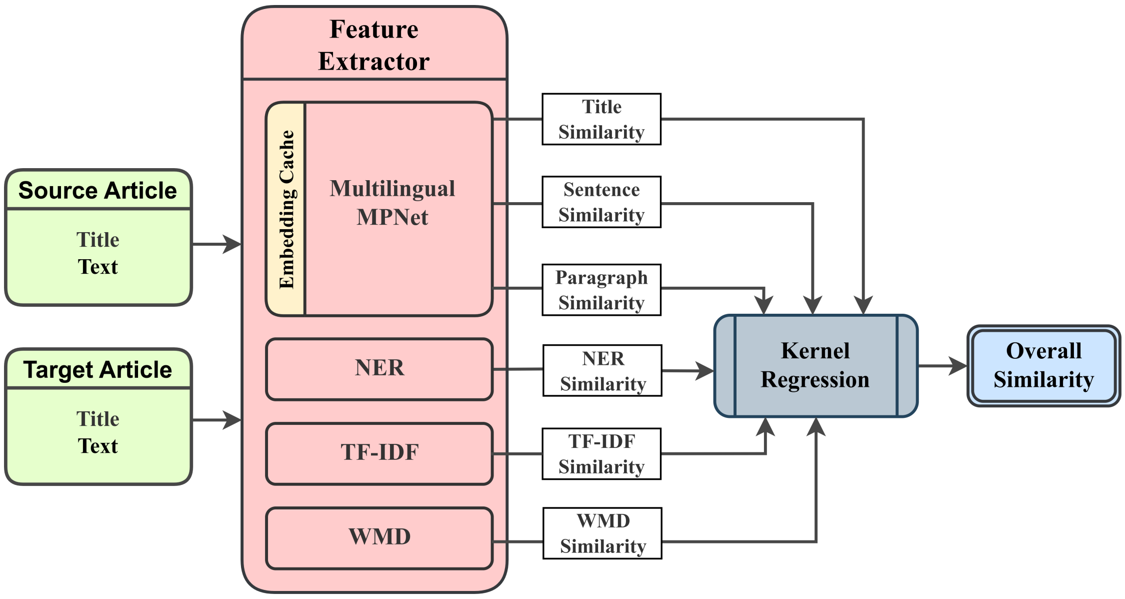


Figure 1: A schematic of our approach. We utilize Zero-shot feature extractors to derive similarity features from the articles. We use these features to train our regression model and obtain overall similarity score.

to be able to compute accurate vector representation of news articles irrespective of their length. Hence, we compute representations for article text at two different granularity levels: sentence-level and paragraph-level.

We obtain separate sentences using a multilingual sentence tokenizer from `SpaCy`.[2] For paragraph-level representation, we tokenize the text using a pre-trained tokenizer, provided along with `MPNet` model, and take the first and the last 512 tokens of the article. We obtain the vector representations of these sentences, the first 512 tokens, and the last 512 tokens using the `SentenceTransformers` (Reimers and Gurevych, 2019) library. Similarly, we compute the vector representations of the news title. From now on, we will refer to the first 512 tokens as the `first paragraph` and the last 512 tokens as the `last paragraph`. Note that for articles shorter than 512 tokens, the first and the last paragraph will be the same.

3.4 Feature Calculation

To estimate overall news article similarity, we derive 14 unique similarity features from the news title and text pairs. In this subsection, we describe all the features in detail.

[2]https://spacy.io/models/xx

3.4.1 Sentence similarity

We derive four similarity features from the article text to capture the sentence-level similarity between articles. These features are obtained as follows:

Sentence mean similarity: The average cosine similarity scores of the top matching sentence vectors between the source and target articles.

Sentence maximum similarity: The maximum cosine score of the top matching sentence vectors between the source and target articles.

Sentence minimum similarity: The minimum cosine score of the top matching sentence vectors between the source and target articles.

Sentence median similarity: The median of the cosine similarity scores of the top matching sentence vectors between the source and target articles.

3.4.2 Paragraph similarity

Additionally, we derive two similarity features from the article text to capture the paragraph-level similarity between articles. These features are obtained as follows:

First paragraph similarity: The cosine similarity value between the `first paragraph` vector representations of the news articles.

Last paragraph similarity: The cosine similarity between the `last paragraph` vector representations of the news articles.

3.4.3 Title similarity

The title should summarize the article text in a meaningful manner. We calculate five similarity features from the article title, capturing title similarity as well as inter and intra title-text similarity between articles. These features are obtained as follows:

Title similarity: The cosine similarity between the title vector representations of the news article pair.

Inter title-text similarity: We calculate inter title-text similarity to see how the source title is similar to the text of the target and vice versa. We utilize stored vector representations of the corresponding news entities and calculate cosine similarity between these entities. Note that we produce two separate features for inter title-text similarity (i.e. $sim(\text{title}_s, \text{text}_t), sim(\text{title}_t, \text{text}_s)$).

Intra title-text similarity: Similarly, to measure title-text coherence, we calculate intra title-text similarity between the title and the text of same news article using the stored vector representations. Note that we produce two separate features for intra title-text similarity (i.e. $sim(\text{title}_s, \text{text}_s), sim(\text{title}_t, \text{text}_t)$).

3.4.4 NER similarity

We calculate named entity similarity NE_{sim} using the below equation.

$$\text{NE}_{sim} = \frac{|\text{NE}_s \cap \text{NE}_t|}{max(|\text{NE}_s|, |\text{NE}_t|)}$$

where NE_s represents set of named entities in the source article, NE_t represents set of named entities in the target article.

3.4.5 TF-IDF Similarity

We first remove stop words using `NLTK`'s language-specific stop words corpus. We then estimate the cosine similarity between TF-IDF representations of the article pair.

3.4.6 WMD Similarity

Similar to TF-IDF similarity, we calculate the Word Mover's distance of two texts without stop words using multilingual `FastText`(Bojanowski et al., 2017) model from `Gensim`.[3]

[3]https://fasttext.cc/docs/en/crawl-vectors.html

3.5 Inference

The last part of our system is estimating the overall similarity using the features obtained. We experimented with three different setups: regression over all the features, multitask regression using additional available scores, and regression over reduced feature space (i.e. principle components, autoencoder representations). In this subsection, we mention all the setups briefly and provide details of our best performing system.

Multitask Regression: The training data contains similarity scores for geography, entities, time, narrative, style, and tone along with Overall article similarity. We trained a Multi-task Lasso model and Multi-task autoencoder on the training set but after initial experiments, we decided not to pursue these setups further.

Regression over reduced feature space: We apply principal component analysis on our feature space for dimensionality reduction.

Regression over entire feature space: We experimented with Linear regression, Decision Tree regression, Random Forest regression, Kernel ridge regression, Multilayer perceptron regression, and TabNet regression (Arık and Pfister, 2021).

Our two best performing systems used Kernel ridge regression. In the highest ranked system, we train a Kernel ridge regressor using a polynomial kernel of degree 3 and a regularization co-efficient of 1.0. This way we allow our model to learn from non-linear features. Our second best performing system uses Kernel ridge regression (KRR) with RBF kernel over top-4 principal components and achieves 70% Pearson correlation. TabNet regression ranked third during the evaluation phase, with a correlation score of 69.1%. We describe the implementation details of our best performing setup below.

Implementation details: We split the data into 95:05 training and evaluation datasets and utilize the entire feature space. Going from 80:20 split to 95:05 split boosted our model accuracy by half point. As a final step, we also clip the model predictions between 1-4 to make sure that our model predictions remain inside the range. We use `scikit-learn` (Pedregosa et al., 2011) for training the kernel regression model and `wandb` (Biewald, 2020) for hyperparameter tuning.

4 Results

All the submissions for this shared task were evaluated with regard to Pearson's correlation coefficient. Our best performing system achieved a score of 70.1% in the evaluation phase. We were officially ranked 20th out of 34 teams on the main task leaderboard. On the English-only subtask, we were ranked 15th out of 34. We report scores for our top three submissions during evaluation phase in Table 1. All three systems use all the features to predict the overall similarity.

Model	Data Split	Score
KRR-poly	95:05	70.1
KRR-rbf	80:20	70.0
TabNet	95:05	69.1

Table 1: Correlation scores of our top performing systems on the hidden test set

We report our results for each monolingual language pairs in Table 2 and cross-lingual pairs in Table 3. The general system performance was similar for monolingual and cross-lingual pairs, with an average accuracy of 0.71 for monolingual pairs and 0.72 for cross-lingual ones.

Language	Score	Language	Score
fr-fr	84.34	tr-tr	70.36
es-es	81.24	zh-zh	64.42
en-en	79.55	ar-ar	62.23
it-it	79.42	pl-pl	61.49
ru-ru	73.50	de-de	59.43

Table 2: Correlation scores for monolingual pairs

Language	Score	Language	Score
pl-en	82.84	fr-pl	74.76
es-en	80.13	es-it	70.95
zh-en	76.91	de-pl	60.11
de-en	76.54	de-fr	55.47

Table 3: Correlation scores for multilingual pairs

The highest accuracy was achieved in the French monolingual pair, with 0.84, but the lowest accuracy score was found in the German-French cross-lingual subsection with 0.55. This can be explained by the general low results that the system achieved in the pairs with news written in German, be it in the monoligual subset (0.59), or cross-lingual pairs German-French and German-Polish (0.60). Only one pair with German language had an above average accuracy, the German-English cross-lingual pair (0.76).

5 Performance Analysis

In this section, we analyze the performance of feature sets used by our system and compare different subsets against each other. We also discuss some errors made by our system and possible improvements.

5.1 Ablation study

We conducted ablation experiments to evaluate the importance of different feature sets on the results. We use released test set labels and our best performing model, Kernel Regression to conduct this study. We report the results of this study in Table 4. Note that these results were obtained after extensive hyperparameter tuning, which was not feasible during the shared task evaluation phase.

Features	Correlation
MPNet features	71.83
Non-MPNet features	47.15
MPNet features + WMD distance	71.35
TF-IDF + WMD distance	47.89
MPNet Sentence features	63.21
MPNet Title features	64.11
MPNet Paragraph feature	63.73

Table 4: Results of feature ablation study

We observe that features derived from `MPNet` perform better than our reported system and slightly improve the correlation score. We also observe that adding other features with `MPNet`-derived features rather degrades the system performance (i.e. `MPNet` features + WMD distance). Other features perform very poorly compared to `MPNet` features through our setup. While `MPNet` feature sets perform substantially better individually compared to other features, combining all `MPNet` gives the best performance on the task.

5.2 Error analysis

In order to closely examine the performance of our system, we analysed a subset of the news pairs which were classified in the wrong category. This subset only included news pairs written in English, German, Spanish, Italian or French and its cross-lingual combinations. Most differences between our systems' result and the annotation guidelines differed in less than one point. Since the different categories of similarity proposed in the dataset also differed in one point, we considered this as our threshold for error analysis.

The following patterns were found when our system overestimated the similarity between two news pairs:

Articles with parallel structures: Some news genres present a more fixed structure than others. Such is the case for police reports, which include similar key phrases but narrate different events.

Same location: Two different events which happened in the same location.

The following patterns were found for when our system underestimated the similarity between two news pairs:

Scraping errors: At least one of the articles did not contain any textual content relevant for the news article.

Lack of information: At least one of the news articles was extremely brief (less than one paragraph), and lacked information about the event.

Different titles: The titles focused on different aspects of the news report.

6 Conclusion

In this paper, we have described our participation in the Task 8 of SemEval-2022, "Multilingual news article similarity". We developed a system to investigate the capabilities of pre-trained language models (LMs) as well as word-embedding models. Our results suggest that the system performs similarly for monolingual and cross-lingual pairs, but its performance varies based on the specific language pairs. The ablation study showcases the strength of the pre-trained multilingual language models for this task. Given the performance of our system, despite the noticeable variation in language-pair distributions, we speculate that our approach can be used to deliver similar results for additional languages as well.

For the future, we would like to explore the system's performance with the same features but also fine-tuning our `MPNet` model on the training dataset. This would allow us to compare the effectiveness of pre-trained models against the fine-tuned model. Additionally, we would like to experiment with new features such as article summary similarity and article topic similarities.

Acknowledgements

We would like to thank Prof. Çağrı Çöltekin for the his inputs and helpful discussions.

References

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 Task 1: Semantic Textual Similarity, Monolingual and Cross-Lingual Evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Sercan O Arık and Tomas Pfister. 2021. Tabnet: Attentive Interpretable Tabular Learning. In *AAAI*, volume 35, pages 6679–6687.

Lukas Biewald. 2020. Experiment Tracking with Weights and Biases. Software available from wandb.com.

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

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity, Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT Sentence Embedding. *arXiv preprint arXiv:2007.01852*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

A Appendix

Language	**#pairs (w\o dup)**	**Downloaded pairs**
en-en:	1800	1773
de-de:	857	853
de-en:	577 (575)	522
es-es:	570	561
tr-tr:	465	428
pl-pl:	349	349
ar-ar:	274	274
fr-fr	72	72

Table 5: Distribution of languages in the training data

Language	**#pairs (w\o dup)**	**Downloaded pairs**
en-en	236	236
de-de	611 (608)	608
de-en	190 (185)	185
es-es	243	243
tr-tr	275	272
pl-pl	224	224
ar-ar	298	298
fr-fr	111	111
zh-zh	769	764
es-en	498 (496)	496
it-it	442 (411)	411
es-it	320	310
ru-ru	287	287
zh-en	223 (213)	213
de-fr	116	111
pl-en	64	64
de-pl	35	34
fr-pl	11	11

Table 6: Distribution of languages in the test data

Team TMA at SemEval-2022 Task 8: Lightweight and Language-Agnostic News Similarity Classifier

Nicolas Stefanovitch
European Commission - Joint Research Centre
nicolas.stefanovitch@ec.europa.eu

Abstract

We present our contribution to the SemEval 22 Shared Task 8: *Multilingual news article similarity*. The approach is lightweight and language-agnostic, it is based on the computation of several lexicographic and embedding-based features, and the use of a simple ML approach: random forests. In a notable departure from the task formulation, which is a ranking task, we tackled this task as a classification one. We present a detailed analysis of the behaviour of our system under different settings.

1 Introduction

Detecting similar news is a key component of media monitoring, such as the one done by the Europe Media Monitor[1], which collects daily about half a million articles in more than 80 languages. This shared task (Chen et al., 2022) allows to study two important challenges that arise in practice: articles can be similar to several extent and in different ways; massive multilingualism requires language agnostic approaches. We report in this paper the result of an experimental approach to tackle this task.

We considered computing simple lexicographic and embedding-based features and using simple ML approach for complexity reasons, having in mind that pair-wise comparison of half a million articles each day is not possible with heavier solution without massive resource cost. There are two important tasks when building such large scale news clustering: determining which pair of articles are worth comparing, and computing the actual similarity between pairs of articles. While this shared task deals only with the second approach, the presented system has been designed to tackle also the first one.

[1]https://emm.newsbrief.eu/

2 System Description

Our system computes several lexicographic- and embedding-based similarity features, which are fed to a standard random forest. We used the following hyperparameters: 100 trees and a leaf split parameter of 3 data points. On the train set, these parameters avoided extreme overfitting to the data, while not degrading significantly the performances. We used 5-folds cross validation in order to choose the hyper parameters.

For each article we consider separately the similarities related to 4 fields of the news item: the title, the description, the first sentence and the snippet. The snippet is constituted of the first 4 sentences comprised within the first 512 characters of the text - everything after was ignored, sentences were not truncated. We present here the set of features that have been computed for each of these fields, and for some cross field (title-description, title-first sentence, description-first sentences), all features are normalized.

Lexicographic measures: we consider two sets of representation: the set of words, and the set of multi-word expressions (MWE). The MWE were computed by splitting the text around stopwords and punctuation marks, essentially similar to RAKE keyword extraction (Piskorski et al., 2021). The features include the proportion of matching words and proportion of matching MWE between both articles. In the case of MWE, a margin of error was allowed in that two non strictly equal MWE were considered equal if the longest common subsequence between them was of 75% the length of the longest one. For both words and MWE we compute the raw count of elements in common and the length of the corresponding span of text.

Embedding-based measures: all embedding-based measures are based on LASER embeddings (Artetxe and Schwenk, 2019) which are aligned multilingual embeddings covering over 100 languages and that perform well for multilingual semantic comparison. They are BiLSTM based, and as such are much faster than transformer-based solutions, it also has to be noted that out of the box BERT solutions do not perform well on the task of semantic similarity, and LASER embeddings have a better performance (Reimers and Gurevych, 2020). On top of the pairwise comparison between the aforementioned fields, we also compute the equivalent of Word Mover Distance (Zhao et al., 2019) between the first 4 sentences of the articles, by binning sentences in pairs of decreasing similarity.

Non linguistic features: the only non linguistic feature considered is the difference in publication date of the two articles.

When developing the system, the results were checked for near misses, by considering all the clusters produced, and checking for missing links between elements of the clusters. This approach was able to catch such misses, including in the training data itself, but it was not used in the final system due to lack of time.

3 Data

The training data provided by the organizers is a list of 4964 article pairs covering 9431 articles in 16 languages, while the 7 main languages represent 99% of the dataset. The pairs are associated with several similarity scores; one of them giving the overall similarity is ranked between 1.0 (maximum) and 4.0 (minimum) and is the only one considered in our approach. Test data contains 4890 pairs over 9715 articles in 10 languages. There is a notable difference between both datasets, in that train data contained almost no pairs of article in different language, while such pairs represented 15% of the evaluation dataset. A major difference was also the introduction of Chinese in the test set.

3.1 Acquisition

The data was given as a list of urls to download, and this proved to be a daunting exercise fraught with difficulties and eventually taking more time and dedication than the development of the system itself. The scrapper provided by the organizer was not used, we relied on the *trafillatura* (Barbaresi, 2020) library, which was desirable thanks to its good metadata extraction capacities and overall performances. A total of 3 different scrapping approaches were used: first trying to scrap directly from the original url, then trying to scrap from the internet archive, finally, in case the url was reachable but the data was not correctly extracted, we wrote an ad-hoc scrapper whose extraction rules were manually written for each problematic news source. Despite these efforts, about 7% of the test data was impossible to download. We report these numbers in Table 1. Most of the articles that had to go through the ad-hoc scrapper were from Chinese sources, for several of these it was possible to extract only the title, this created a difference in the features available with several missing values.

set	direct	arch.	ad-hoc	total	absent
train	8108	951	159	9431	274
test	7348	847	838	9715	682

Table 1: Number of additional downloaded pairs of urls by scrapping method, total and missing pairs

3.2 Preprocessing

The data underwent significant prepossessing. The title was cleaned of mentions to the news source. In order to do that, if the source name was extracted from the metadata and was present in the title it was removed, if a token was ending with an internet top level domain, the token was removed, if a pipe bar was present, the leftmost part of the bar was kept. If it was detected that a source always had the same title, indicating that the scrapper was not correctly extracting it, then the title was replaced by the first sentence of the description if available. If the description was not available, the first sentence of the article was used instead, in case the article had no text but had a description, the description was used for the text, and in case there was not description but a title, the title was used for the description. This procedure aimed at minimising the impact of missing values.

The text was preprocessed by removing likely source name, author name and date mention at the beginning of the text, using an heuristic rule-based procedure focusing on the presence of numbers, uppercase letter, short sentences and typical start of text markers, such a double dashes. Out of the remaining text, a snippet was extracted, taking the sentences spanned by the first 512 characters, with a maximum of 4 sentences. Unicode letters were normalized to canonical forms. The language annotation provided in the dataset was not trusted, and we instead relied on the one provided by the lan-

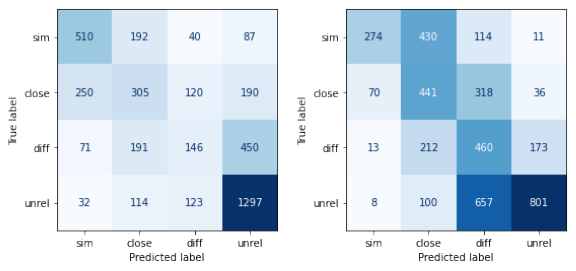


Figure 1: Confusion matrix for random forest classification (left) and rounded random forest regression (right)

guage detector of `fasttext` (Grave et al., 2018). Chinese texts were segmented using the `jieba` library in order to insert spaces between words so as to further deal with them in a language agnostic way.

4 Experimental Results

4.1 Experiments on train dataset

We settled on using random forests on the classification task as our contribution model after preliminary trial showed that random forests had better performances than neural networks and that the label distribution produced by random forests on the classification task was much more coherent with the ground truth than the one produced by random forests applied to the regression task. Given that we will tackle the problem as a classification problem, we will consider 4 classes labelled from 1 to 4, denoting the following relations in article pairs: similarity (sim), close similarity (close), different (diff) and unrelated (unrel). Unless specified otherwise, all the values computed over the train set are an average over 5-fold cross validation.

In Figure 1 we report the confusion matrix over the 4 classes using all the features with a random forest classifier (RF-C) and regressor whose output have been rounded to the closest integer (RF-R). RF-R has more errors than RF-C, but these are less significant as classes are mistaken mostly between related classes: for instance the classes sim and close are more often confused than with RF-C, but the classes sim and unrel are significantly less confused.

In Figure 3 we report the label distribution of the classifier and the regressor when trained and tested over the full training set and compare it with the ground truth. In this same setting, in Table 2 we report the Jensen-Shannon divergence of these label distributions, measuring the distance with the train data label distribution (Fuglede and Topsoe, 2004), we also report the micro F_1 and macro F_1 performance of RF-C and RF-R. Despite a seeming overfitting of RF-C, the distribution of labels produced by RF-C is clearly closer to the one of the ground truth than the one of RC-R. For all these different reasons we decided to tackle this task as a classification problem instead of a regression one.

Figure 2: Caption

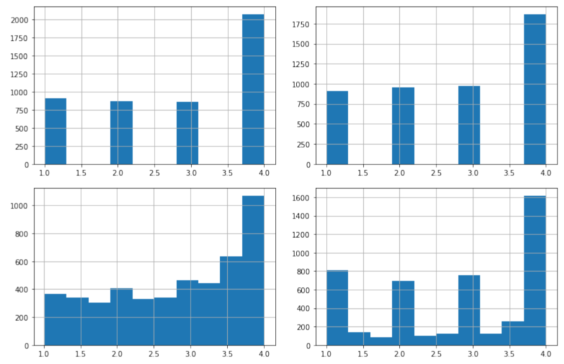


Figure 3: Label distribution for random forest classification (up. left), random forest regression (low. left), ground truth (low. right) and rounded ground truth (up. right)

measure	RF-C	RF-R
micro F1	93.6	80.2
macro F1	93.7	78.9
JS div.	0.0016	0.0098

Table 2: Performances on the train data, and distance of the label distribution with respect to the train dataset distribution

In Table 3 we report the performance of our classifier (RF-C) over different subsets of the training data, in function of the language pairs considered: all (MULTI), only same languages (SAME), only English (EN) and only different languages (CROSS). Both micro an macro F_1 are the highest for EN, nevertheless MULTI is the second best performing subset in terms of macro, with only one point less, while the micro difference of 6 points is important. The performance of SAME an MULTI are similar, this seems to indicate that specifically for English the performances are better. This could be due to the fact that English is the less flexional language of all the languages in the training set, as such, string matching is more likely to report the correct answer, both for exact and approximate matching. CROSS has the second highest micro score, but has the lowest macro score. This approach tends to rely more heavily on the embed-

subset	micro F_1	macro F_1
MULTI	55.2	47.5
SAME	54.1	47.2
EN	61.1	48.3
CROSS	58.0	40.1

Table 3: multilabel evaluation of RF-C: micro F_1 and macro F_1 performance of different subsets of language pairs of the train dataset

subset	F_1 (1 vs rest)	F_1 (1+2 vs rest)
MULTI	60.1	78.0
SAME	59.1	74.1
EN	64.5	75.1
CROSS	56.3	76.1

Table 4: binary evaluation of RF-C: the F_1 measure is reported for two binary classifier: class 1 (similar articles) vs rest and class 1+2 (similar and close articles) vs rest.

ding feature as lexical matches are unlikely except for named entities.

In Table 4 we report the measures for the same subsets, but applied when considering binary classifications problems. We consider two such classifiers: when considering class 1 (similar articles) versus the rest of the classes and when considering class 1 and 2 (similar and close articles) versus the rest of the classes. The performance of the binary classifiers are clearly superior to the performance of the multilabel classifier: on MULTI the former has a micro F_1 performance about 5 points better, and the later has a better performance by 22 points.

Study of the correlation matrix of the features with the ground truth, not reported in this paper for readability reasons, shows that all lexicographic-based features are the highest correlated features, with a correlation percent of about 50%. Among the several embedding-based features, only the distance between titles correlated highly with ground truth with 46%, second to it only similarity between title and description has a significant correlation of around 32%.

In Table 5 we report the performance of our classifier on the train dataset for different subsets of the features: all the features (ALL), only lexicographic-based ones (LEX), only embedding-based one (EMB), a combination of both (LEX+EMB) and date (DATE). ALL has clearly the best performance both in terms of macro and micro F_1, despite integrating the date feature, which on itself performs

features	micro F_1	macro F_1
ALL	55.2	47.5
LEX+EMB	53.3	44.8
LEX	49.6	40.4
EMB	51.3	42.6
DATE	37.5	22.6

Table 5: evaluation of different subsets measured as the micro and macro F_1 of RF-C in a multilabel setting

subset	support	micro F_1	macro F_1
all eval dataset	4902	44.1	40.5
all downloaded	4455	46.5	41.5
all with text	3946	47.6	42.4

Table 6: micro and macro F_1 performance of our approach on the evaluation dataset, for different subsets of articles pairs

quite poorly. However, we believe that the impact of this feature is heavily biased by the way both the train and test datasets have been constructed, as the organizer of the shared task have fetched news over a time period of several years, while about half the items are related, and about a quarter are similar. As a consequence, it happens that close dates are related to similar news more than it would appear in an uniform sample over the same time period. For that reason, we don't expect that this feature would generalise well, but we left it nevertheless as we expect the train and test distributions to be similar.

4.2 Experiments on test dataset

When evaluating on the test dataset, we use a classifier (RF-C) trained on the full training dataset. Because of the difficulties in downloading the test data, we report the performance of the classifier on different subsets of the downloaded data.

In Table 6 we report the micro and macro F_1 performances of our classifier, as well as the support, counted in number or article pairs. We consider three subsets: all evaluation data (including also article pairs whose articles were not downloaded, and for which a default score of 2.69 was used - which was the average predicted value of our classifier for the other articles), all pairs whose corresponding articles have been successfully downloaded, all pairs for which the text of the corresponding articles have been successfully downloaded. The best performance on the test data is on average 5 points lower both in term of micro an macro than on the

subset	micro F_1	macro F_1	F_1
MULTI	46.5	41.5	74.0
SAME	43.7	42.0	78.6
EN	58.4	42.7	79.0
CROSS	43.8	38.3	71.7

Table 7: performance on the evaluation dataset of the classifier, using micro and macro F_1 as well as F_1 for the corresponding binary classification problem

train dataset. Expectedly, the model evaluated on the full dataset performs the worst, but reasonably so with only 2 points down the performance on the subset of successfully downloaded articles while 9% of the data rely on the default value. Further restricting the evaluation on the articles that had a successfully downloaded text content only provides one additional point of performance.

In Table 7 we report the performance of the classifier on the test dataset for different subsets of language pairs, as already previously described. The measures are micro and macro F_1, as well as the F_1 of the class 1+2 when considering tackling the problem as binary classification. Interestingly, while the mutlilabel performances are lower than on the training set by about 9 points, the binary classification performance is only lower by 5 points in a cross language setting and actually higher by 4 points in a same language setting.

In Figure 5 we report the true positive rate by language pairs for pairs of languages having more than 10 data points, evaluated on the subset of the dataset for which articles were successfully downloaded. We consider a true positive in case the predicted and ground truth had exactly the same label, as such it is not possible to assess how close to the actual label the predicted values are. Therefore, this figure only allows to give an overall picture of the respective performances for pairs of languages. When considering pairs of articles in the same languages, French, Spanish and English performs the best. Among these, Chinese has the worst score, this could be related to the fact that a few Chinese sources representing a significant share of articles were impossible to be correctly downloaded (text and other metadata are absent). Surprisingly some pairs of different languages perform better than for these languages considered individually, this is the case for German and Chinese. Nevertheless, articles pairs in different languages tend to perform worse than pairs in the same language.

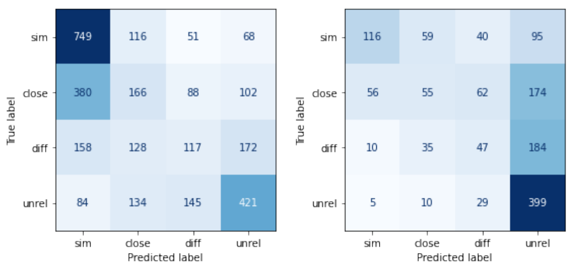


Figure 4: Confusion matrix of RF-C on the test set, for same-language pairs (left) and cross-language pairs (right)

In Figure 4 we report the confusion matrix of our classifier, by considering separately two cases: the confusion matrix for same-language pairs and for cross-language pairs. From this figure it is clear that our classifier tends to over-estimate the similarity of articles written in the same language, while underestimating the similarity of articles written in different languages.

5 Discussion

Despite the problem of the shared task being posed as a regression one, we have chosen to address it as a classification problem. This has two direct negative consequences on the performance of the model: firstly due the mandatory discretization step of the real-valued evaluation score provided in the training data and expected by the evaluation system; secondly due to the fact that the notion of related classes, such as "similar" and "close", are lost and penalized as much as if "similar" had been predicted as "unrelated". This is clearly shown by the fact that the multilabel classifier trained on 4 labels performs significantly worse than a binary classifier trained on two classes, which has actually acceptable performances. Despite being language agnostic our approach performs better in a same language setting than in a cross language one. This indicates either the high importance of the lexicographic as being a good predictor, or that multilingual embeddings perform significantly differently on different language pairs.

The lexicographic-based features perform surprisingly well, only two points under the performance of the embeddings-based features when evaluated on the full training dataset contain. A potential reason for that could be that named entities can differ only slightly between languages and that the soft lexicographic measure used is good at capturing these variations. The performance of our

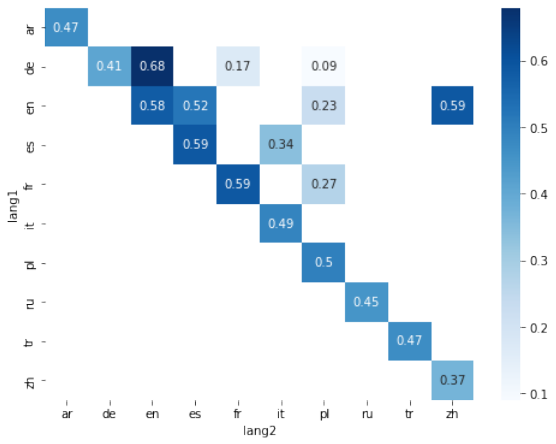


Figure 5: True Positive Rate by language pair of the multilabel classifier on the test dataset

approach is lower on the test dataset, such a drop in performance could be due to different reasons among others: the fuzzy lexicographic matching having lower performance in a cross language setting (much more prevalent in the test dataset than in the training); the difficulty to download data and correctly extract it, making the data incomplete; the heuristic reconstitution of missing description and text content could have introduced noise in some cases; maybe the fact that only tiny snippets of the full article were considered; or potentially a more fundamental limitation of the approach. Our approach could be improved by considering the actual language pairs as features, using more advanced features based on named entity extraction, and using exclusively data without missing values. However, we lacked time to investigates all of these configurations. Interestingly, of the embedding-based features, the title and the description are the most important features, and work better than more advanced ones, such as word mover distance applied to the sentences of the snippets.

Given these results, we can argue that the language agnostic approach we developed is an interesting solution for a coarse grain similarity evaluation, but not for a fine grained one. Given the fast computation time, a few minutes to compute all the features on a CPU machine without using multi-threading, our approach could be used as a preprocessing step before using more precise but also more time consuming approaches. Particularly, this approach is interesting when a news processing system has to process hundreds of thousands articles a day, preventing the use of costly solutions.

6 Conclusion

In this paper we presented the system we used to tackle the multilingual news clustering shared task at SemEval-22. Our approach is language-agnostic and inherently multilingual. It relies on a set of relatively simple lexicographic- and embedding-based features, and as such is able to process documents efficiently. We tackle the task as a classification problem rather than a regression problem. Our approach performs satisfyingly when evaluated over a simpler binary classification problem.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Adrien Barbaresi. 2020. htmldate: A Python package to extract publication dates from web pages. *Journal of Open Source Software*, 5(51):2439.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Bent Fuglede and Flemming Topsoe. 2004. Jensen-shannon divergence and hilbert space embedding. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, page 31. IEEE.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893*.

Jakub Piskorski, Nicolas Stefanovitch, Guillaume Jacquet, and Aldo Podavini. 2021. Exploring linguistically-lightweight keyword extraction techniques for indexing news articles in a multilingual set-up. In *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, pages 35–44.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. *arXiv preprint arXiv:2004.09813*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*.

ITNLP2022 at SemEval-2022 Task 8: Pre-trained Model with Data Augmentation and Voting for Multilingual News Similarity

Zhongan Chen[*], **Weiwei Chen**[*], **Yunlong Sun**[*], **Hongqing Xu**[*], **Shuzhe Zhou**[*], **Bohan Chen** [*], **Chengjie Sun**, **Yuanchao Liu**
Intelligence Technology and Natural Language Processing Lab,
School of Computer Science and Technology,
Harbin Institute of Technology
{zachen, wwchen, ylsun, hqxu, szzhou, bhchen}@insun.hit.edu.cn
{cjsun, lyc}@insun.hit.edu.cn

Abstract

This article introduces a system to solve the SemEval 2022 Task 8: Multilingual News Article Similarity. The task focuses on the consistency of events reported in two news articles. The system consists of a pre-trained model(e.g., INFOXLM and XLM-RoBERTa) to extract multilingual news features, following fully-connected networks to measure the similarity. In addition, data augmentation and Ten Folds Voting are used to enhance the model. Our final submitted model is an ensemble of three base models, with a Pearson value of 0.784 on the test dataset.

1 Introduction

The task (Chen et al., 2022) aims to design a system that can find the similarities of multi-language news articles. The task focuses on the consistency of actual events reported in two news articles, but not the subjective factors such as writing style or political factors. Geographical location, time, common entity, and common narrative were used to judge similarity. Consistency is measured by a float value between [1,4]. The lower the value is, the more likely the two news are reporting the same thing. The metric is the Pearson Coefficient of the predicted and ground truth values.

Challenge of this task: 1. Need to understand every aspect of a news event: what happened, where and when, who was involved, and why and how it happened 2. Make the writing style and common phrases clear because some unnecessary content can be misleading. 3. Some information about the event is hard to get, such as the time, location, and description of the event.

We use the pre-trained language model XLM-Roberta and INFOXLM to fine-tune. Specifically, we splice the available information of two news articles (exact method will be described in Chapter 3), input it into the pre-trained model, and then transmit the output vector to the fully connected layer of downstream tasks. For the original fine-tuning task, we try to use various techniques to make the program run faster and work better. The techniques include 1. Freezing the lower layers of the pre-trained model, means not updating their parameters during training. 2. Data Augmentation. Use translation software to translate the original news text to expand the data. 3. Divide the training set into 10 parts. the ten folds voting was adopted to make full use of the data set.

Our code is available on github[1].

2 Related Work

2.1 Background

The input of the task is the content of two news, and the output is the Overall label of them. Overall label is a float value between 1 and 4, which is used to measure whether two news report a same thing.The lower the Overall score, the more likely the content of the two news to be the same. The datasets including nearly 5,000 pairs of news with Overall label given. In addition to the Overall label, Geography, Entities, Time, Narrative, Style, and Tone label are also noted in the datasets. The news is given in the form of links and contains seven languages(en, de, ar, es, fr, pl, tr), while the test set contains the other three languages(it, ru, zh). It should be noted that the dataset contains news pairs in different languages.

2.2 Pre-trained language model

Pre-trained language models such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) start to make a difference in the way of word representations rather than static word embedding methods, and Word2Vec (Mikolov et al., 2013) and FastText

[1]*These authors contributed equally.

[1]https://github.com/SemevalITNLP/Semeval8NewsCorrelation

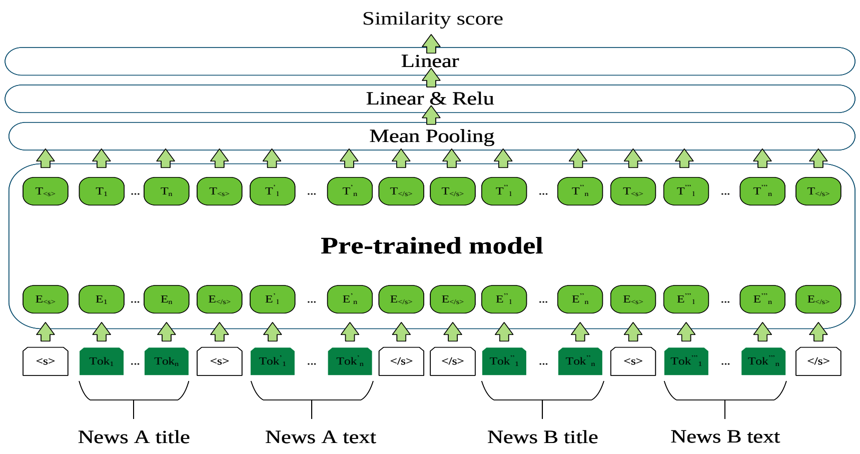


Figure 1: The architecture of the system.

(Joulin et al., 2016) are two examples. In particular, the XLM-RoBERTa (Conneau et al., 2019) model is a newly released large-scale cross-lingual language model based on RoBERTa and trained on 2.5TB filtered CommonCrawl data in 100 languages. Unlike other XLM models (Lample and Conneau, 2019), XLM-RoBERTa does not require language tokens to understand which language is used and can recognize the correct language from the input id.

INFOXLM (Chi et al., 2020) is a cross-lingual pre-trained model based on the XLM-RoBERTa structure, using monolingual and parallel corpora to train the model. Specifically, in addition to the masked language modeling(MLM) and the translating language modeling tasks(TLM), INFOXLM is jointly pre-trained with a newly introduced cross-lingual contrastive learning task. Through comparison, the cross-lingual uses bilingual pairs as the two views of the same meaning, making their encoded representations more similar than the negative examples. It uses the [CLS] tokens from the BERT encoder as sentence representations with linear projection heads. The momentum encoder is used to encode the query, while the online encoder is updated with InfoNCE (Van den Oord et al., 2018) loss.

2.3 Sentence similarity

There are usually two methods for comparing the similarity between two sentences. Cross-encoders perform full-attention over the input pair, and Bi-encoders map each input independently to a dense vector space.(Thakur et al., 2020)

3 System Description

3.1 Data processing

We use crawlers and the newspaper3 [1] library to download and parse news web pages to obtain various information about news, including title, text, pictures, keywords and abstracts. For the news data in the training and test set, some missing titles and body texts. At the same time, some web pages have errors during the crawling process, or the crawler crawls wrong news information. We revisit the news link and the alternate link to modify the news data. In addition, there are still problems such as advertising webpages and link failures, which are omitted in the training process.

3.2 Model

We designed cross-encoder model to solve the task. The structure of the cross-encoder model shows as Fig 1. It contains the pre-trained language model, pooling layer, and downstream layers, consisting of two fully connected layers and a relu activation function, to learn the Overall value of each news pair.

The inputs of cross-encoder model are composed of contents of two news which are named news A and news B. For each news, we use title, text and keywords. We use the symbol <s> to separate different parts of the news, the symbol </s> as the separator for two news articles, and add <s> and </s> at the beginning of each news pair. The input form finally can be illustrated as: <s> news A </s></s> news B </s>.

[1] https://github.com/codelucas/newspaper

For the pooling methods, we compared two methods of learning the representation of the whole news pair: 1. use the output of the model Pooler, which is similar to the token's representation of the [CLS] of BERT; 2.mean pooling: use the average vector of the whole hidden state. Experiments have shown that the latter works better.

The model eventually outputs a value between 1 and 4, which calculate MSE loss (Mean Squared Error Loss) with the standard Overall label of the data.

3.3 Methods

- **Ten-Fold Voting** Ten-Fold Voting method shuffles and divides the training set into ten parts equally, and separately chooses one of them as a validation set, while the remaining nine part are used as the training set. As a result, we end up with 10 models. These ten models use averaging method to vote, which increases model generalization. For example, we use 10 models to obtain 10 Overall predictions for a given news pair, and the average of these 10 results is taken as the final prediction for that news pair.

- **Data Augmentation** Due to the addition of three new languages (zh, ru, it) in the test set, we added translation corpus to the training data so that the model can better deal with the new languages. Specifically, we use Baidu translation API [1] to obtain the translated text, and the news in each language is translated into the other 9 languages. Different epochs are trained alternately with the original and translated data. Each news is translated into one of the nine other languages with the same probability at the epoch using translated data.

- **Frozen Layers** Generally, lower layers of a language model encode more local syntax while higher layers capture more complex semantics (Tenney et al., 2019). Therefore, during training, we freeze the parameters of the embedding layer and some lower layers of the pre-trained model. The parameters of frozen layers cannot participate in back propagation thus keeping the original parameters. Therefore, we aim to choose the most memory-efficient hyperparameters.

- **Multi-task Learning** We noticed that the training data not only has the Overall label but also has other labels. Considering that Narrative and Entity are consistent with the target task, we let the model fit the Overall label and the Narrative label and Entity label. Specifically, after the pre-trained model, we separately use a dense layer to obtain different predicted values for Entity, Narrative, and Overall and calculate the MSE loss of the three labels: $loss_{entity}$, $loss_{narrative}$ and $loss_{Overall}$. Then the multi-task loss of the model, $loss_{multi-task}$, take the weighted sum of the three.

- **Auxiliary Loss** We add a loss function to help the model distinguish those news pairs. In addition to calculating the similarity of each news pair, we also calculate the MSE loss between the predicted Overall labels of the two news pairs and their true Overall labels, which is as following.

$$loss_{AL} = mse(\hat{y_1} - \hat{y_2}, y_1 - y_2) \qquad (1)$$

  where y_1 is the standard Overall value of news pair 1 and $\hat{y_1}$ is the predicted Overall value of news pair 1, so as y_2 and $\hat{y_2}$. Two news pairs are randomly selected. In practice, we randomly draw several news pairs from a batch and combine them. The training data was shuffled before the start of each training epoch so that the data within the batch are not precisely the same in different epochs.

4 Experiment

All experiments were run on two GPUs: NVIDIA GeForce RTX 3080 Ti and NVIDIA GeForce RTX 3090. For optimizer selection, we use Adamw optimizer with weight decay taken as 0.01 and set 1e-5 as learning rate for the pre-trained layer and 1e-4 for downstream layers with the batchsize 8.If not specified, we use **INFOXLM-large** as the pre-trained model from Hugging Face[2].The weight of the $loss_{entity}$, $loss_{narrative}$ and $loss_{Overall}$ in the $loss_{multitask}$ are set to 0.3, 0.7 and 1.0 respectively.The weight of the $loss_{AL}$ is set to 0.1.

4.1 Information Selection

This section presents experiments to explore which news information should be used. From Table 1

[1] http://api.fanyi.baidu.com/

[2] https://huggingface.co/models

,it can be seen that the result of using only title and only text is not as good as using title and text at the same time. The model shows better performance when facing more text types, which may be because different texts contain different information. The title focuses on summarizing the news and represents the article's central idea; The text focuses on describing the event and conveying more detailed information. The performance of using title, keywords and text is not improved. Most of the keywords information is likely contained in the first two.

Model	$\mathcal{D}_{val}$	$\mathcal{D}_{test}$
Title60	0.798	0.684
Text100	0.813	0.702
Title60+Text100	0.852	0.759
Title60+Text100+Keywords50	0.855	0.760
Title60+Text150	0.859	**0.772**
Title60+Text200	**0.859**	0.769

Table 1: The impact of using different information and different content lengths on the model. Pearson coefficients in the table are the maximum values for the validation set, notated $\mathcal{D}_{val}$ in 15 epochs, while $\mathcal{D}_{test}$ is the corresponding pearson coefficients in test set at this time.The number after the title and text in the table indicates how many text words are used. For example, Title60 indicates using the first 60 words of the title; The ratio of validation set to training set is 1:9.

The text length of each news article is related to the final prediction of the system. The table shows the Overall scores of different text lengths. Due to the input length limitations, our max length of the title is 60, while the max length of the text is between 100 and 200. The model works better with longer text lengths in the validation set, probably because longer texts contain richer information, while it achieves the best result in the test set when text length is 150. The reason may be that the behavior of the test set is inconsistent with the validation set.

4.2 Frozen layers

We tested the performance of unfrozen, frozen embedding layers and partial transformer layers respectively. The results are shown in Table 2. As we can see, when freezing embedding layers and the layers from 0 to 11, the result is similar to not freezing all layers, but it saves memory and reduces training time. So we take these kinds of parameters in the following experiments.

Frozen layers	$\mathcal{D}_{test}$
no freezing	0.760
freeze embedding + layers 0$\sim$5	0.759
freeze embedding + layers 0$\sim$11	0.759
freeze embedding + layers 0$\sim$14	0.750

Table 2: Results of freezing different pre-trained model layers on the test set.

4.3 Strategies

We also conducted additional experiments, including data augmentation through translation, a multi-task learning approach, and Auxiliary Loss mentioned before. These methods are effective in the validation set, but some did not improve much in the final test set.

Table 3 shows the comparison of different methods used in the model. The parameters used by the base model are INFOXLM$_{\text{large}}$, Title60 and Text100. The split ratio between the training set and the validation set is 9:1. According to the table, we have the following conclusions:

In the validation set, when the model cumulatively uses Multi-task Learning, Auxiliary Loss and Data Augmentation, the performance is continuously improved. When the three methods are used together, the model performance is best to reach 0.8627 in the validation set. Multi-task Learning and Auxiliary Loss increase the learning and representation ability of the model by modifying the task and loss. Data Augmentation improves the generalization of the model by introducing the translation corpus.

In the test set, when the model uses Multi-task Learning and Auxiliary Loss, the performance is not much different from that of the base model. This may be that Multi-task Learning and Auxiliary Loss can improve the representation ability of the model in the validation set, but they are lack of generalization and generally perform in the face of a large amount of data and new data. However, when the model uses Data Augmentation, the performance reaches 0.7667 in the test set, which is greatly improved compared with the base model and the model using Multi-task Learning and Auxiliary Loss. This may be because the translation corpus introduces languages (Chinese, Russian and Italian) that the model has not encountered before. The newly introduced language enhances the model's generalization ability in the test set and enables the model to understand the news in the test set better.

Model	$\mathcal{D}_{val}$	$\mathcal{D}_{test}$
Base	0.852	0.759
Base + MT	0.854	0.758
Base + MT + AL	0.859	0.759
Base + MT + AL + DA	**0.862**	0.766
Base + MT + AL + DA + TFV	/	**0.781**

Table 3: Results of system under different methods. Base: $\text{INFOXLM}_{\text{large}}$ +Title60+Text100; MT:Multi-task Learning; AL:Auxiliary Loss; DA:Data Augmentation; TFV:Ten-fold voting

Finally, we can see that when the model uses Ten-Fold voting, the performance is improved from 0.7667 to 0.7810. There may be two reasons for this. On the one hand, the Ten-Fold voting essentially uses all the training data, and the expansion of the amount of data may increase the performance of the model. On the other hand, the Ten-Fold voting integrates the model results under ten different training data sets, which greatly improves the robustness of the results and strengthens its generalization ability.

4.4 Ensemble

The ensemble technique is a widely used strategy. Ensemble methods work by aggregating the predictions of multiple single models. The strategy we use in the competition is simple averaging. The final prediction value is obtained by averaging the prediction results of different models, which will improve the robustness of the prediction results. The results are shown in Table 4.

Model	$\mathcal{D}_{test}$
INFOXLM	0.776
INFOXLM + DA	0.781
XLM-RoBERTa + DA	0.779
Ensemble	**0.784**

Table 4: Results of base models and ensemble model.

All three base models use Title60, Text100, Multi-Task Learning and Auxiliary Loss. The latter two models additionally use the Data Augmentation strategy. The result of the ensemble model achieves the best performance in our competition.

5 Conclusion

In this paper, we summarize our work in Multilingual News Article Similarity. We utilize INFOXLM and XLM-RoBERTa pre-trained models to handle multilingual news. In addition, many methods are used such as Data Augmentation and Ten-Fold Voting. Our final submitted model is an ensemble of three base models, and we achieve Pearson value of 0.784 on the test dataset.

Acknowledgements

Thank the organizer for holding a great competition and providing the large multilingual news article similarity dataset. This research was supported in part by the National Natural Science Foundation of China under contract 62176074.

References

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2020. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. *arXiv preprint arXiv:2007.07834*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *north american chapter of the association for computational linguistics*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2020. Augmented sbert: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. *arXiv preprint arXiv:2010.08240*.

Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807.

LSX_team5 at SemEval-2022 Task 8: Multilingual News Article Similarity Assessment based on Word- and Sentence Mover's Distance

Stefan Heil and **Karina Kopp**
{stefan.heil,karina.sultangaleeva}@stud-mail.uni-wuerzburg.de
Konstantin Kobs and **Albin Zehe** and **Andreas Hotho**
{kobs,zehe,hotho}@informatik.uni-wuerzburg.de
University of Würzburg

Abstract

This paper introduces our submission for the SemEval 2022 Task 8: Multilingual News Article Similarity. The task of the competition consisted of the development of a model, capable of determining the similarity between pairs of multilingual news articles. To address this challenge, we evaluated the *Word Mover's Distance* in conjunction with word embeddings from *ConceptNet Numberbatch* and term frequencies of *WorldLex*, as well the *Sentence Mover's Distance* based on sentence embeddings generated by pretrained transformer models of *Sentence-BERT*. To facilitate the comparison of multilingual articles with *Sentence-BERT* models, we deployed a Neural Machine Translation system. All our models achieve stable results in multilingual similarity estimation without learning parameters.

1 Introduction

The assessment of similarity between documents is a central challenge in the context of information retrieval. Especially the evaluation of similarities between news articles across different languages opens up opportunities for numerous downstream tasks, such as the analysis of regional differences in news coverage of topics or sentiments towards events and news.

Task 8 of the *SemEval* challenge 2022 (Chen et al., 2022) posed the problem of the assessment of similarity of news articles across a variety of languages and provided a dataset of news article pairs in seven languages, as well as a number of bilingual pairs. Apart from an overall similarity score to be estimated in the challenge, a number of scores for similarity in different categories, such as *narrative* or *entities* were provided.

For our submission for this task, we evaluated the performance of the *Word Mover's Distance* (WMD) and *Sentence Mover's Distance* (SMD) in the context of similarity assessment of multilingual news articles. We participated in all given languages, comparing both document pairs in the same language, as well as pairs in different languages. Our code is available at `https://github.com/StefanJMU/SemEval2022_Task_8`.

We considered WMD and SMD due to their conceptual simplicity and capability to integrate well with resources such as pretrained word embeddings or sentence embeddings, produced by state-of-the-art language models. Additionally, they offer the appeal of being themselves parameter-free, and hence independent of labelled training data.

The rest of the paper is organized as follows: We introduce the approaches deployed in the challenge submission, as well as the used resources, supplementing the *WMD* and *SMD*. Subsequently, we present the results achieved and conclude with a discussion of the results.

2 System Overview

Figure 1 shows a schematic overview of the methods we investigated for our submission. We evaluated two different methods for this task, namely the *Word Mover's Distance* (WMD) (Kusner et al., 2015) and the *Sentence Mover's distance* (SMD) (Clark et al., 2019). Both approaches take two news articles and calculate a similarity score from the representation of both texts as either *Bag of Words* or *Bag of Sentences*, respectively. Both approaches have been found to constitute a metric exhibiting a pronounced correlation with the human-assessed similarity scores of the text pairs (Kusner et al., 2015). We create the required word and sentence representations using *word embeddings* and *sentence embeddings* generated from state-of-the-art language models.

For the *WMD* approach, we deployed a preprocessing pipeline, involving the tokenization of both news articles, as well as the removal of stopwords and punctuations. Subsequently, the preprocessed texts were transformed into a *Bag of Words* using

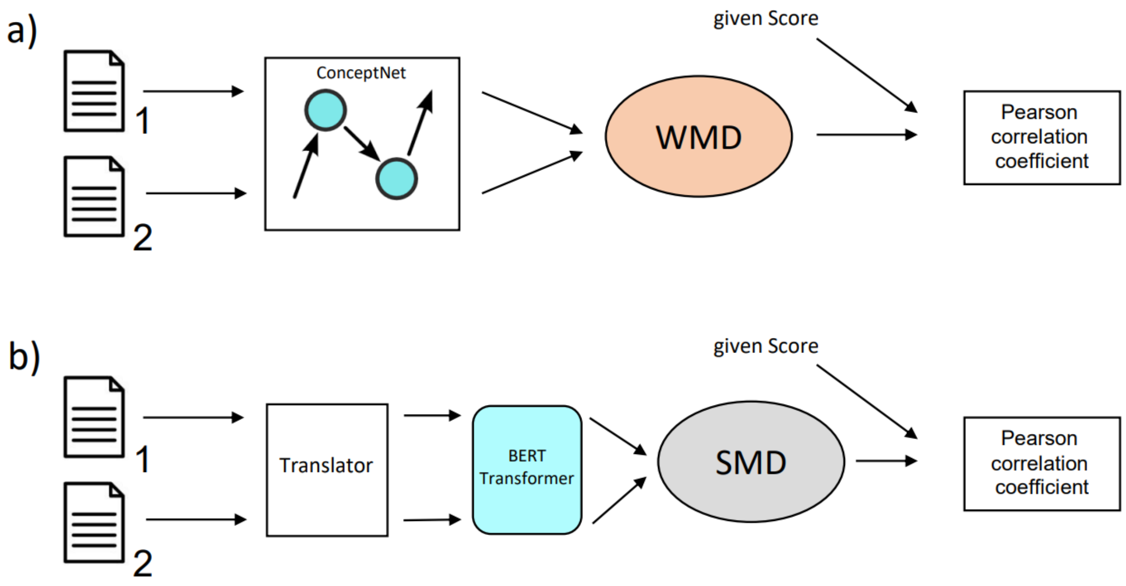


Figure 1: Schematic overview of our experiments in order to determine the best model for submission. Given two news articles, we **a)** use pretrained multilingual ConceptNet Numberbatch word embeddings for both articles and compute the *Word Mover's Distance* (Kusner et al., 2015) between both non-translated texts; also **b)**, we obtain sentence embeddings from a Sentence-BERT model (Reimers and Gurevych, 2019) by first translating both articles to English and then computing the *Sentence Mover's Distance* (Clark et al., 2019) on the translated documents.

the vector representations provided by *ConceptNet Numberbatch*. These vector representations, together with the dot-product similarity, constituted the metric at the core of the linear optimization problem forming the *Word Mover's Distance* between two texts. The *WMD* also allows for a different weighting of the words of texts in the evaluation of similarity. These weights were chosen according to the respective *TF-IDF*, calculated with the help of *WorldLex* (Gimenes and New, 2015).

The *SMD* approach required the decomposition of texts into sentences, which were subsequently encoded with a transformer language model, resulting in a *Bag of Sentences* representation of texts. By interpreting these sentence embeddings as words, a similarity score is readily computable using the *WMD* again.

The following subsections introduce both metrics in more detail, as well as the word embeddings and models involved, which were deployed to extend the application of the metrics to similarity assessment of texts across of different languages.

2.1 ConceptNet Numberbatch

For the calculation of *WMD* for multilingual articles, we deployed ConceptNet Numberbatch word embeddings (Speer and Lowry-Duda, 2017). ConceptNet Numberbatch are embeddings based on the knowledge graph ConceptNet (Speer et al., 2017). Due to its multilinguality (and support of all languages used in the challenge), we selected these embeddings to facilitate the calculation of the *WMD*.

2.2 WorldLex

Evaluating semantic similarity by matching words across texts can be obstructed by the presence of stop words or words, which can occur in many semantic contexts and are therefore no compelling indicators for semantic similarity. Apart from the removal of stopwords, we deployed a weighting of words according to the occurrence frequency in Twitter, blogs and newspapers gathered by Gimenes and New (2015), which are curated in the WorldLex database and were available for all languages used in the SemEval task. To create the WorldLex database, the authors converted all collected documents to lowercase. After that, the frequencies of all of the different words were calculated and lists of words were extracted utilizing spellcheckers to remove words with orthographic and typographic errors, as well as foreign words.

2.3 Word Mover's Distance

In 2015, the *Word Mover's Distance* has been proposed by Kusner et al. (2015) and constitutes a conceptually simple mean of quantifying the distance (and inversely correlated, the similarity) between texts, by optimizing a linear program for a cost minimal mapping between words of two texts with respect to a similarity measure between words. A text with n distinct words is considered as a vector $t \in \mathbb{R}^n$, with $||t||_1 = 1$ and t_i indicating the weight (from the WorldLex Database described in Section 2.2) of the ith distinct word of the text. The rationale of the *WMD* is, that for each word in a text, the weight has to be accounted for by words in the other text, while no word can account for more than its own weight. Considering two vector representations t^1 and t^2 of texts T_1 and T_2 of length n_1 and n_2 respectively, the distance between both texts Ω can be mathematically formulated as

$$\Omega = \min_M \sum_{i,j} M_{ij} c(i,j) \tag{1}$$

$$s.t. \sum_{j=1}^{n_2} M_{ij} = t_i^1 \, , 1 \leq i \leq n_1 \tag{2}$$

$$\sum_{j=1}^{n_1} M_{ji} = t_i^2 \, , 1 \leq i \leq n_2 \, , \tag{3}$$

where $c(i,j)$ is the distance between the ith distinct word of T_1 and the jth distinct word of T_2, and M is the accounting matrix, where M_{ij} indicates the amount of weight the ith word of T_1 provides for the accounting of the weight of the jth word of T_2. The resulting Ω can subsequently be used as measure correlated with smililarty or distance of text pairs.

Many options for choosing the weights of words, such as the document frequency of the word, are possible. We deployed a TF-IDF weighting of the words, where the inverse document frequency was derived from the *WorldLex* database introduced in the previous section. The rationale for introducing also the inverse document frequency, instead of only a term frequency weighting as suggested by Kusner et al. (2015), was to have a stronger emphasis of words, which are potentially more semantically meaningful and are therefore more suited for comparing themes and content of two texts.

For the quantification of distances between words, Kusner et al. (2015) propose distance measures such as cosine similarity on dense word embeddings, and use themselves word embeddings generated by *Word2Vec* (Mikolov et al., 2013). To facilitate the comparison of multilingual text pairs, we used multilingual *ConceptNet Numberbatch* embeddings (Speer and Lowry-Duda, 2017).

2.4 Sentence Mover's Distance

For our second line of experiments, we used the *Sentence Mover's Distance*, which was introduced by Clark et al. (2019), who adapted the concept of the *Word Mover's Distance* to calculate the distance of texts based on sentences instead of words, in order to address the typical shortcomings of approaches considering only Bag of Words without the incorporation of any compositional information contained in the word order. For the required representations of sentences, Clark et al. (2019) suggest averaging the word embeddings of the words in a sentence and a subsequent weighting of the sentences within the *Bag of Sentences* according to the number of constituent words.

Apart from the operating mode of the *SMD* proposed by Clark et al. (2019), the Sentence Mover's Distance allows also the incorporation of more rich representations of sentences, such as sentence embeddings produced by complex transformer-based models. Trained implementations of such models are readily available, such as the *Sentence-BERT* proposed by Reimers and Gurevych (2019). For the evaluation of the *Sentence Mover's Distance*, we deployed the pretrained *all-MiniLM-L12-v2* provided on `sbert.net`, which embeds an English sentence into a 384-dimensional embedding vector and supports the dot-product as similarity measure.

2.5 Neural Machine Translation

Since the used pretrained transformer can only embed English sentences, we propose to use English as an intermediary language, into which the multilingual news articles were automatically translated using a Neural Machine Translation system.

For the translation of the articles, we used a model submitted to WMT's 2021 News translation task (Chau et al., 2021). This translator has ranked first in the most language directions the team participated in, introducing the first multilingual model with stronger translation performance than bilingual ones. Using the *any-to-English* model and *DeepL*[1] (for the languages not supported by the first translation model), we translated all news articles

[1] `https://www.deepl.com/translator`

Dataset	**Language**									
	en	**pl**	**es**	**de**	**zh**	**ru**	**tr**	**fr**	**it**	**ar**
Training	4048	663	1114	2198	-	-	903	144	-	541
Test	1487	555	1291	1536	1728	574	548	345	1127	589

Table 1: Number of used articles for each language in training and test data.

and used them to calculate BERT transformer embeddings. These in turn were used for the *SMD* calculation. We also evaluated possible performance differences for the *WMD*, if texts are translated beforehand, instead of relying on the power of the multilingual word embeddings of *ConceptNet Numberbatch*.

3 Experimental Setup

For preprocessing routines, the Python *nltk*-package was utilized (version 3.6.5). The embeddings have been conducted using *ConceptNet Numberbatch* version 19.08. The *TF-IDF* weighting used *WorldLex*. *WorldLex* is no longer under active development.[2] The transformer model was *all-MiniLM-L12-v2* provided on `sbert.net`.

We employed three experiments to find the best model used for submitting to the task's evaluation:

1. Word Mover's Distance with multilingual ConceptNet Numberbatch embeddings

2. Word Mover's Distance with translated news articles before embedding them using ConceptNet Numberbatch embeddings

3. Sentence Mover's Distance using translated news articles embedded with an English Sentence-BERT model

For the evaluation, the Pearson Correlation Coefficient (Pearson, 1895) was used. The target similarity scores range from one (very similar) to four (not similar), so these scores really correspond numerically to distances. We can thus use the computed distances directly as predictions. Since the calculation of the Pearson Correlation Coefficient normalizes all predicted and actual similarity scores using the mean and standard deviation, we do not need to scale the predicted scores to the range of the labels.

Due to the fact that our employed methods do not need any training labels, we can directly evaluate the results on the training data provided by the task organizers. The model found to perform best was then used for the task submission. The number of articles used for each language is listed in Table 1.

[2]The deployed data can be retrieved from `http://www.lexique.org/?page_id=250`.

4 Results

Table 2 shows the Pearson Correlation Coefficient of the articles for the provided training data consisting of eight language pairs. Comparing the results of *WMD* applied to original and translated documents shows, that the estimation of text similarity with the proposed model benefits from a translation in a common language, instead of solely relying on a multilingual word embedding (all documents were translated into English). It can also be seen from the table, that for all language pairs except for *ar-ar* and *de-de*, *SMD* shows the best performance. As in the analysis of Kusner et al. (2015), we also find, that using the richer representations of complete sentences, instead of words, is generally beneficial. For the two remaining language pairs, *WMD*, applied on translated articles, achieves higher scores than the other metrics.

Given these results, we used the *Sentence Mover's Distance* with translated input texts and the pretrained language model for sentence embeddings as the model for our task submission. We achieved the 23nd place with an overall Pearson Correlation Coefficient of 0.57. Table 2 also shows the final scores for all 18 language pairs present in the test set. For the most language pairs, the results on the test data are very similar or even slightly better compared to the scores on the training set. We expected this behavior, since the used approach does not use any training labels for optimization and is hence not prone to overfitting. The rigidness of the optimization and straightforwardness of the similarity notion however, allows also for higher performance fluctuations observed for instance in the two language pairs (**en-en** and **de-de**), which perform noticeably worse for the test set than for the training data, while **pl-pl** shows much better performance on the test data.

Also, language pairs including German news articles are typically performing worse than average

	Training Language Pairs								Test
Model	**en-en**	**de-en**	**es-es**	**fr-fr**	**tr-tr**	**ar-ar**	**pl-pl**	**de-de**	**ru-ru**
WMD	0.77	0.60	0.67	0.68	0.72	0.55	0.36	0.54	—
WMD translated	0.77	0.66	0.68	0.69	0.76	**0.60**	0.42	**0.63**	—
SMD	**0.78**	**0.68**	**0.71**	**0.73**	**0.77**	0.59	**0.45**	0.59	—
Test Results SMD	0.68	0.69	0.71	0.70	0.71	0.65	0.59	0.31	0.56

	Testing Language Pairs								
	zh-zh	**es-en**	**it-it**	**pl-en**	**zh-en**	**es-it**	**de-fr**	**de-pl**	**fr-pl**
Test Results SMD	0.56	0.77	0.69	0.69	0.66	0.65	0.40	0.47	0.75

Table 2: Pearson Correlation Score for all language pairs in the provided training and test data of the task evaluation. Since the *SMD* worked best on the training data overall, we used this method for the final submission.

and thus decrease the overall test score in the challenge. While looking for possible reasons for such poor results, we found some articles in training and test data, which could not be extracted correctly. For instance, some German articles from the test data are identical and contain only information about the issue of opening the web pages due to privacy regulations[3]. Another reason for bad performance in similarity estimation of pairs containing German article (or comparison of two German articles) could be the incompletely loaded content of some German pages, since some articles only contain the beginning of the actual text[4], which is indicated by the spontaneous termination of the article content with the following sentence "read the full article..."[5].
In order to eliminate such externally imposed assessment impediments, the data scraping system[6] would need to be overhauled.

5 Discussion and Conclusion

For our submission, we evaluated multiple approaches, that operate on the word or sentence level and calculate a distance between two texts using a linear program optimized on pretrained word or sentence embeddings. To be able to apply English-only models for the representation of sentences, we used a Neural Machine Translation system that, in our experiments, improved the performance of multilingual word embeddings. The proposed model shows stable performance in similarity estimation between mono- and multilingual document pairs. The usage of state-of-the art pretrained word and sentence embeddings led to a fast system with low computational cost, allowing implementation without use of graphics processing units. The use of an extensively pretrained *Sentence-BERT* transformer for sentence embeddings of documents, that were translated into English, confirmed, that the proposed model is well suited for the similarity comparison of multilingual articles without optimizing any parameters in the model.

[3]we found two groups of articles, each with the same content:
1.Group: 1586615494, 1490686353,1520406037,1524031333, 1525352422
2.Group: 1572312750, 1576180076,1611845398,1612866403 ,1617051090,1619154724,1627621567, 1551767123,1562891463, etc.

[4]articles with IDs 1488265289,1493242324, 1505316713,1516114270,1517039073, 1519376267,1531637961,1549821395, etc.

[5]translation of original German sentence: "Den vollständigen Inhalt lesen..."

[6]https://github.com/euagendas/semeval_8_2022_ia_downloader

Acknowledgements

We would like to thank all organizers of the SemEval-2022 Challenge. Our special thanks go to our supervisors from the Data Science Chair and Prof. Dr. Andreas Hotho for the given opportunity, computing resources and fascinating learning experience in natural language processing.

References

Tran Chau, Bhosale Shruti, Cross James, Koehn Philipp, Edunov Sergey, and Fan Angela. 2021. Facebook ai´s wmt21 news translation task submission. arXiv:1503.06733.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity.

Elizabeth Clark, Asli Celikyilmaz, and Noah A. Smith. 2019. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. *Proceedings of the*

57th Annual Meeting of the Association for Computational Linguistics, pages 2748–2760.

Manuel Gimenes and Boris New. 2015. Worldlex : Twitter and blog word frequencies for 66 languages. *Behaviour Research methods*, 48:963–972.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. *Proceedings of Machine Learning Research*, 37:957–966.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Karl Pearson. 1895. Vii. note on regression and inheritance in the case of two parents. *Proceedings of the royal society of London*, 58(347-352):240–242.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3973–3983.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Robyn Speer and Joanna Lowry-Duda. 2017. ConceptNet at SemEval-2017 task 2: Extending word embeddings with multilingual relational knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 85–89, Vancouver, Canada. Association for Computational Linguistics.

Team dina at SemEval-2022 Task 8: Pre-trained Language Models as Baselines for Semantic Similarity

Dina Pisarevskaya, Arkaitz Zubiaga
Queen Mary University of London, UK
dinabpr@gmail.com, a.zubiaga@qmul.ac.uk

Abstract

This paper describes the participation of the team “dina” in the Multilingual News Similarity task at SemEval 2022. To build our system for the task, we experimented with several multilingual language models which were originally pre-trained for semantic similarity but were not further fine-tuned. We use these models in combination with state-of-the-art packages for machine translation and named entity recognition with the expectation of providing valuable input to the model. Our work assesses the applicability of such “pure” models to solve the multilingual semantic similarity task in the case of news articles. Our best model achieved a score of 0.511, but shows that there is room for improvement.

1 Introduction

The Multilingual News Article Similarity Task[1] (SemEval 2022 Task 8) is designed as a shared task to encourage participants to build systems that check if a monolingual or cross-lingual pair of news articles belong to the same story (Chen et al., 2022). The task consists in providing a similarity score from 1 to 4 for a pair of news articles.

In this paper, we describe the participation of the “dina” team in the shared task. In our participation, we took an exploratory approach focusing on language features, while trying several multilingual language models as baselines. These language models had been pre-trained for semantic similarity and are “pure” (without any fine-tuning conducted for the task), which we enhance to use jointly with state-of-the-art packages for machine translation and named entity recognition.

Our best-performing submission to the task is based on a “pure” paraphrase-xlm-r-multilingual-v1 model combined with the presence of overlapping named entities and overlapping dates, which achieved a Pearson’s correlation score of 0.511 in the final evaluation.

2 Related Work

Multiple prior SemEval semantic similarity tasks (2012-2017) focused on estimating the degree of semantic equivalence between two text fragments. In 2014 and 2015, two subtasks were proposed for semantic textual similarity in English and Spanish (Agirre et al., 2015). It was found that aligning words between sentences worked best for English, using features such as WordNet, word embeddings, or paraphrase databases. In 2016, a new cross-lingual subtask was added for English and Spanish. In 2017, participants were instructed to predict the degree of semantic similarity (namely, a continuous valued similarity score on a scale from 0 to 5) between monolingual and cross-lingual sentences in Arabic, English and Spanish (Cer et al., 2017). State-of-the-art deep learning models and feature engineered systems were implemented (Tian et al., 2017), and machine translation was widely used for cross-lingual and non-English setups, in order to convert two sentences into the same language. Based on the corpus of English tasks data (2012-2017), the STS benchmark was presented for training and evaluation[2]. Its extended version is often used to evaluate the performance of multilingual pre-trained models[3].

Currently, state-of-the-art methods are based on pre-train transformer language models, which are fine-tuned for downstream tasks. Multilingual pre-trained language models for 50+ languages are freely available[4], such as distiluse-base-multilingual-cased models,

[1]https://competitions.codalab.org/competitions/33835

[2]http://ixa2.si.ehu.eus/stswiki/index.php/STSbenchmark

[3]https://www.sbert.net/examples/training/multilingual/README.html

[4]https://www.sbert.net/docs/pretrained_models.html#multi-lingual-models

paraphrase-multilingual-MiniLM-L12-v2, paraphrase-multilingual-mpnet-base-v2, LaBSE (Feng et al., 2020), and other various models, such as paraphrase-xlm-r-multilingual-v1[5] (Reimers and Gurevych, 2019). All these models are applicable to the sentence similarity task.

Event clustering can be considered as a task that is close to the task of semantic similarity between news articles. Miranda et al. (2018) used similarity metrics and ranking for clustering documents into monolingual and cross-lingual story clusters. Linger and Hajaiej (2020) used multilingual DistilBERT and Sentence-BERT for multilingual document representation. In general, BERT-like models with different averaging and pooling are widely used for document representation in the clustering task too.

In the system proposed in our paper, we build on this line of research of leveraging large multilingual language models, which we further experiment with by incorporating machine translation and named entity recognition components.

3 Task and Data

The aim of the task was to check, at the document level, if a pair of news articles, which can be written in the same or different languages, provide similar information. The training set provided by the organisers includes 4,964 monolingual and cross-lingual pairs of news articles: 1800 English-English pairs, 857 German-German pairs, 577 German-English pairs, 570 Spanish-Spanish pairs, 465 Turkish-Turkish pairs, 349 Polish-Polish pairs, 274 Arabic-Arabic pairs, and 72 French-French pairs. For test data, the organisers added three new languages: Italian, Russian, Chinese.

For each entry in the dataset, different similarity scores are given for "Geography", "Entities", "Time", "Narrative", "Style" and "Tone", in addition to the main "Overall" similarity score. These scores are based on a 4-point scale, from most (1) to least (4) similar. The aim of the task is to predict the "Overall" score, but the participants can make use of the other auxiliary scores. For the final evaluation, the organisers proposed to use Pearson's correlation between the predicted similarity score and the gold "Overall" similarity score. Correlation scores for different article pairs are then averaged to compute the final score.

A script[6] provided by the task organisers enabled downloading news article pairs pertaining to train and test subsets. This script retrieves the URLs of news articles, using the Internet Archive[7] to support this retrieval.

4 Data Preprocessing

Upon downloading the news articles with the script provided by the organisers, we noticed that some texts were not correctly parsed: they contained only non-relevant texts (e.g. error messages), without the main content. Therefore, we removed such problematic text pairs from the training set, based on the manually collected list of strings and heuristic rules for them, mostly for English (e.g. texts that started with "Get full access to" or "TAKE A FREE TRIAL"); pairs with short texts (< 400 characters) that contained mostly metadata were also removed from the training set. Finally, our training set consists of 4,228 text pairs. From texts both in the training and test sets, we also applied heuristic rules based on the manually collected list of strings (mostly for English) to remove non-relevant text fragments (e.g. "Your browser does not support the audio element" or "Share this item on Twitter") from texts.

All texts (in the training and test sets) were split into sentences using the stanza[8] python package, which supports all the languages under consideration. We relied on the assumption that news texts usually contain the most important information at the beginning, while the remainder of the story contains other less important details, according to the 'inverted pyramid' principle (Pöttker, 2003). According to this principle, the first paragraph or sentences of a news article are generally expected to cover the main information addressing the who, what, when, where, and why of a story. Therefore, aiming to improve both the efficiency and effectiveness of our system, we only take the first 10 sentences of all non-English texts and we translate them into English using m2m100 models (Fan et al., 2021): 1.2B model[9] for the training set and 418M[10] model for the test set (the latter model was

[5]https://huggingface.co/sentence-transformers/paraphrase-xlm-r-multilingual-v1

[6]https://github.com/euagendas/semeval_8_2022_ia_downloader

[7]http://web.archive.org

[8]https://github.com/stanfordnlp/stanza

[9]https://huggingface.co/facebook/m2m100_1.2B

[10]https://huggingface.co/facebook/

Model	Train set	En-En data
(1)		
4 sent	0.51	0.65
5 sent	0.51	0.67
3 sent merged	0.64	0.70
(2)		
4 sent	0.61	0.72
5 sent	0.61	0.73
3 sent merged	0.66	0.73
(3)		
3 sent merged	0.63	0.70

Table 1: Pearson's correlation scores for train set for pre-trained language models: LaBSE (1), paraphrase-xlm-r-multilingual-v1 (2), and distiluse-base-multilingual-cased-v1 (3). Scores are presented for the whole train set and only for the English-English pairs of the train set.

chosen, as it is faster). Given that similar events are expected to mention the same or related named entities, we used the spacy[11] python package on English texts (original and translated) for named entity recognition.

All the preprocessing, fine-tuning and evaluation tasks were performed using Google Colab and two NVIDIA GeForce GTX 1080 Ti Graphics Cards (22 GB RAM).

5 Description of Models and Experimental Setup

In order to detect similarity at the document level, we focused on sentence-transformer models. All similarity scores were calculated based on cosine similarity. During the development stage, we conducted all our experiments on the training set, with a held-out subset reserved for evaluation.

We tested different transformer models, including LaBSE (Feng et al., 2020) and paraphrase-xlm-r-multilingual-v1[12]. Based on the assumption that the main information about a story is expected to appear in the beginning of news articles, different setups for both models were taken: mean semantic similarity (cosine similarity) for all pairwise sentence-to-sentence similarities for the first 4 sentences in both articles (`4 sent`) and the first 5 sentences in both articles (`5 sent`); and the semantic similarity between the merged set of first 3 sentences from each article (`3 sent merged`). To come up with the best approach to submit to the shared task, we conducted three sets of experiments.

Experiment set 1. Table 1 provides performance scores for the different "pure" pre-trained language models, measured based on Pearson's correlation values between predicted and gold similarity scores. Among the sentence selection approaches, the best-performing setup is the one based on the merged combination of the first 3 sentences (`3 sent merged`). Among the pre-trained language models, the paraphrase-xlm-r-multilingual-v1 model performs better than the other one. This finding confirms the notion that LaBSE works less well in detecting similarity of sentence pairs that are not translations of each other other[13] (Reimers and Gurevych, 2020). Multilingual distiluse-base-multilingual-cased-v1[14] model, one of the models recommended[15] for semantic similarity tasks, was also included in experiments, namely, for the best setup (first 3 sentences merged), but performed worse than paraphrase-xlm-r-multilingual-v1 model.

Experiment set 2. In addition, we also conducted some baseline fine-tuning experiments. We took XLSum dataset as the biggest dataset with available news texts that contains most of the languages present in our train and test sets (Arabic, Chinese, English, Spanish, Russian, and Turkish are included; it does not include German, Italian, and Polish). XLSum consists of 1.35 million article-summary pairs from the BBC in 44 languages (Hasan et al., 2021). We selected a smaller sample of 98,697 news texts with as balanced representation of 6 languages as possible (15,000 or fewer random examples for each language). Unsupervised domain-specific fine-tuning of the paraphrase-xlm-r-multilingual-v1 model on the news dataset (1 epoch with MultipleNegativesRankingLoss) did not improve the results: Pearson's correlation score was 0.63 for all text pairs from the preprocessed

m2m100_418M

[11]https://spacy.io/

[12]https://huggingface.co/sentence-transformers/paraphrase-xlm-r-multilingual-v1

[13]https://www.sbert.net/docs/pretrained_models.html#multi-lingual-models

[14]https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v1

[15]https://www.sbert.net/docs/pretrained_models.html#multi-lingual-models

training set, and 0.71 for only English-English pairs. Masked Language Model approach (1 epoch, batch size 16) also yielded worse scores (0.41 and 0.51 respectively).

Experiment set 3. We also conducted more model fine-tuning experiments. For internal evaluation, 10% of the training set was selected as an internal test set (with a balanced representation of English, Spanish, German, and Polish languages), and training was made on the remaining 90% (internal train set). The models distiluse-base-multilingual-cased-v1 and paraphrase-xlm-r-multilingual-v1 were fine-tuned on the internal train set (with MultipleNegativesRankingLoss, 3, 5, 8 epochs with 5 epochs as the best option, the latter one better). The model distiluse-base-multilingual-cased-v1 was also fine-tuned on the aforementioned XlSum dataset part (3 epochs with MultipleNegativesRankingLoss). On the internal test set, the models ensemble of distiluse-base-multilingual-cased-v1 fine-tuned on train set and distiluse-base-multilingual-cased-v1 fine-tuned on news dataset, with addition of named entities intersection ratio, achieved the best score, but still lower than the baseline "pure" models.

Other models considered. Apart from the three main sets of experiments above, we also considered other options. Our experiments with baseline experiments with generative pre-trained models did not lead to competitive results. Experimental results with GPT-Neo 2.7B[16] on English texts yielded only 0.13 correlation score between the perplexity scores for the first sentences and the gold scores. Basic fine-tuning for the mt5-base model[17] (setups up to 5 epochs, the task was handled as a multi-class classification task with 7 labels) did not give relevant results (all texts were misclassified for the highest scores).

Other features considered. Another possibility we considered was to include the dates mentioned in the news articles into our model. This was based on the assumption that news articles that are related to each other are supposed to be published in similar dates. However, our attempts at parsing dates from the articles (using the dateparser and num2words python packages) led to noisy outcomes, so we ended up using a rule-based approach to extract months and days from texts in English (original or translated). We check if both texts contain the same months and days (handled as intersections with two different sets).

[16]https://huggingface.co/EleutherAI/gpt-neo-2.7B
[17]https://huggingface.co/google/mt5-base

6 Results

6.1 Analysis of Results

On the test set, among our submitted results, the setup leading to the best performance score was a "pure" paraphrase-xlm-r-multilingual-v1 model combined with named entities intersections and dates intersections (0.511 Pearson's correlation). This setup performed better than a "pure" paraphrase-xlm-r-multilingual-v1 model (0.502 Pearson's correlation) or a "pure" paraphrase-xlm-r-multilingual-v1 model combined with named entities intersections (0.508 Pearson's correlation), so the features helped improve the model's scores. Named entity and date intersection scores were added to the model's scores using rule-based coefficients selected manually.

The best model also performed slightly better than the ensemble of a "pure" paraphrase-xlm-r-multilingual-v1 model and a distiluse-base-multilingual-cased-v1 model fine-tuned on news (0.502 Pearson's correlation). It shows that more domain-specified models and more complicated fine-tuning techniques should be used for the task. Among the setups we considered, we observe that "pure" models can be deemed stronger baselines.

6.2 Error Analysis

We performed an error analysis to understand why our proposed models yielded moderate performance scores. We identified two main reasons that can inform future directions of our research in improving these models:

1. Text parsing errors in the test set: some texts include, or sometimes solely consist of, meta-information that was not excluded by rules. Our rules did not cover all cases for all languages, as they required language knowledge, and should ideally be further pre-processed to reduce the noise and improve model performance. Examples of these cases include:

- German: *"OK Wir setzen auf unserer Website Cookies und andere Technologien ein, um Ihnen den vollen Funktionsumfang unseres Angebotes anzubieten"*

- German: *"Um die Funktion unserer Website zu verbessern und die relevantesten Nachrichten und zielgrichtete Werbung anzuzeigen, sammeln wir technische anonymisierte Informationen über Sie, unter anderem mit Instrumenten unserer Partner. Ausführliche Informationen zur Datenverarbeitung finden Sie in den Datenschutzrichtlinien. Ausführliche Informationen zu den von uns genutzten Technologien finden Sie in den Regeln der Cookies-Nutzung und des automatischen Einloggens. Indem Sie „Akzeptieren und schließen" anklicken, stimmen Sie ausdrücklich der Verarbeitung Ihrer persönlichen Daten zu, damit das beschriebene Ziel erreicht wird. Ihre Zustimmung können Sie auf die Weise widerrufen, wie in den Datenschutzrichtlinien beschrieben."*

- Italian: *"Informativa Privacy Questo sito utilizza cookies per migliorare servizi ed esperienza dei lettori. Le informazioni raccolte dai cookies sono conservate nel tuo browser e hanno la funzione di riconoscere l'utente quando ritorna sul nostro sito web e aiutare il nostro team a capire quali sono le sezioni del sito ritenute più interessanti ed utili."*

2. Translation errors. It produced model "hallucinations" and repetitions, such as "Chief Executive Officer of the Ministry of Foreign Affairs and Foreign Affairs of the Ministry of Foreign Affairs" or "WASHINGTON-SANA WASHINGTON-SANA WASHINGTON-SANA" instead of the correct English translations. We noticed this to be the case particularly for articles originally written in Chinese and Arabic (translations of more than 30% of test samples in these languages contained such translation errors). Therefore, other translation approaches, i.e. using Google Translate API, might be used to obtain better English translations to detect named entities intersections.

Table 2 provides the best model setup scores, broken down by language. For this analysis, we selected only monolingual language pairs. The best results are for Spanish, French and English, while Arabic, German and Chinese yield the lowest scores. It can be caused by parsing errors, as well as by translation errors. Separate monolingual language models for these languages can be applied in further research, as well as models with better transfer learning techniques for these languages. In the future, further experiments could be conducted for the multilingual XLSum dataset, using different sampling techniques and sample size.

Language	Score	No. of pairs
`French`	0.692	111
`Spanish`	0.678	243
`English`	0.625	236
`Turkish`	0.610	275
`Italian`	0.573	411
`Russian`	0.530	287
`Polish`	0.512	224
`Chinese`	0.426	769
`German`	0.250	608
`Arabic`	0.154	298

Table 2: Pearson's correlation scores for paraphrase-xlm-r-multilingual-v1 model combined with named entities intersections and dates intersections, for different languages from the test set (monolingual text pairs).

7 Conclusion

This paper presents the participation results of our team "dina" in the Multilingual News Similarity shared task held as part of SemEval 2022. We tested a range of state-of-the-art pre-trained multilingual transformer models, which were further tested by incorporating features based on dates, machine translation and named entity recognition. Our best model achieved a Pearson's correlation score of 0.511. It can be considered as a moderate performance score with substantial room for improvement, based on performance scores from other participants in the task, where the best system achieved a score of 0.818. In future experiments on cross-lingual semantic similarity of news texts, we aim to focus on more sophisticated fine-tuning techniques for domain adaptation on a further preprocessed and cleaned dataset.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Mathis Linger and Mhamed Hajaiej. 2020. Batch clustering for multilingual news streaming. In *Proceedings of Text2Story - Third Workshop on Narrative Extraction From Texts co-located with 42nd European Conference on Information Retrieval, Text2Story@ECIR 2020, April 14th, 2020. Vol. 2593 of CEUR Workshop Proceedings*, pages 55–61, Lisbon, Portugal.

Sebastião Miranda, Artūrs Znotiņš, Shay B. Cohen, and Guntis Barzdins. 2018. Multilingual clustering of streaming news. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4535–4544, Brussels, Belgium. Association for Computational Linguistics.

Horst Pöttker. 2003. News and its communicative quality: the inverted pyramid—when and why did it appear? *Journalism Studies*, 4(4):501–511.

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

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525.

Junfeng Tian, Zhiheng Zhou, Man Lan, and Yuanbin Wu. 2017. ECNU at SemEval-2017 task 1: Leverage kernel-based traditional NLP features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 191–197, Vancouver, Canada. Association for Computational Linguistics.

TCU at SemEval-2022 Task 8: A Stacking Ensemble Transformer Model for Multilingual News Article Similarity

Xiang Luo[1] ,Yanqing Niu[1] and Boer Zhu[2]
[1]School of Mathematics and Statistics, South Central University for Nationalities, Wuhan, China
[2]School of Computer Science, South Central University for Nationalities, Wuhan, China
Contact: niuyanqing@mail.scuec.edu.cn

Abstract

Previous studies focus on measuring the degree of similarity of texts by using traditional machine learning methods, such as Support Vector Regression (SVR). Based on Transformers, this paper describes our contribution to SemEval-2022 Task 8 Multilingual News Article Similarity. The similarity of multilingual news articles requires a regression prediction on the similarity of multilingual articles, rather than a classification for judging text similarity. This paper mainly describes the architecture of the model and how to adjust the parameters in the experiment and strengthen the generalization ability. In this paper, we implement and construct different models through transformer-based models. We applied different transformer-based models, as well as ensemble them together by using ensemble learning. To avoid the overfit, we focus on the adjustment of parameters and the increase of generalization ability in our experiments. In the last submitted contest, we achieve a score of 0.715 and rank the 21st place.

1 Introduction

Providing computer the ability to understand the abstract meaning of real world is a fundamental tasks. Given a pair of news articles, this task seek to evaluate the semantic similarity between them, which focuses on the real world-happenings covered in the news articles. It's a regression problem for measuring similarity of multilingual texts.

Previous studies measured the similarity between texts by using traditional machine learning methods, such as using Support Vector Regression (SVR) (Šarić et al., 2012). Recently many deep learning methods came out, such as pre-trained model. It had attracted the interest of researchers and had shown good result. For example, Exploring Bidirectional Encoder Representations from Transformers (BERT), XLNet and Robustly optimized BERT approach (RoBERTa) and finally got a good ranking (Yang et al., 2020). And a new hybridized approach using Weighted Fine-Tuned BERT Feature extraction with Siamese Bi-LSTM model has been implemented. It is employed for determining question pair sets using Semantic-text-similarity from Quora dataset (Viji and Revathy, 2022). These novel deep learning methods have performed well.

In this study, we explored some transformer-based models. We had employed BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), A Lite BERT for Self-supervised Learning of Language Representations (ALBERT) (Lan et al., 2020), DistilBERT (Sanh et al., 2020; Gou et al., 2021), besides this, we used them as base models, merged them together with ensemble learning, and the prediction result is used as a new training set, and then SVR is used as the meta model. The training set generated by the base model is put into the meta-model. Afterwards, the final result is predicted by the meta-model. The advantage of the pre-trained model is that the upstream corpus has already trained the parameters of the model well. We only need to fine-tune it, and we don't need a huge training set for training (Kong et al., 2022).

The rest of this paper is organized as follows. Section 2 describes all the models which are used for measuring similarity between sentence pairs. Experimental results are summarized in Section 3. Conclusion is drawn in Section 4.

2 Model Description

This section will describe what models we have used, and how they organized. Because of the attention mechanism, pre-trained model had been made a huge success in Nature Language Processing (NLP). We use transformer-based models such as BERT, RoBERTa, ALBERT, DistilBERT to produce hidden representations. Then, a stacking ensemble strategy was used to ensemble the results. The details are presented as follow.

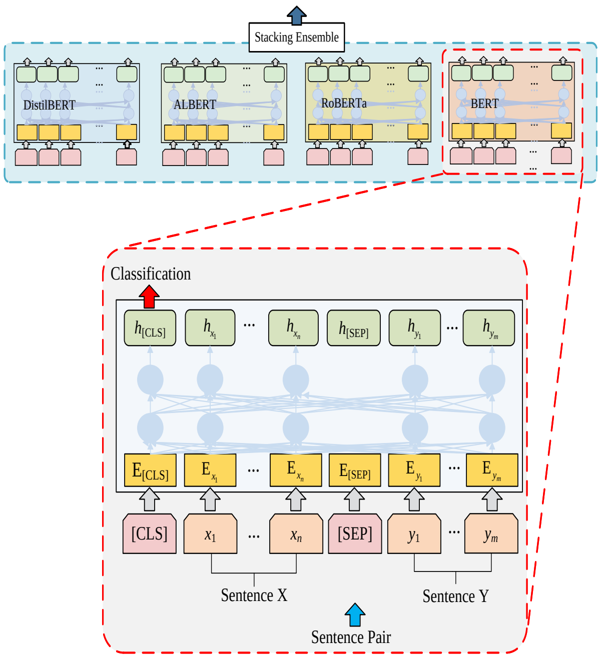


Figure 1: The overall architecture of the proposed method.

2.1 Base Models

Based on the self-attention mechanism, transformer-based model become a popular method in NLP. The models, such as BERT, ALBERT, RoBERTa, DistilBERT are variants on the improvement of the transformer architecture (Huang et al., 2021).

For four Transformer encoders, we applied a similar architecture as sentence pair classification to learn representation for the final regression. Given two sentences and $\mathbf{X} = [x_1, x_2, ..., x_n]$ and $\mathbf{Y} = [y_1, y_2, ..., y_m]$. The model used WordPiece tokenizer to obtain subwords sequences. Two special tokens, i.e., [CLS] and [SEP], were added to the beginning of the whole sequence and between two sentences.The model architecture we use is shown in Fig. 1 The details of each Transformer encoder are presented as follows.

BERT. BERT stands for Bidirectional Encoder Representations from Transformers. BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. The checkpoint we use is “bert-base-multilingual-cased”, which uses 12-layer, 768-hidden, 12-heads, 110M parameters. Trained on cased text in the top 104 languages with the largest Wikipedias. The outputs tensor contains the batch_size, sequence_length, hidden_state and we use the first token to regress (Zhang et al., 2021). Moreover, BERT uses character-level BPE encoding.

RoBERTa. RoBERTa is a robustly optimized BERT pretraining approach, it’s an improved recipe for training BERT models, that can match or exceed the performance of all of the post-BERT methods. Our modifications are simple, they include: (1) training the model longer, with bigger batches, over more data; (2) removing the next sentence prediction objective; (3) training on longer sequences; and (4) dynamically changing the masking pattern applied to the training data. The checkpoint we use is “roberta-base”, which uses 12-layer, 768-hidden, 12-heads, 125M parameters. RoBERTa using the BERT-base architecture. So the output of RoBERTa is similar to BERT. The token of RoBERTa is called ‘sos’. After that, RoBERTa uses byte-level BPE encoding.

ALBERT. ALBERT is a lite BERT for self-supervised learning of language representations which lead to models that scale much better compared to the original BERT and it uses a self-supervised loss that focuses on modeling inter-sentence coherence, and show it consistently helps downstream tasks with multi-sentence inputs. The checkpoint we use is “albert-base-v2”, which uses 12 repeating layers, 128 embedding, 768-hidden,

MODEL	PEARSON	MSE
DistilBERT+ALBERT+BERT+RoBERTa	**0.880**	**0.298**
DistilBERT+BERT+ALBERT	0.879	0.301
DistilBERT+ALBERT+RoBERTa	0.815	0.450
DistilBERT+BERT+RoBERTa	0.879	0.301
ALBERT+RoBERTa	0.815	0.452
ALBERT+RoBERTa+BERT	0.880	0.298
BERT+ALBERT	0.877	0.306
BERT+RoBERTa	0.878	0.303
BERT+DistilBERT	0.879	0.301
ALBERT+DistilBERT	0.805	0.472
RoBERTa+DistilBERT	0.799	0.483
DistilBERT	0.749	0.587
BERT	0.874	0.310
ALBERT	0.792	0.504
RoBERTa	0.790	0.509

Table 1: The Pearson Score and MSE of Each Model in Test Data.

12-heads, 11M parameters. ALBERT base model with no dropout, additional training data and longer training. And ALBERT is also using the first position of token to regress which is similar to the token of BERT.

DistilBERT. DistilBERT is a distilled version of BERT,which pre-train a smaller general-purpose language representation model and can then be finetuned with good performances on a wide range of tasks like its larger counterparts. The checkpoint we use is "distilbert-base-cased", which use 6-layer, 768-hidden, 12-heads, 65M parameters. The DistilBERT model distilled from the BERT model bert-base-cased checkpoint. And the tokenization of DistilBERT is also similar to BERT.

2.2 Ensemble Learning

In ensemble learning, we train multiple models to solve the same problem and combine them to get better results. The most important assumption is that when weak models are combined correctly, we can get more accurate or robust models. We decide to use stacking as our ensemble learning model. Stacking usually considers heterogeneous weak learners and stacking learning to combine base models with meta-models.

We concatenate the output of the base regressor. Then we put the output into the meta-model which we use SVR as. After that, we use grid sweep to get the optimizer parameters, using SVR to output the prediction results.

We divide the base models into RoBERTa, BERT, DistilBERT and ALBERT, we first train each base model and save the best performing model, and then we combine them separately. We use SVR as our meta-model, take the output of the base model as the input of the meta-model, and then train the input data through the meta-model.The data we use is the test set divided from the training set, and the pearson score and MSE are used to judge the quality of the entire model. For different base models, we will adjust the parameters on the meta-model, so that each set of base models perform as best as possible. The final result is shown in Table 1.

3 Experimental Results

In this section, we will describe how the whole experimental part is done, and the main focus will be on the part that implements the model. The experimental part will be divided in to 5 parts as follows.

3.1 Datasets

In raw dataset, there are many descriptions about the news from different part such as Geography, Entities, Time, Narrative, Overall, Style, Tone. However, as the issue overview said, the annotation task consists of carefully reading each of the two news articles in a pair and selecting the Overall similarity score. As written in the description, systems will be evaluated on their ability to estimate the Overall similarity between two pairs of news stories, not any of the other scores. So we focus on the relationship between the Overall and the sentences, we use sentences separately.

3.2 Evaluation Metrics

In the evaluation dataset, we find that the evaluation dataset is mixed with many languages that does not appear in the training set, such as Chinese, so we try to add some languages that appear in the evaluation dataset in the text back translation. We use Google's translation API, we translate non-Chinese source language into Chinese, and increase the amount of train data through this form The submissions were scored using Pearson's correlation with the 'Overall' column. We use Pearson's correlation as our evaluation metrics. The definition of Pearson's correlation is as follows:

$$\rho_{x,y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\sqrt{E(Y^2) - E^2(Y)}} \quad (1)$$

where the X is the predicted value, and Y is the ground-truth value. Further, mean squared error is calculated as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \overset{\wedge}{y_i}\right)^2 \quad (2)$$

where y_i is the predicted value, and $\overset{\wedge}{y_i}$ is the ground-truth value.

3.3 Implementation Details

The train data is split into 3 parts, the base data which is for the part of base regressor, the ensemble data which is for the part of ensemble learning and the test data for the final test, and the test data is used in Table 1. According to the paper (Sun et al., 2020), we truncate the middle part of the text of tokens larger than 512.

In the part of base regressor. At first, we clean the data, fill and delete the missing values in the dataset. The cleaned data is split to train data, validation data and test data. The raw sentences are put into the tokenizer that the tokenizer is corresponding to each model such as BertTokenizer and the tokenizer uses the upstream model to complete the tokenization. After that, we use tf.data.dataset to wrap the tokenizer so that the it can be used by regressor. Then The transformer-based model is used as our regressor such as BertForSequenceClassification, when the parameter num_labels=1, it can be used as a regressor. Adam (Kingma and Ba, 2017) is chosen as our optimizer and MSE is used as the loss function and Pearson as evaluation metrics to train the model. The validation data is used in each epoch to judge the performance of model. After training, we use the trained model to make predictions. The test data is used to detect which model perform better. We choose the best model and save it. In addition, we start to tune the parameters such as learning rate, weight decay and epochs. Because of the limit of the memory, we set batch size to 8 so that the model can run smoothly. We use grid search so that we can find the optimal parameters accurately and quickly.

3.4 Parameters Fine-tuning

WeightsBiases is a visualization tool to supervise the model training process. When tuning the parameters, we use it so that we can record the change curve of the parameters and easy to find optimal parameters (Wang et al., 2022). After tuning the parameters, we use test data to test. Finally, we set learning rate to 1e-5, set weight decay to 1e-6, set epochs to 50. And the C parameter of the SVR is set to 10, and the C parameter is essentially a regularisation parameter, which controls the trade-off between achieving a low error on the training data and minimising the norm of the weights.The kernel parameter is set to "linear". We show the adjustment process of our parameters through two line graphs fig 2 and fig 3.

3.5 Comparative Results

We use the test data which is split from train data. We use this test data to calculate the Pearson score and MSE for each model's predictions, and the result is shown in Table 1.

We submit these models to the organizer. But the method of stacking doesn't achieve a good result. The model of BERT gets the best score in this competition which the score is 0.715. After submitting our final prediction, the best score is obtained by BERT instead of stacking. We submit these models to the final evaluation, but the only scores returned to us are BERT and stacking. The model of BERT got 0.715, the model of stacking only got 0.464. The reason why we set the topic as stacking ensemble learning is because we spend most of our time on it during the entire competition, and we think it does achieve better results on the training set, so we set the topic to stacking ensemble learning. And we reflect that it may be caused by insufficient generalization ability of our base model or meta-model. It may also be caused by insufficient differences in the base model during

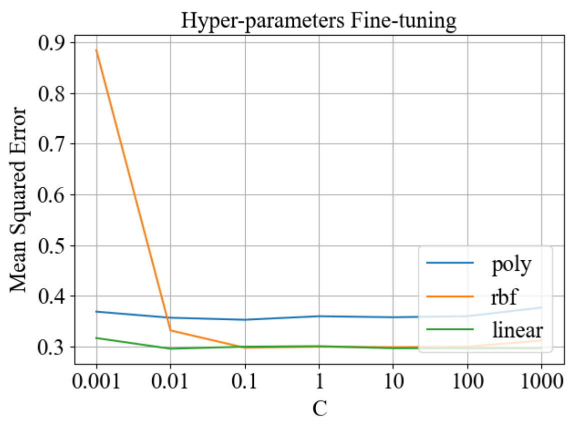


Figure 2: The performance of different parameters on MSE.

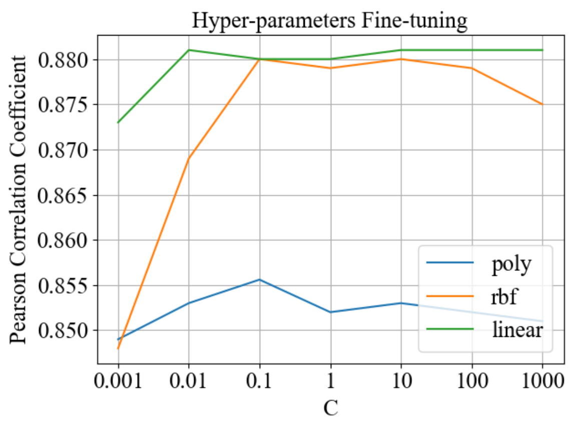


Figure 3: The performance of different parameters on PEARSON.

stacking, or it may be that the selected parameter adjustment method does not make the parameters optimal. This is what we reflect on after getting the feedback.

4 Conclusions

In this paper, we are participating in SemEval-2022 Task 8 (Chen et al., 2022). In this task, we perform regression prediction on the similarity of multilingual news articles, and we use various methods such as BERT, ALBERT, RoBERTa, DistilBERT, and the stacking method built with them as the base model. The model we proposed can effectively predict this task. Among the multiple models we submitted, the BERT model we finally submitted achieved the best score with a score of 0.715, ranking 21st in the leaderboard. At present, in terms of deep learning, the processing methods of multilingual texts have not been widely popularized. So in the future, we hope to go further in the processing of multilingual texts.

References

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flock, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. 2021. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6):1789–1819. ArXiv: 2006.05525.

Bo Huang, Yang Bai, and Xiaobing Zhou. 2021. hub at SemEval-2021 Task 2: Word Meaning Similarity Prediction Model Based on RoBERTa and Word Frequency. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 719–723, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*. ArXiv: 1412.6980.

Jun Kong, Jin Wang, and Xuejie Zhang. 2022. Hierarchical BERT with an adaptive fine-tuning strategy for document classification. *Knowledge-Based Systems*, 238:107872.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *arXiv:1909.11942 [cs]*. ArXiv: 1909.11942.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. ArXiv: 1907.11692.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv:1910.01108 [cs]*. ArXiv: 1910.01108.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to Fine-Tune BERT for Text Classification? *arXiv:1905.05583 [cs]*. ArXiv: 1905.05583.

D. Viji and S. Revathy. 2022. A hybrid approach of Weighted Fine-Tuned BERT extraction with deep

Siamese Bi – LSTM model for semantic text similarity identification. *Multimedia Tools and Applications*.

Jin Wang, You Zhang, Liang-Chih Yu, and Xuejie Zhang. 2022. Contextual sentiment embeddings via bi-directional GRU language model. *Knowledge-Based Systems*, 235:107663.

Xi Yang, Xing He, Hansi Zhang, Yinghan Ma, Jiang Bian, and Yonghui Wu. 2020. Measurement of Semantic Textual Similarity in Clinical Texts: Comparison of Transformer-Based Models. *JMIR Medical Informatics*, 8(11):e19735.

You Zhang, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. 2021. MA-BERT: Learning Representation by Incorporating Multi-Attribute Knowledge in Transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2338–2343, Online. Association for Computational Linguistics.

Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. 2012. TakeLab: Systems for Measuring Semantic Text Similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 441–448, Montréal, Canada. Association for Computational Linguistics.

Nikkei at SemEval-2022 Task 8: Exploring BERT-based Bi-Encoder Approach for Pairwise Multilingual News Article Similarity

Shotaro Ishihara **Hono Shirai**
Nikkei, Inc. Chiyoda, Tokyo, Japan
{shotaro.ishihara,hono.shirai}@nex.nikkei.com

Abstract

This paper describes our system in SemEval-2022 Task 8, where participants were required to predict the similarity of two multilingual news articles. In the task of pairwise sentence and document scoring, there are two main approaches: Cross-Encoder, which inputs pairs of texts into a single encoder, and Bi-Encoder, which encodes each input independently. The former method often achieves higher performance, but the latter gave us a better result in SemEval-2022 Task 8. This paper presents our exploration of BERT-based Bi-Encoder approach for this task, and there are several findings such as pretrained models, pooling methods, translation, data separation, and the number of tokens. The weighted average ensemble of the four models achieved the competitive result and ranked in the top 12.

1 Introduction

Measuring sentence and document similarity is a task that has been studied for many years in the field of natural language processing. One of the applications is to identify whether news articles address the same subject. If news articles can be properly clustered, they can be used for a wide range of purposes, such as recommendation and displaying related articles. SemEval-2022 Task 8 attempts to tackle this task with multilingual news articles (Chen et al., 2022).

Nowadays, it is common for this kind of tasks to use transformer-based models like BERT (Devlin et al., 2019). There are several research directions, including post-processing (Li et al., 2020; Wang and Kuo, 2020), unsupervised learning (Zhang et al., 2020; Tiyajamorn et al., 2021), and supervised learning (Reimers and Gurevych, 2019; Thakur et al., 2021; Feng et al., 2020; Jiang et al., 2022). Here, since a labeled dataset was provided, we decided to use a supervised learning approach.

It is standard practice to combine two sentences as input when dealing with pairwise similarity of

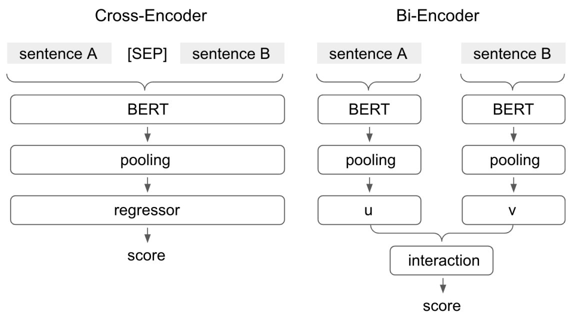


Figure 1: The architectures of Cross-Encoder and Bi-Encoder. The input of the Cross-Encoder architecture is two sentences joined by `SEP`, and through BERT and a pooling layer, the regressor outputs a score. In the Bi-Encoder architecture, each sentence is transformed by BERT and pooling layers, and the score is calculated through interaction of the two vectors.

sentences in a supervised learning approach with BERT (Lin et al., 2021; Reimers and Gurevych, 2019). In contrast, we chose an approach that embeds each sentence separately. Figure 1 shows these two approaches, named Cross-Encoder and Bi-Encoder to follow the previous research (Thakur et al., 2021).

This paper describes our system in SemEval-2022 Task 8. First, we explain the experimental results by adopting the Bi-Encoder architecture rather than Cross-Encoder. We also present the following research questions: 1) which pretrained model works well when dealing with multilingual news articles, 2) what kind of pooling method is proper for this task, 3) is it useful for translating the other language into English, and 4) is there some effect of data splitting and max length? Our code is available at https://github.com/upura/semeval2022-task8-multilingual-news-article-similarity.

2 Task Description

SemEval-2022 Task 8 provides a dataset that contains pairs of news articles. The dataset contains the information like the language and URL of each arti-

Table 1: Language pairs in the training and evaluation dataset. There are eight language pairs in the training dataset, and an additional ten language pairs appear in the evaluation dataset.

language-language	traning	evaluation
English-English	1800	236
German-German	857	608
German-English	577	185
Spanish-Spanish	570	243
Turkish-Turkish	465	275
Polish-Polish	349	224
Arabic-Arabic	274	298
French-French	72	111
Russian-Russian		287
Chinese-Chinese		769
Spanish-English		496
Italian-Italian		411
Polish-English		64
Chinese-English		213
Spanish-Italian		320
German-French		116
German-Polish		35
French-Polish		11

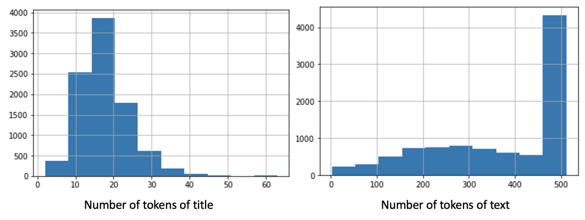


Figure 2: The histogram of the number of tokens. Left and right shows the number of tokens in the title and body text.

cle and the target score named `Overall`. Training dataset consists of 4,964 pairs of articles. Table 1 shows the number of each language pair. It is interesting that a large number of new language pairs appear in the evaluation dataset. It is inferred that building machine learning models using only the language pairs in the training dataset results in poor performance for these unknown language pairs. Therefore, it is essential to address multilingual datasets in some way. Information such as the title and body text of the article is available by scraping the data from the URL[1]. Figure 2 shows the histogram of the number of tokens[2]. Most titles are around 20-30 tokens, and the body text often has a maximum token length of 512. Each `Overall` score is calculated by averaging the annotators' scores.

The evaluation dataset, in which the labels are

[1]`https://github.com/euagendas/semeval_8_2022_ia_downloader`

[2]As a tokenizer, we used the pretrained BERT model named `bert-base-multilingual-uncased`.

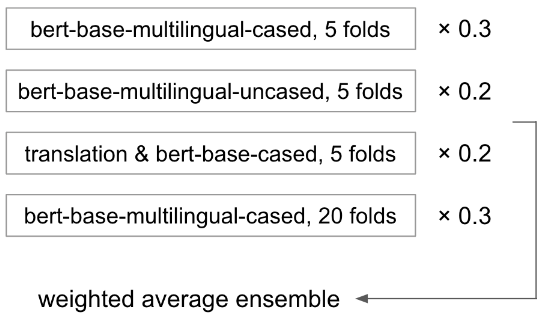

bert-base-multilingual-cased, 5 folds
× 0.3
bert-base-multilingual-uncased, 5 folds
× 0.2
translation & bert-base-cased, 5 folds
× 0.2
bert-base-multilingual-cased, 20 folds
× 0.3
weighted average ensemble

Figure 3: The overview of the developed system. Four neural networks output predictions and the final result is calculated by weighted average ensemble.

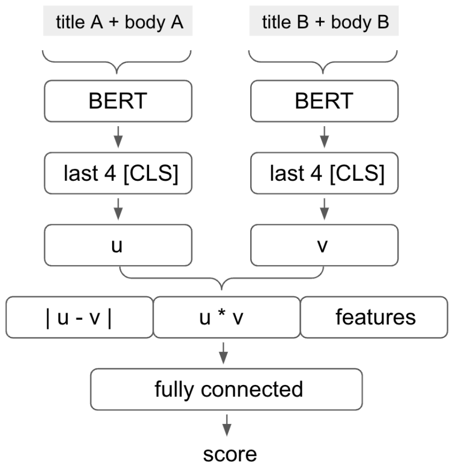


Figure 4: Base architecture of each neural network in the developed system.

hidden from the participants, consists of 4,902 pairs. The ranking of the task is determined by Pearson's correlation between the labels in the evaluation dataset and the submitted predictions. Participants are allowed to submit their predictions five times per day, but none of the scores could be observed until the deadline. The leaderboard is also kept private until the end, so it is impossible to know the scores of the other teams.

3 System Overview

The developed system is outlined in Figure 3. The final prediction is calculated by a weighted average of the output of the four neural networks. Figure 4 illustrates the base architecture of each neural network. We use a Bi-Encoder approach where the two texts are entered into separate BERT. Each input is the combined title and body text of the article. As a pooling method, the representations of the last four `CLS` tokens are concatenated. `CLS` is the token to be attached to the beginning of an input of BERT. At the end, the score is given based on the interaction of the two sentence vectors.

The rest of this section describes the key points of the system. First, we consider whether Cross-Encoder or Bi-Encoder should be used. Next, we address the perspectives of the research questions listed in Section 1: pretrained models, pooling methods, translation, data splitting, and maximum length. Finally, we explain the ensemble of models.

3.1 Cross-Encoder vs Bi-Encoder

In the Bi-Encoder architecture, each sentence is transformed into an embedding by BERT, and the sentence vectors are obtained by a pooling layer. Denote two vectors A and B, then their interactions are designed as `distance` and `angle` with reference to (Tai et al., 2015) in the following formula. Here, `distance` is calculated as an element-wise absolute error, and `angle` is calculated as an element-wise multiplication.

$$\texttt{distance} = |A - B|, \quad \texttt{angle} = A \otimes B$$

In addition to `distance` and `angle`, traditional features are also created and combined. We use Jaccard Index (Jaccard, 1912), Dice Index (Dice, 1945), and cosine similarity. These features may not work when dealing with pairs with different languages. However, when building a prediction model with LightGBM (Ke et al., 2017) using only these features, Pearson's correlation achieved 0.2989 in the evaluation data set. We believe that these features can contribute to some extent and combine them into the layer.

A comparison between Cross-Encoder and Bi-Encoder is reported in Section 4.1. The Cross-Encoder architecture for the comparison is shown in Figure 5. We use two BERT models, one with concatenated article headlines and one with concatenated body text. The pooling method and features are fixed to the same settings. The regressor is a simple fully connected layer.

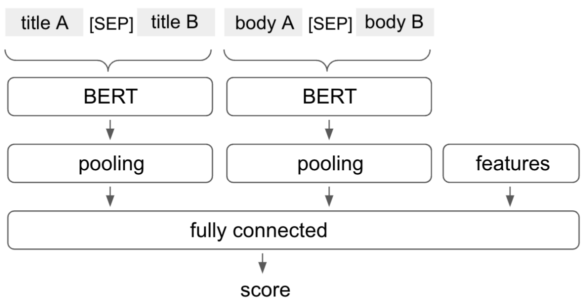


Figure 5: Cross-Encoder architecture for the comparison with Bi-Encoder architecture shown in Figure 4.

3.2 Pretrained Models

We considered various pretrained BERT models of Hugging Face Transformers (Wolf et al., 2020). There are some multi-lingual pretrained models such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) available[3] and the following models are selected as candidates.

- `bert-base-multilingual-uncased`
- `bert-base-multilingual-cased`
- `xlm-roberta-base`

Since none of the models performed badly, we used all of them for the final submission as shown in Figure 3. A comparison among models is reported in Section 4.2.

3.3 Pooling Methods

There are several ways to extract the sentence vector from the output through BERT. One of the simplest ways is to use the embedding of the `CLS` token in the final layer, but some methods are proposed to use information from other layers as well. A number of experimental results have been reported (Reimers and Gurevych, 2019; Gao et al., 2021; Jiang et al., 2022; Conneau and Kiela, 2018), but we believe that the results depend largely on the individual task. Therefore, we consider the following four methods.

- `CLS`: Concatenate the last four representations of `CLS` token.
- `CNN`: Use the convolutional neural network (CNN) to extract sentence vectors.
- `LSTM`: Use the long short-term memory (LSTM) for extracting sentence vectors.
- `MAX`: Use max-pooling to extract sentence vectors.

On the basis of the results of our experiments, we adopted the first method. A comparison among the pooling methods is reported in Section 4.3.

3.4 Translation

One of the ways of dealing with multilingual datasets is the translation. Here we examine a method of translating all datasets into English and using pretrained models in English.

[3] `https://huggingface.co/docs/transformers/multilingual`

Table 2: Experimental results of Pearson's correlation for the validation and evaluation dataset. The columns named "pool" and "length" represent pooling methods and max length respectively.

id	architecture	model	pool	folds	length	validation	evaluation
0	Cross-Encoder	`bert-base-multilingual-cased`	`CLS`	5	512	0.7045	0.6188
1	Bi-Encoder	`bert-base-multilingual-cased`	`CLS`	5	512	0.7688	0.6922
2	Bi-Encoder	`bert-base-multilingual-uncased`	`CLS`	5	512	0.7627	0.6940
3	Bi-Encoder	`xlm-roberta-base`	`CLS`	5	512	0.7118	0.6153
4	Bi-Encoder	`bert-base-multilingual-cased`	`CNN`	5	512	0.4892	0.3269
5	Bi-Encoder	`bert-base-multilingual-cased`	`LSTM`	5	512	0.4979	0.3271
6	Bi-Encoder	`bert-base-multilingual-cased`	`MAX`	5	512	0.7221	0.6313
7	Bi-Encoder	translation with `bert-base-uncased`	`CLS`	5	512	0.7505	0.6748
8	Bi-Encoder	`bert-base-multilingual-cased`	`CLS`	20	512	0.7853	0.7107
9	Bi-Encoder	`bert-base-multilingual-cased`	`CLS`	5	256	0.7427	0.6616
10	Bi-Encoder	`bert-base-multilingual-cased`	`CLS`	5	128	0.7137	0.6355
11	Bi-Encoder	`bert-base-multilingual-cased`	`CLS`	5	64	0.6744	0.5782
12		weighted average ensemble				**0.7902**	**0.7425**

We use Googletrans[4] for the translation, and `bert-base-cased` as a pretrained model. In conclusion, the translation approach did not improve the performance of the multilingual models. A comparison result is shown in Section 4.4.

3.5 Data Splitting and Max Length

We also investigate the effect of data splitting and max length. The number of data partitions in cross validation (Blum et al., 1999) affects the number of available training samples. When the size of the training dataset is not very large, as in this task, it may affect the performance. The setting of the max length based on the distribution of the token size introduced in Section 2 is also an adjustable element. In general, news articles contain important information early in the article, so there is a possibility that a smaller max length works well.

For data splitting, it was suggested through the experiment that the larger the number of splits, that is, the larger the training dataset, the higher the performance. We set the max length to 512, because, in contrast to the hypothesis, the smaller max length led to the poor performance. Both comparison results are shown in Section 4.5.

3.6 Weighted Average Ensemble

In the weighted average of the models, the performance on the validation dataset was taken into account to determine the models to be used and their weights. In the final submission, we used the average of all models obtained in the cross validation process. In case of folds set 5, five models were generated. This means that $(5+5+5+20)$ models were used in Figure 3. The improvement through the ensemble is reported in Section 4.6.

[4]`https://github.com/ssut/py-googletrans`

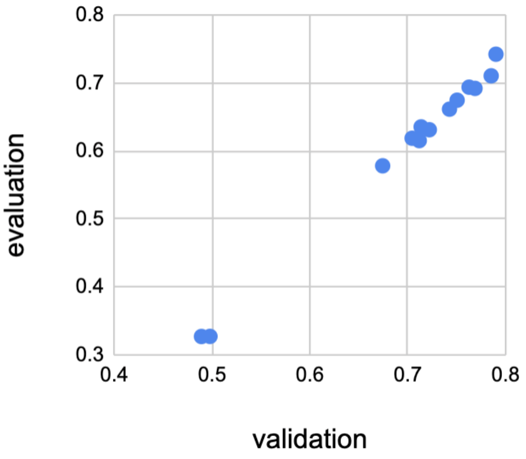


Figure 6: Scatter plot of Pearson's correlation for the validation and evaluation dataset. It can be observed that the scores are correlated.

4 Results

This section reports the experimental results that facilitated the design of the system described in the previous section. Table 2 lists the scores of Pearson's correlation for the validation and evaluation dataset.

The validation dataset is extracted from the training dataset. The column named "folds" shows the number of folds in cross validation. That is, in case of folds set 5, 20 % of the training dataset is removed for the validation. Each model is trained for seven epochs, and the scores with the best performance on the validation dataset are reported. In our experimental setting, the performance for the validation dataset converged after about five training epochs.

The scatter plots of the performance for the validation and evaluation datasets are shown in Figure 6, and it can be observed that they are correlated. It is suggested that the improved performance on the validation dataset is also useful on the evaluation dataset, and that the validation framework works well.

The rest of this section describes the result in detail from the same perspective as in the previous section. The models in Table 2 are referred to as experiments 0-12 respectively, and their performances are compared for the discussions.

4.1 Cross-Encoder vs Bi-Encoder

Comparing the results of experiments 0 and 1, we see that the architecture of Bi-Encoder worked well rather than Cross-Encoder. There is a large difference of more than 0.06 in Pearson's correlation. The results are only for this task and our experimental setup and should not be overly generalized. However, it is an interesting case study, as the results contrast with the use of Cross-Encoder in some of the previous studies presented in Section 1. The results suggest that it is important to try both Cross-Encoder and Bi-Encoder in the search for high performance.

4.2 Pretrained Models

The results of experiments 1-3 show the performance of each pretrained model. The model with `bert-base-multilingual-uncased` and `bert-base-multilingual-cased` performed better than `xlm-roberta-base`.

4.3 Pooling Methods

Seeing the results of experiments 1 and 4-6, we can say that the best pooling method in this task is `CLS`. It outperformed the other three methods.

4.4 Translation

The results of experiment 7 show that the translation approach did not improve the performance from experiment 1.

4.5 Data Splitting and Max Length

In experiment 8, the number of folds was changed from 5 to 20. This was not an exact comparison since the validation dataset was changed, but there was 0.03 improvement in Pearson's correlation from experiment 1. For max length, we examined values from 64 to 256 in experiments 9-11. It was observed that the performance was getting worse as the max length was decreased.

Table 3: Correlation between the experiments used in the final submission (1, 2, 7, and 8).

id	2	7	8
1	0.9156	0.8812	0.9348
2	-	0.8752	0.9105
7	-	-	0.8730

Table 4: Median of absolute error and the number of samples for the evaluation dataset for each language pair. The symbol ✓ means the language pair is included in the training dataset.

language-langage	median	samples	training
German-English	**0.2971**	185	✓
Chinese-English	0.4092	213	
French-French	0.4213	111	✓
Spanish-Italian	0.4251	320	
Polish-English	0.4286	64	
English-English	0.4662	236	✓
Spanish-Spanish	0.5223	243	✓
Spanish-English	0.5239	496	
Polish-Polish	0.5256	224	✓
Italian-Italian	0.5444	411	
Chinese-Chinese	0.5450	769	
Russian-Russian	0.5722	287	
Turkish-Turkish	0.5789	275	✓
French-Polish	0.5855	11	
German-German	0.6030	608	✓
Arabic-Arabic	0.6765	298	✓
German-French	0.6905	116	
German-Polish	**0.7751**	35	

4.6 Weighted Average Ensemble

The results of experiment 12 show that the weighted average of the models boosted the performance. It is important to highlight that when comparing experiment 8 and 12, the performance improved by only 0.005 on the validation dataset, but by more than 0.03 on the evaluation dataset. Table 3 describes the correlation between the experiments used in the final submission (1, 2, 7, and 8). We can see that there is a high similarity between experiments 1 and 8, where the only difference is the number of folds. It is also observed that the translation approach contributes to the diversity, because its correlation is low. This was our submission for SemEval-2022 task 8. The score 0.7425 for the evaluation dataset ranked 12th out of 32 teams.

4.7 Error Analysis

Here we describe the analysis results using labels of the evaluation dataset. First, a comparison of each pair of languages shows the performance differences observed in Table 4. The median absolute error for German-English is 0.2971, whereas German-Polish is 0.7751. German-German performs relatively poorly, and their proportion in the evaluation dataset is high as shown in Table 1. Focusing on these language pairs may improve the overall performance of the system. It is noteworthy that even language pairs that are not part of the training dataset, such as Chinese-English, show excellent performance. It can be suggested that models pretrained in multilingual languages worked well.

Next, we identified the problem in obtaining the article body text by checking extremely incorrectly predicted samples. For example, consider the sample with `pair_id` equals `1512411298_1512618793` where the system predicted 3.555 and the correct answer was 1.000. We checked the body text of the article `1512618793` and found that the extracted text by the script was different from the actual body text for some reasons such as the page layout like advertisements or related articles.

5 Conclusion

This paper presented our exploration of BERT-based Bi-Encoder approach for SemEval-2022 task 8. The experiment showed that Bi-Encoder architecture worked better than Cross-Encoder. There are several findings, such as pretrained models, pooling methods, translation, data separation, and the number of tokens. The exploration of these different variants led to the creation of several diverse models. Finally, a weighted average ensemble of the four models achieved the competitive result.

Acknowledgements

We would like to thank Takanori Hayashi for the discussions on the Quora Question Pairs[5], which is a related competition held in Kaggle. We also thank anonymous reviewers for their many insightful comments and suggestions.

References

Avrim Blum, Adam Kalai, and John Langford. 1999. Beating the hold-out: bounds for k-fold and progressive cross-validation. In *Proceedings of the twelfth annual conference on Computational learning theory*, COLT '99, pages 203–208, New York, NY, USA. Association for Computing Machinery.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

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

Lee R Dice. 1945. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT sentence embedding.

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

Paul Jaccard. 1912. The distribution of the flora in the alpine zone.1. *New Phytol.*, 11(2):37–50.

Ting Jiang, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Liangjie Zhang, and Qi Zhang. 2022. Prompt-BERT: Improving BERT sentence embeddings with prompts.

Guolin Ke, Qi Meng, Thomas Finely, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30 (NIP 2017)*.

[5]`https://www.kaggle.com/c/quora-question-pairs`

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2021. Pretrained transformers for text ranking: BERT and beyond. *Synthesis Lectures on Human Language Technologies*, 14(4):1–325.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from Tree-Structured long Short-Term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Nattapong Tiyajamorn, Tomoyuki Kajiwara, Yuki Arase, and Makoto Onizuka. 2021. Language-agnostic representation from multilingual sentence encoders for cross-lingual similarity estimation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7764–7774, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bin Wang and C-C Jay Kuo. 2020. SBERT-WK: A sentence embedding method by dissecting BERT-Based word models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2146–2157.

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

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610, Online. Association for Computational Linguistics.

YNU-HPCC at SemEval-2022 Task 8: Transformer-based Ensemble Model for Multilingual News Article Similarity

Zihan Nai, Jin Wang and **Xuejie Zhang**
School of Information Science and Engineering
Yunnan University
Kunming, China
{wangjin,xjzhang}@ynu.edu.cn

Abstract

This paper describes the system submitted by our team (YNU-HPCC) to SemEval-2022 Task 8: Multilingual news article similarity. This task requires participants to develop a system which could evaluate the similarity between multilingual news article pairs. We propose an approach that relies on Transformers to compute the similarity between pairs of news. We tried different models namely BERT, ALBERT, ELECTRA, RoBERTa, M-BERT and Compared their results. At last, we chose M-BERT as our System, which has achieved the best Pearson Correlation Coefficient score of 0.738.

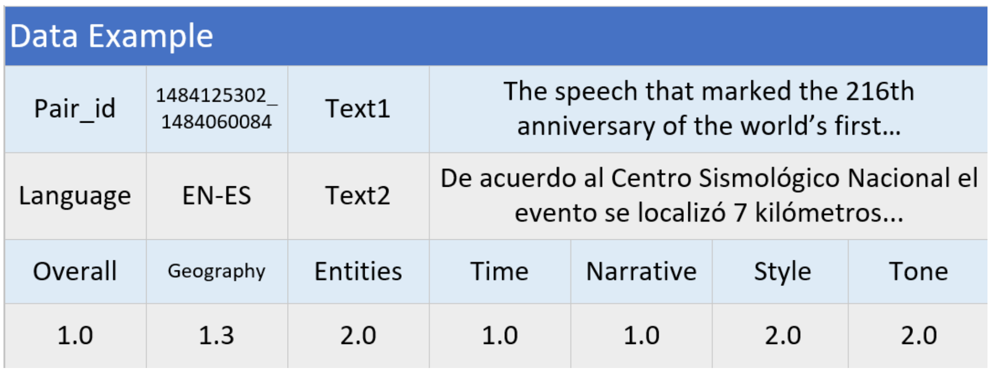
1 Introduction

In the field of Natural Language Processing (NLP), how to measure the similarity of two texts has been a topic of interest among researchers for decades. Text similarity measures play an increasingly important role in text related research and applications in tasks such as information retrieval, text classification, document clustering, topic detection, topic tracking, questions generation, question answering, essay scoring, short answer scoring, machine translation, text summarization and others(Gomaa, Fahmy, et al. 2013). It is widely known that text is a high-dimensional semantic space, hence how to abstractly decompose it so that we can quantify its similarity from a mathematical point of view has become the focus for many researchers. There are three methods to measure text similarity: one is the traditional method based on keyword matching, such as N-gram similarity; the second is to map the text to the vector space, and then use the cosine similarity and other methods; the third is the method of deep learning, such as the deep learning semantic matching model DSSM based on user click data, ConvNet based on convolutional neural network, and the current state-of-art Siamese LSTM and other methods. However, since the introduction of bidirectional encoder representations from transformers (BERT)(Devlin, Chang, Lee, and Toutanova 2018), the accuracy and training efficiency in both text classification and sequence labeling have reached new heights.

SemEval 2022 Task 8 is a multilingual news article similarity task(X. Chen, Zeynali, Camargo, Flöck, Gaffney, Grabowicz, Hale, Jurgens, and Samory 2022). There are mainly 3 difficulties in the task compared with regular text similarity task: (1) the task is interested in the real world-happenings covered in the news articles, not their style of writing, political spin, tone, or any other more subjective design. Therefore, the system built by participant should neglect the subjective part of the text and focus on objective part only;(2) there were over six different languages in both training and test dataset, and some of test data set were composed of multilingual text pair to test the multilingual ability of participating system; (3) the test dataset contains new languages which have never appeared in training data. This situation disguisedly reduces the training data required to train the model.

In this paper, we primarily present a deep learning system for the SemEval-2021 Task 8: Multilingual News Article Similarity. Since there are not any subtasks in the SemEval 2022 Task 8, our system will focus on calculating the overall similarity only. Our approach is based on Transformers, which is a classic NLP model proposed by Google's team in 2017(Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin 2017). We fine-tuned pre-trained masked language models namely BERT, ALBERT, ELECTRA, RoBERTa and M-BERT, which are all based on Transformers. We compared their performance at the task, then picked best of them as our system.

Experimental results show that most of the Transformers based model are valid in the field of text similarity(Mittal and Modi 2021). However, when it comes to multilingual text, there is a clear drop



Data Example						
Pair_id	1484125302_ 1484060084	Text1	The speech that marked the 216th anniversary of the world's first...			
Language	EN-ES	Text2	De acuerdo al Centro Sismológico Nacional el evento se localizó 7 kilómetros...			
Overall	Geography	Entities	Time	Narrative	Style	Tone
1.0	1.3	2.0	1.0	1.0	2.0	2.0

Figure 1: the example of the given training data

in results for all models. Among these models, M-BERT has achieved the best score with the Pearson Correlation Coefficient of 0.738. That is why we chose M-BERT as our system at last. The implementation for our system is made available via Github [1].

The rest of the paper is organized as follow. Section 2 describes the specific requirements of the task and dataset. Section 3 describes the details of the Transformers model used in our system. Section 4 presents the experimental results. Finally, the results and conclusions are presented in Section 4 and 5.

2 Task Overview

The task organizers have provided training and test dataset for the task. The details of this task are given below, and some examples are shown in the Figure 1.

2.1 Problem Description

Given a pair of news articles, are they covering the same news story? Based on the same event, different journalist could write completely different news article, because of their political stance. The kernel of this task is to ignore the subjective part of news article, and calculate the similarity score according to objective facts such as time, geography, entities, etc. A pair of news article will be rate pairwise on a 4-point scale from most to least similar. Systems will be evaluated on their ability to estimate the Overall Similarity between two pairs of news stories, not any of the other scores. The similarity ratings will be compared with the gold standard ratings using Pearson's correlation.

[1]https://github.com/151140043/Sem2022task8.git

2.2 Data Description

The task organizers have provided training and test dataset. The training data consists of 4,964 pairs of news articles, and every pair of news articles have their unique pair_id, the counts of language-pairs is following: en-en: 1800, de-de: 857, de-en: 577, es-es: 570, tr-tr: 465, pl-pl: 349, ar-ar: 274, fr-fr: 72. Apart from the overall score, the score of "Geography", "Entities", "Time", "Narrative", "Style", and "Tone" are also given to contestants. However, they are all for reference only and will not be evaluated as final score. The test data consists of 4954 pairs of news articles. It is worth mentioning that the test dataset is contained more forms of pairs of multilingual news article which are not existed in training data.

3 System Description

We use the transformer based pre-trained model as solution to accomplish the task. As shown in Figure 2, the system we built contains a tokenizer, a model layer, fully connected layer and mean squared error function. The mean squared error is loss function of our system. The model layer represent a Transfomers based pre-trained model, it will be replaced by BERT or any model mentioned above to compare their effect on task 8. The rest of the section will describe the details of every part of the system and their mechanics.

3.1 Tokenizer

Tokenization is essentially splitting a phrase, sentence, paragraph, or an entire text document into smaller units, such as individual words or terms. Each of these smaller units are called tokens. Since the requirement of the task is to evaluate the similarity between two news articles, both the articles will be entered into the tokenizer at the same time. To

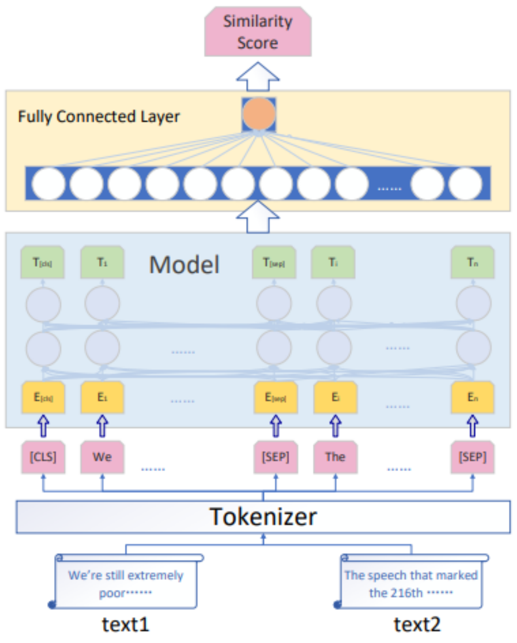


Figure 2: The structure of system

distinguish them, the tokenizer will add a special token named "[SEP]" between them and the last of second article. in addition to "[SEP]", the tokenizer will also add a special token named "[CLS]" at the begin of the first article which is necessary for subsequent step.

3.2 Model

Model layer is an abstraction of a pre-trained Transformer model, it can be a BERT model or any models mentioned below in practical application. In this step, the tokens we got from tokenizer will be passed to the layer, and layer will generate 768-dimensional word embeddings for each word in the news article(Xinge Ma and Zhang 2021) (Most of the models we used generate 768-dimensional word embeddings, a few models generate other parameters, which we will mention later). Then, the model will take the word embeddings of the first token of each article (i.e., '[CLS]') to evaluate the similarity between two news articles, because it integrated the semantic information of the whole sentence. Below we will introduce the specific model we used in the task.

BERT. BERT is a pretrained language representation model, which stands for Bidirectional Encoder Representations from Transformers (Devlin, Chang, Lee, and Toutanova 2018). BERT builds two pre-training tasks, Masked Language Model (MLM) and Next Sentence Prediction (NSP). Unlike traditional left-to-right language model pre-training, BERT is using a MLM pre-training objective, which make BERT to generate deep Bidirectional Linguistic Representation (Devlin J, Chang M W, Lee K, et al. 2018). We used the "bert-base-uncased" in our task. The size of BERT model we use in the task: Layers=12, Hidden Dimension=768, self-attention head =12, Word Piece embedding size =768, Total Parameters=110M.

ALBERT. ALBERT,which stands for A Lite Bert, is proposed to solve the problem that the parameters of the current pre-training model are too large (Lan, M. Chen, Goodman, Gimpel, Sharma, and Soricut 2019). In the classic BERT model, the size of Word Piece embedding (E) is always the same as the hidden layer size(H), i.e., E=H. ALBERT break the binding relationship between E and H, thereby reducing the number of parameters of the model and improving the performance of the model. Another method for ALBERT to reduce the amount of parameters is parameter sharing between layers, which mean, multiple layers could use the same parameters. There are three ways to share parameters: (1) Only share the parameters of the feed-forward network. (2) Only share the parameters of the attention. (3) Share all the parameters. Through these methods, ALBERT could greatly reduce the total parameters. We chose "albert-base-v2" as the model, and the size of model we use in the task: L=12, H=768, A=12, E=128, Total Parameters=12M.

ELECTRA. ELECTRA is a model that share some ideas with BERT, but the main structure is still different. It also can be named as "Efficiently Learning an Encoder that Classifies Token Replacements Accurately" (Clark, Luong, Le, and Manning 2020). The pre-training of ELECTRA can be divided into two parts, which are generator and discriminator. The generator is still MLM, the structure is similar to BERT, but the model will be much smaller than BERT. The output of generator is the input of discriminator. The role of discriminator is to distinguish whether each token input is original or replaced. For each token, the discriminator will perform a binary classification on it, and get the loss. The approach above is called replaced token detection. We chose the "google/electra-base-discriminator" as our model, whose size is: L=12, H=768, A=12, E=768, Total Parameters=110M.

RoBERTa. The full name of RoBERTa is "Ro-

model name	L	H	E	A	P
BERT	12	768	768	12	110M
ALBERT	12	768	128	12	12M
ELECTRA	12	768	768	12	110M
RoBERTa	24	1024	1024	16	355M
M-BERT	12	768	768	12	110M

Table 1: Model structure
L represents L Layers,H represents Hidden Dimension H, E represents WordPiece embedding size E, A represents A self-attention head, P represents Total Parameters

bustly optimized BERT approach" (Liu, Ott, Goyal, Du, Joshi, D. Chen, Levy, Lewis, Zettlemoyer, and Stoyanov 2019). From the perspective of the model, there are not novel innovation in RoBERTa. There are only some adjustment made on the basis of BERT: 1) The training time is longer, the batch size is larger, and the training data is more; 2) The next predict loss is removed; 3) The training sequence is longer; 4) The Masking mechanism is dynamically adjusted. The model we used in the task is "roberta-base", and the architecture of it is: L = 24, H = 1024, E=1024, A = 16, Total Parameters =355M.

M-BERT. The structure of Multilingual-BERT(M-BERT) is exactly the same with the common BERT model. The biggest difference between M-BERT and BERT is that M-BERT is pre-trained on the top 104 languages with the largest Wikipedia using a masked language modeling (MLM) objective. While the common BERT is pre-trained on English Corpus.

3.3 Transformers

Transformers is the base of all the model we mentioned above. Like many neural sequence transduction models, Transformers also have an encoder-decoder structure.

Encoder. The encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a fully connected feed-forward network (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin 2017). The output of the sub-layer can be expressed as:

$$sub\_layer\_output = LayerNorm(x + Sublayer(x))$$

Decoder. The decoder is also composed of a stack of N = 6 identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack.

Multi-Head Attention

Given a set of vector set values, and a vector query, the attention mechanism is a mechanism that computes a weighted sum of values based on the query. In Transformers, they compute attention as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

The multi-head attention allows the model to concatenate different attention results, and it can be represent as:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O$$

$$where \quad head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

3.4 Fully Connected Layer

In the fully connected layer, word embeddings acquired from the previous step will be converted into 1-dimensional numerical values. then, the fully connected layer will output the similarity score depend on the 1-dimensional numerical values.

4 Experiments

4.1 Data Preprocessing

As shown in Figure 2, tokenizer is the data processing structure of our system. After feeding data into tokenizer, it will return a dictionary of 3 lists of ints, which are input_ids, attention_mask and token_type_ids. The input ids are token indices, numerical representations of tokens building the sequences that will be used as input by the model. The attention mask is an optional argument used when batching sequences together. This argument indicates to the model which tokens should be attended to, and which should not. The token_type_ids allow some models to understand where one sequence ends and where another begins. But RoBERTa is an exception, RoBERTa removes NSP, so RoBERTa do not need the token_type_ids as input. We will split ten percent of the training data as validation data to prevent overfitting.

4.2 Evaluation Metrics

Systems will be evaluated on their ability to estimate the Overall Similarity between two pairs of news stories, not any of the other scores. The similarity ratings will be compared with the gold standard ratings using Pearson's correlation.

Epoch	1	2	3	4	5
score	0.681	0.712	0.738	0.721	0.694

Table 2: The relationship between epoch and Pearson coefficient

4.3 Implementation Details

The 5 models that mentioned above are all applicated in the task to evaluate the overall similarity between the given pairs of news articles. The datasets we used were all provided by the competition, with no other external corpus. For all the models, we set learning rate = $5e^{-6}$, epsilon=$1e^{-8}$, loss function as mean squared error, and a batch size of 8 for three epochs.

4.4 Hyper-parameters Fine-tuning

In the experiment, we have tried to change specific parameters while controlling other parameters unchanged, to see if we can get better results. The following will introduce our attempts on fine-tuning parameters.

Loss function. At the beginning of the experiment, we had to choose loss function from mean squared error and cosine similarity. Therefore, we trained an ALBERT model with the loss function of mean squared error based on the news article provided by official, and get the Pearson's correlation of 0.543. Then, we trained another ALBERT model with the loss function of cosine similarity, but only got Pearson's correlation score of 0.055. As a result, we chose mean squared error as loss function.

Batch size. Appropriate batch size is important for the optimization of model. If batch size is too small, the result may be poor. If the batch is too large, it will cause memory overflow. So we chose a batch size of 8.

Epoch. Take M-BERT as an example, the test data provided by task organizers are feed into the model to exam the effect of different epochs. The relationship between epoch and Pearson coefficient score is shown in table 2. It is obvious that the Pearson coefficient score come to the highest when epoch=3. So, we set the epoch equals to 3 in the experiment.

4.5 Comparative Results and Discussion

The results are evaluated by Pearson Correlation Coefficient with the test data provided by official, which is shown in table 3. BERT reach an accuracy of 0.464, ALBERT of 0.543, ELECTRA of 0.474, RoBERTa of 0.475, and M-BERT of 0.738. Our best individual score is 0.738 for M-BERT. As can be seen from the results, BERT, ALBERT, ELECTRA and RoBERTa have similar scores which greater than 0.45 and less than 0.5. M-BERT is the highest among them, whose score is over 0.7. Our results show that M-BERT is able to perform cross-lingual generalization surprisingly well. We believe that the reason why M-BERT outperforms other models is that M-BERT is pre-trained on the Corpus contained 104 languages while other models are pre-trained on a Corpus contained English only. Our conjectures are not groundless. A research (Papadimitriou, Chi, Futrell, and Mahowald 2021) demonstrate that mBERT representations are influenced by high-level grammatical features that are not manifested in any one input sentence, and that this is robust across languages. And mBERT does not encode subjecthood purely syntactically, but that subjecthood embedding is continuous and dependent on semantic and discourse factors, as is proposed in much of the functional linguistics literature. But there is a defect in M-BERT which is while M-BERT's multilingual representation is able to map learned structures onto new vocabularies, it does not seem to learn systematic transformations of those structures to accommodate a target language with different word order (Pires, Schlinger, and Garrette 2019). For example, cross-script transfer is less accurate for pairs like English and Japanese, which have a different order of subject. Therefore, our experiments still have many areas for improvement.

model	Pearson Correlation Coefficient
BERT	0.464
ALBERT	0.543
ELECTRA	0.474
RoBERTa	0.475
M-BERT	0.738

Table 3: Comparable results of experiments

5 Conclusion

In this paper, we described our deep learning models for the multilingual text similarity task SemEval-2022 shared Task 8. The best Pearson's correlation score we got was 0.738. We showed that the Transformer based approaches is valid in the field of multilingual text similarity. However, our system is far from perfect, lots of possible improvement can be implemented in the current model. We would like to further explore how to improve it, and employ more interesting methods in the task.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61966038. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Chen, Xi, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory (2022). “SemEval-2022 Task 8: Multilingual news article similarity”. In: *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Clark, Kevin, Minh-Thang Luong, Quoc V Le, and Christopher D Manning (2020). “Electra: Pre-training text encoders as discriminators rather than generators”. In: *arXiv preprint arXiv:2003.10555*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805*.

Gomaa, Wael H, Aly A Fahmy, et al. (2013). “A survey of text similarity approaches”. In: *international journal of Computer Applications* 68.13, pp. 13–18.

Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2019). “Albert: A lite bert for self-supervised learning of language representations”. In: *arXiv preprint arXiv:1909.11942*.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692*.

Mittal, Abhishek and Ashutosh Modi (2021). “ReCAM@IITK at SemEval-2021 Task 4:BERT and ALBERT based Ensemble for Abstract Word Prediction”. In: *In Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021).*

Papadimitriou, Isabel, Ethan A Chi, Richard Futrell, and Kyle Mahowald (2021). “Deep subjecthood: Higher-order grammatical features in multilingual BERT”. In: *arXiv preprint arXiv:2101.11043*.

Pires, Telmo, Eva Schlinger, and Dan Garrette (2019). “How multilingual is multilingual BERT?” In: *arXiv preprint arXiv:1906.01502*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30.

Xinge Ma, Jin Wang and Xuejie Zhang (2021). “YNU-HPCC at SemEval-2021 Task 11: Using a BERT Model to Extract Contributions from NLP Scholarly Articles”. In: *In Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 478–484,*

BL.Research at SemEval-2022 Task 8: Using various Semantic Information to evaluate document-level Semantic Textual Similarity

Sébastien Dufour†, Mohamed Mehdi Kandi*, Karim Boutamine†, Camille Gosset*, Mokhtar Boumedyen Billami*, Christophe Bortolaso*, Youssef Miloudi†*

†CARL Berger-Levrault, 361 All. des Noisetiers, 69760 Limonest, France

*Berger-Levrault, 64 Rue Jean Rostand, 31670 Labège, France

{sebastien.dufour, youssef.miloudi}@carl.eu, {mohamed.kandi, karim.boutamine, camille.gosset, mb.billami, christophe.bortolaso}@berger-levrault.com

Abstract

This paper presents our system for document-level semantic textual similarity (STS) evaluation at SemEval-2022 Task 8: “Multilingual News Article Similarity”. The semantic information used is obtained by using different semantic models ranging from the extraction of key terms and named entities to the document classification and obtaining similarity from automatic summarization of documents. All these semantic information’s are then used as features to feed a supervised system in order to evaluate the degree of similarity of a pair of documents. We obtained a Pearson correlation score of 0.706 compared to the best score of 0.818 from teams that participated in this task. Our source code can be found at GitHub[1].

1 Introduction

Measuring semantic textual similarity has been a research subject in natural language processing, information retrieval and artificial intelligence for many years. Accurate modelling of textual similarity is fundamental for many applications. Previous efforts have focused on comparing a short text with a long text (e.g., Web search), two sentences or other short text sequences (e.g., paraphrase recognition, image retrieval by captions and Twitter tweets search). There are many other tasks requiring computing the semantic similarity between two long texts (e.g., document classification, document clustering and tracking the similarity of news coverage between different regions).

Semantic textual similarity (STS) measures the level of semantic equivalence between two textual contents. In this paper, we are interested to develop a system that identifies news articles that provide similar semantic information. More specifically, given two news articles, we aim to compute their similarity based on four characteristics: geolocation, time, shared entities, and shared narratives. To achieve this goal, we must identify important elements of the news articles content, such as the discussed event, location, time, and people involved. More precisely, the task 8 of SemEval-2022 edition focuses on the analysis of documents (long texts) that can share different semantic information and can be from different natural languages in both monolingual and cross-lingual settings.

The STS task has been held in SemEval since 2012. More precisely, the semantic similarity systems between sentences using paraphrase datasets have been proposed (Agirre *et al.*, 2012). In the 2013 edition of Joint Conference on Lexical and Computational Semantics – *SEM (Agirre *et al.*, 2013), STS task was focused in using cultural heritage items which are described with metadata such as title, author, or description. Thereafter, systems that compare snippets of text are proposed in the 2016 edition of SemEval (Agirre *et al.*, 2016). Afterwards, the 2017 edition of SemEval proposed to compare sentences in a dimension of monolingual and cross-lingual contents (Cer et al., 2017). In the last few years, many semantic

[1] https://github.com/jln-brtn/BL.Research-at-SemEval-2022-Task-8

similarity datasets and systems have been explored. Among the elements that characterize this new edition of SemEval-2022 compared to 2017 is the fact that we are interested in measuring the similarity between long news contents and not sentences, snippets of texts or short texts.

We encountered many challenges: First, the content scraping of the provided URL (in the first attempt, we had empty or incomplete content in many examples). Then, we have many languages (new languages in the test set are not present in the train set). Finally, it was challenging to choose the type of the predicted score: decimal or integer, which is different according to the number of human annotators per document.

In this paper, we present our approach for SemEval-2022 Task 8. The system that we propose is based on the computation of different scores by document pairs. These scores are used as features to train a supervised system. Regarding the multiple languages, we have developed a dedicated model for some of them, and translations in a pivot language for others.

The paper is organized as follows: we describe the problem and the data in section 2. Then, we give an overview of the related work in section 3. Next, we present the proposed system in section 4. We detail the performed experiments and the results in sections 5 and 6. Finally, we conclude the paper in section 7.

2 Background

2.1 Problem Description

The objective of this task at SemEval-2022 is to compare the similarity between two news articles and to be as consistent as possible with manual annotations provided by native annotators. We identified several challenges and issues in this task:

- **Scraping of the data**: probably not foreseen, because the final access to a clean and complete text was not always possible. For example, scraping sometimes empty, incomplete, or not correct (name of the newspaper, badly identified characters, and others).
- **Multilingual aspect of this task**: several languages used, some of which difficult or poorly modeled like Russian, Chinese, Turkish, and also text pair analysis where text content are provided by different languages.
- **Similarity evaluation of texts**: on a precise topic basis, on concordant geographical or temporal elements, on a similarity of tone and common style.

2.2 Data Description

Data used to train consisted of a total of 4,964 pairs of news articles written in seven different languages, namely: English (EN), French (FR), Spanish (SP), German (DE), Polish (PL), Arabic (AR) and Turkish (TR). The pairs were formed either with the same language or with different languages. In the training corpus, seven couple types are monolingual and only one is cross-lingual (DE_EN). For the test corpus, we have a total of 4,953 pairs of news articles and ten different languages with three new languages, namely: Russian (RU), Chinese (ZH) and Italian (IT). In this corpus, we have eight cross-lingual couple types with seven couple types that never seen in train corpus. Table 1 describes the number of document pairs for each corpus and each language pair.

Couple Type	Monolingual			Couple Type	Cross-lingual	
	Train	Test			Train	Test
EN_EN	**1,800**	236		DE_EN	577	190
FR_FR	72	111		SP_EN	—	498
SP_SP	570	243		PL_EN	—	64
DE_DE	857	611		ZH_EN	—	223
PL_PL	349	224		SP_IT	—	320
AR_AR	274	298		DE_FR	—	116
TR_TR	465	275		DE_PL	—	35
RU_RU	—	287		FR_PL	—	11
ZH_ZH	—	**769**				
IT_IT	—	442				

Table 1: statistics on the number of examples of train and test corpus.

Each pair of documents was annotated by one to eight annotators based on seven score categories that are “Geography”, “Entities”, “Time”, “Narrative”, “Overall”, “Style”, and “Tone”. For each category of each pair, a score on a 4-point scale was given by the available annotators then the average resulted in floats or integers. The score ranges from 1 as most similar to 4 as least similar. In addition to the mentioned scoring categories, the article pair identifiers, languages, and URLs were given for each pair. Using the URLs, we were able to retrieve data from the articles such as: titles, texts, keywords, tags, authors, publication date, abstracts, and meta-descriptions along with other irrelevant information. Finally, evaluation data were released later by organizers in the same format as train data except for the previous cited scoring categories as our system were judged based on one of those scores that is the “Overall score”.

3 Related Work

Semantic textual similarity deals with determining how similar two pieces of texts are. This can be done by assigning a rating from 1 to 4 (or 1 to 5) for more similar to less similar content pairs. Related tasks are paraphrasing or duplication identification. There are many interesting works for STS, an Evaluation Toolkit for Universal Sentence Representations, named SentEval has been proposed (Conneau and Kiela, 2018). It includes 17 downstream tasks[2], including common STS tasks from 2012-2016. We describe in this section some existing systems from the previous editions of SemEval.

Tian *et al.* (2017) proposed three feature-engineered models using Random Forest, Gradient Boosting, and XGBoost regression methods. Their features are based on n-gram overlap; edit distance, longest common prefix/suffix/substring, tree kernels, word alignments, to cite a few. They also propose four deep learning methods. The difference between the methods is the approach to sentence embeddings using either: averaged word embeddings, projected word embeddings, a deep averaging network, or LSTM (Long-Short Term Memory) (Hochreiter and Schmidhuber, 1997). To build the global model, they average scores of the deep learning and the feature-engineered models.

Wu *et al.* (2017) use sentence information content with WordNet (Wallace, 2007) and BNC word frequencies (Leech, Rayson and Wilson, 2001). One variant uses sentence information content exclusively. Another variant uses ensembles information content with Sultan, Bethard and Sumner (2015)’s alignment method. The third variant uses ensembles information content with a cosine similarity of summed word embeddings with an IDF (Inverse Document Frequency) weighting scheme (Jones, 1972).

Shao (2017) proposed a convolutional Deep Structured Semantic Model for the generation of sentence embeddings. The embeddings are compared using cosine similarity and element-wise difference with the resulting values fed to additional layers. This architecture is similar to Tian *et al.* (2017)’s deep learning models.

Henderson *et al.* (2017) proposed a feature engineering approach that they complete with deep learning. Ensembled components include alignment similarity; string similarity measures such as matching n-grams, summarization, Machine Translation (MT) metrics, an RNN (Recurrent Neural Networks), and RCNN (Recurrent Convolutional Neural Networks) over word alignments, and a BiLSTM (Bidirectional Long-Short Term Memory) networks.

4 System Overview

Some strategies can be considered to resolve the task 8 of SemEval-2022 :

- Building a model for each training dataset by language: English, French, Spanish, German, Polish, Arabic and Turkish.
- Building a unique model in English[3] and translating all texts into that language.
- Building a multilingual model that can handle two texts into different languages.

The advantage of the first and the third strategies is that they avoid translation to a pivot language. The second strategy has the advantage of enabling consistent training and being able to handle all languages using a single model. Our system is neither based on the translation of all texts in English, nor on the construction of a multilingual model. We have chosen to build learning models in some main languages, namely: English, French, Spanish, German, Arabic and Turkish. We abandoned Polish language because we did not find adequate pretrained models. When two texts to be compared are not in the same language, they are translated into the main language selected.

A fundamental point for the final score to be generated is the choice on the precision of the answer in terms of decimals. We have noticed that for some languages, the Overall score obtained is

[2]http://nlpprogress.com/english/semantic_textual_similarity.html

[3] The global reference language and therefore the one best managed by all techniques in natural language processing.

an integer. In English, for example, there is a decimal score and more annotators. The actual number of annotations cannot be determined in advance. We have therefore defined rules to complete our evaluation according to the score obtained and the language. We did not use the metadata of the news articles (authors, dates, newspaper, tags, etc.) because these data were incomplete. We simply retained both titles and text contents. We also noticed:

- Strong correlations between the Overall score and the scores of Entities, Narrative, and Geography. Table 2 describes the correlations between annotator scores.
- The training dataset is unbalanced, especially in English. We have more than four scores. The figure 1 describes more information about this dataset.

	Geography	Entities	Time	Narrative	Overall	Style	Tone
Geography	1	**0.63**	0.12	0.52	0.59	0.33	0.35
Entities		1	0.25	**0.74**	**0.80**	0.32	0.39
Time			1	0.4	0.43	0.10	0.18
Narrative				1	**0.88**	0.32	0.45
Overall					1	0.33	0.45
Style						1	0.57
Tone							1

Table 2: Correlations between annotator scores.

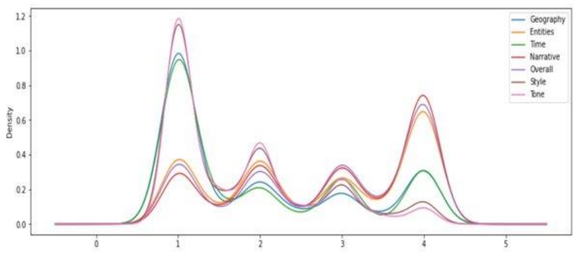


Figure 1 : Density of annotator scores.

Following the observations that are seen in table 2 and figure 1, we implemented a scoring system based on each language who will be features to feed a supervised system. We detail below our features:

1. A similarity score of titles based on sentence transformers with a pretrained encoder model (Reimers and Gurevych, 2019).
2. We used text summarization based on pretrained transformers in each language to measure a similarity score between these summaries. We tested several models on some examples and selected those which seemed to be the best. Based on the language, we were able to obtain one or two models of summaries and thus one or two scores. We have mainly turned to BART (Lewis *et al.*, 2019) or Sequence-to-Sequence models (Chen *et al.*, 2021; Zhang *et al.*, 2019; Shleifer and Rush, 2020; Eddine, Tixier and Vazirgiannis, 2021). The models are pretrained on different summarization corpus variants, for example MLSUM, the Multilingual Summarization Corpus (Scialom *et al.*, 2020).
3. We used identification and extraction of keywords/'key terms' in titles and content texts by using the PKE toolkit library (Boudin, 2016). The extracted tags are nouns, proper nouns, verbs, and adjectives only with stemming adding the ten semantically closest words by using Word2Vec (Mikolov *et al.*, 2013) with Gensim library (Řehůřek and Sojka, 2010), if a model exists in a specific language. We compute the number of common terms in both texts.
4. We used identification and extraction of common named entities between titles and content texts, namely: places, persons, organizations, and dates. We used two libraries: Spacy (Honnibal *et al.*, 2020) and Stanza (Qi *et al.*, 2020), and pretrained transformers models. As for keywords and key terms, we compute the number of identical entities in both texts.
5. For the various geographical detected entities (cities, regions, countries), we calculate a score for the proximity between places based on geocoding[4].
6. We used the zero-shot classification models (Yin, Hay and Roth, 2019) based on Press topics that we have defined manually: *politics*, *sport*, *health*, *economy* and *technology*. The similarity score obtained reflects the number of common topics between two texts.

[4] https://geopy.readthedocs.io/en/stable/#nominatim

7. Finally, sentiment analysis models are used to identify if the sentiment polarity is positive, negative, or neutral in both texts. The models used are proposed by Wolf *et al.* (2019), Guhr *et al.* (2020), Demange (2021) and Pérez, Giudici and Luque (2021). The similarity score obtained thus reflects the number of common points.

With all these features, it is possible to make a final rating based on the classification or regression techniques. For the classification, we have tested Random Forest classifier and Logistic Regression for the algorithms that achieve the best performance. For the regression, we have tested Linear Regression, Partial Least Squares (PLS) and Extra Trees Regressor. That said, we can obtain a final evaluation of the selected strategy. To optimize the Pearson correlation score (the measure chosen by the organizers for the evaluation), we use PyCaret library (Moez, 2020) to compare all possible models (by using cross-validation with 10 folds).

For our English model, the various elements found allowed to have a good performance quickly (approximately 0.85 of Pearson correlation). The only concern was the strong imbalance of the training dataset that we needed to rebalance. The French model had a poor training dataset (only 72 pair examples). Thus, we selected a more efficient NER (Named Entity Recognition) Transformer model [5] than Spacy; and supplemented and balanced it to 200 pair examples with Spanish train texts to obtain a correct performance. We also focused on optimising the Turkish model in the same way with an efficient NER transformer[6]. We have not worked to optimize the German model which ought to have been much better.

5 Experimental Setup

We applied our scoring models to every pair of titles and content texts. For Polish language and the new languages observed in the evaluation dataset like Chinese, Russian and Italian, all texts were translated into English with deep Translator library[7] (Google Translate model) and then applied the English model. When there are two different languages, they are translated into English (for DE_EN, SP_EN, PL_EN and ZH_EN pairs), French (for FR_PL pairs), Spanish (for SP_IT pairs) or German (for DE_PL and DE_FR pairs). Finally, a compromise strategy was used between the score obtained in classification (integers) and in regression (value between 1 and 4 with selected rounding). Table 3 describes the confusion matrix after a test on a subset of train corpus.

Annotation score	1	2	3	4
1	**136**	27	4	0
2	50	**84**	35	16
3	6	42	**53**	89
4	0	20	33	**363**

Table 3: Example of Confusion Matrix obtained in English (test on the training dataset) after a Classification Random Forest Model.

6 Results and Analysis

Our final Pearson correlation on the test corpus is 0.706. A closer look at the results shows inconsistent performance scores for different languages. Table 4 shows the obtained results on the test corpus for each language pair. There are very good results for English and French (0.82 as Pearson correlation), good for Spanish (0.75), rather interesting for Turkish (0.74), disappointing for German (0.60, poorly optimized and badly managed model), and weak for Arabic (0.64). The final result for a specific language is generally consistent with the evaluations made previously on the subset (cross-validation) of training dataset.

For the translation part, we observe a fast drift according to the languages: the ZH_EN examples translated into English remain very accurate (0.79) but the ZH_ZH examples only obtain an average score of 0.69. We thus have a significant performance reduction in IT_IT (0.74) and SP_IT (0.61) compared to cases when English is used.

We found that most of the large deviations in evaluation (45 greater than 2 and 98 greater than 1.5) were related to scraping errors (blank or inconsistent text) where 2 and 1.5 are "Overall scores". This derivation also resulted in an evaluation of 3 to 4 on our part for an actual close to 1. We did not find a reverse case where our score was close to 1 and the actual close to 4.

[5] https://huggingface.co/Jean-Baptiste/camembert-ner
[6] https://huggingface.co/savasy/bert-base-turkish-ner-cased
[7] https://deep-translator.readthedocs.io/en/latest/

We observe in conclusion that 71% scores of pairs were excellent (below 0.1), 84% were good (below 0.5) and 96% below 1. We believe that our English, French, Turkish and Spanish models are correct and could have been further optimized cleanly. We had technical difficulties in making a correct Arabic model. As for the German model, we did not work on it enough and it should have obtained a performance close to 0.80.

Language pair	Pearson correlation
EN_EN	**0.82**
DE_DE	0.60
SP_SP	0.75
PL_PL	0.55
TR_TR	0.74
AR_AR	0.64
RU_RU	0.67
ZH_ZH	0.69
FR_FR	**0.82**
DE_EN	0.74
SP_EN	0.79
IT_IT	0.74
PL_EN	0.73
ZH_EN	0.79
SP_IT	0.61
DE_FR	0.61
DE_PL	0.4
FR_PL	0.74
Global	**0.706**

Table 4: System results on the test corpus.

7 Conclusion

In this paper, we described our supervised semantic textual similarity system developed for the SemEval-2022 task 8 and the result of the corresponding run we submitted. Our system uses different features reflecting the similarity that can be obtained, for example, between shared key terms and named entities, or even topics by using zero-shot learning for text classification systems. In addition, we use geolocation for location entities and measure the semantic similarity through the use of lexical embeddings at the sentence level (text title) and paragraph level (text summarizer obtained automatically by using transformers).

Beyond the use of a supervised system to measure the degree of similarity between two given texts, we are in a context of documents that can come from different languages by processing both pairs of monolingual documents and pairs of cross-lingual documents. Since the test corpus may contain documents written in natural languages not processed during learning phase, our system is able to perform an automatic translation into a pivot language in order to project new documents into already known spaces.

Acknowledgments

We would like to thank Berger-Levrault for giving us the opportunity to participate in the 2022 edition of SemEval. Our thanks also go to the research and technological innovation department (DRIT) which is an important part of Berger-Levrault.

References

Agirre, E. et al. (2012) ‘SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity’, in *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation.* USA: Association for Computational Linguistics (SemEval ’12), pp. 385–393. Available at: https://aclanthology.org/S12-1051.pdf.

Agirre, E. et al. (2013) ‘*SEM 2013 shared task: Semantic Textual Similarity’, in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity.* Atlanta, Georgia, USA: Association for Computational Linguistics, pp. 32–43. Available at: https://aclanthology.org/S13-1004.

Agirre, E. et al. (2016) ‘SemEval-2016 Task 1: Semantic Textual Similarity, Monolingual and Cross-Lingual Evaluation’, in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016).* San Diego, California: Association for Computational Linguistics, pp. 497–511. doi:10.18653/v1/S16-1081.

Boudin, F. (2016) ‘pke: an open-source python-based keyphrase extraction toolkit’, in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations.* Osaka, Japan, pp. 69–73. Available at: http://aclweb.org/anthology/C16-2015.

Cer, D. et al. (2017) ‘SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation’, in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017).* Vancouver, Canada: *Association* *for Computational Linguistics*, pp. 1–14. doi:10.18653/v1/S17-2001.

Chen, C. et al. (2021) 'bert2BERT: Towards Reusable Pretrained Language Models', arXiv:2110.07143 [cs] [Preprint]. Available at: http://arxiv.org/abs/2110.07143 (Accessed: 25 February 2022).

Conneau, A. and Kiela, D. (2018) 'SentEval: An Evaluation Toolkit for Universal Sentence Representations', *CoRR*, abs/1803.05449. Available at: http://arxiv.org/abs/1803.05449.

Demange, J. (2021) 'Four sentiments with FlauBERT', Hugging Face repository: https://huggingface.co/DemangeJeremy/4-sentiments-with-flaubert. [Preprint].

Eddine, M.K., Tixier, A.J.-P. and Vazirgiannis, M. (2021) 'BARThez: a Skilled Pretrained French Sequence-to-Sequence Model', arXiv:2010.12321 [cs] [Preprint]. Available at: http://arxiv.org/abs/2010.12321 (Accessed: 25 February 2022).

Guhr, O. et al. (2020) 'Training a Broad-Coverage German Sentiment Classification Model for Dialog Systems', in *Proceedings of The 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, pp. 1620–1625. Available at: https://www.aclweb.org/anthology/2020.lrec-1.202.

Henderson, J. et al. (2017) 'MITRE at SemEval-2017 Task 1: Simple semantic similarity', in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 185–190.

Hochreiter, S. and Schmidhuber, J. (1997) 'Long Short-Term Memory', Neural Computation, 9(8), pp. 1735–1780. doi:10.1162/neco.1997.9.8.1735.

Honnibal, M. et al. (2020) 'spaCy: Industrial-strength Natural Language Processing in Python', Zenodo [Preprint]. doi:10.5281/zenodo.1212303.

Jones, K.S. (1972) 'A statistical interpretation of term specificity and its application in retrieval', Journal of Documentation, 28(1), pp. 11–21. doi:10.1108/eb026526.

Leech, G., Rayson, P. and Wilson, A. (2001) Word Frequencies in Written and Spoken English based on the British National Corpus.

Lewis, M. et al. (2019) 'BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension', arXiv:1910.13461 [cs, stat] [Preprint]. Available at: http://arxiv.org/abs/1910.13461 (Accessed: 25 February 2022).

Mikolov, T. et al. (2013) 'Efficient Estimation of Word Representations in Vector Space', in *Proceedings of the International Conference on Learning Representations (ICLR)*, pp. 1–12.

Moez, A. (2020) PyCaret: An open source, low-code machine learning library in Python. Available at: https://www.pycaret.org.

Pérez, J.M., Giudici, J.C. and Luque, F. (2021) 'pysentimiento: A Python Toolkit for Sentiment Analysis and Social NLP tasks', arXiv:2106.09462 [cs] [Preprint]. Available at: http://arxiv.org/abs/2106.09462 (Accessed: 25 February 2022).

Qi, P. et al. (2020) 'Stanza: A Python Natural Language Processing Toolkit for Many Human Languages', CoRR, abs/2003.07082. Available at: https://arxiv.org/abs/2003.07082.

Řehůřek, R. and Sojka, P. (2010) Software Framework for Topic Modelling with Large Corpora. University of Malta. Available at: https://is.muni.cz/publication/884893/en/Software-Framework-for-Topic-Modelling-with-Large-Corpora/Rehurek-Sojka (Accessed: 24 February 2022).

Reimers, N. and Gurevych, I. (2019) 'Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks', in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics.* Available at: https://arxiv.org/abs/1908.10084.

Scialom, T. et al. (2020) 'MLSUM: The Multilingual Summarization Corpus'. In the Computing Research Repository (CoRR). Available at https://arxiv.org/abs/2004.14900.

Shao, Y. (2017) 'Hcti at semeval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity', in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 130–133.

Shleifer, S. and Rush, A.M. (2020) 'Pre-trained Summarization Distillation', CoRR, abs/2010.13002. Available at: https://arxiv.org/abs/2010.13002.

Sultan, M.A., Bethard, S. and Sumner, T. (2015) 'Dls@ cu: Sentence similarity from word alignment and semantic vector composition', in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 148–153.

Tian, J. et al. (2017) 'ECNU at SemEval-2017 Task 1: Leverage Kernel-based Traditional NLP features and Neural Networks to Build a Universal Model for Multilingual and Cross-lingual Semantic Textual Similarity', in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, pp. 191–197.

Wallace, M. (2007) Jawbone Java WordNet API. Available at: https://sites.google.com/site/mfwallace/jawbone.

Wolf, T. et al. (2019) 'HuggingFace's Transformers: State-of-the-art Natural Language Processing', ArXiv, abs/1910.03771.

Wu, H. et al. (2017) 'BIT at SemEval-2017 Task 1: Using semantic information space to evaluate semantic textual similarity', in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 77–84.

Yin, W., Hay, J. and Roth, D. (2019) 'Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach', CoRR, abs/1909.00161. Available at: http://arxiv.org/abs/1909.00161.

Zhang, J. et al. (2019) 'PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization'. In the Computing Research Repository (CoRR). Available at https://arxiv.org/abs/1912.08777.

DataScience-Polimi at SemEval-2022 Task 8: Stacking Language Models to Predict News Article Similarity

Marco Di Giovanni[*] [†] **Thomas Tasca**[*] [‡] **Marco Brambilla**[†]
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy
[*]These authors contributed equally to this work.
[†] {marco.digiovanni, marco.brambilla}@polimi.it [‡] thomas.tasca@mail.polimi.it

Abstract

In this paper, we describe the approach we designed to solve SemEval-2022 Task 8: Multilingual News Article Similarity. We collect and use exclusively textual features (title, description and body) of articles. Our best model is a stacking of 14 Transformer-based Language models fine-tuned on single or multiple fields, using data in the original language or translated to English. It placed fourth on the original leaderboard, sixth on the complete official one and fourth on the English-subset official one. We observe the data collection as our principal source of error due to a relevant fraction of missing or wrong fields.

1 Introduction

SemEval-2022 task 8 (Chen et al., 2022) (Multilingual News Article Similarity) is a document-level similarity task on news articles data. The goal is to predict whether two multilingual news articles cover the same real-world happening regardless of their writing style, political spin and tone. The task included resources written in 10 different languages: English, German, Spanish, Turkish, Polish, Arabic, French, Chinese, Italian and Russian (the training dataset included news written only in the first seven languages). This task is interesting as we can apply the obtained approaches to cluster news articles and track the similarity of news coverage between different outlets or regions as done in the Agenda Setting project[1].

Our best model is a simple but effective Stacking of a set of Language Models trained on different combinations of textual features. We fine-tune 14 Language Models, half of them with original multilingual texts and half with texts translated to English. We select three textual fields from the features extracted by our scraper (title, body and description of the news article), and we fine-tune a model for every combination.

Our model achieved a maximum Pearson correlation score of 0.790 and scored 4th on the leaderboard that considers the best test result for each team. However, the final ranking, based on a bootstrapping approach across teams' submissions to estimate the expected rank, penalizes us to 6th place since it assumes that submissions are an exploration of the hyperparameter/model configuration space of the system. The assumption, released after the end of the competition, does not hold for our team.

Our model mainly struggles with missing or wrong data since the training and evaluation datasets were released as links to scrape due to privacy policies. We noticed that we could not collect parts of the datasets, and some of the collected data were clearly wrong (e.g. "Get in touch with us. All rights reserved" as the body of an article).

The code will be available on GitHub[2].

2 Background

2.1 Task Setup

The organizers of the competitions provided two datasets: the training dataset and the test dataset. The training dataset is a collection of 4964 pairs of links to news articles with gold labels: real numbers ranging from 1 to 4, where 4 represents completely different articles. The test dataset is a collection of 4953 pairs of links to news articles without gold labels. Both datasets included the languages of the original articles so that we do not need to infer them from the data. Both datasets also included duplicated rows that we discarded. Due to copyright problems, the datasets did not directly contain the contents of the articles but only the links (original link and the Internet Archive version) to scrape them with a public script. Figure 1 shows the distribution of languages is in the two datasets. The test data contains news written in languages never

[1]`http://www.euagendas.org/`

[2]`https://github.com/DataSciencePolimi/MultilingualNewsArticleSimilarity`

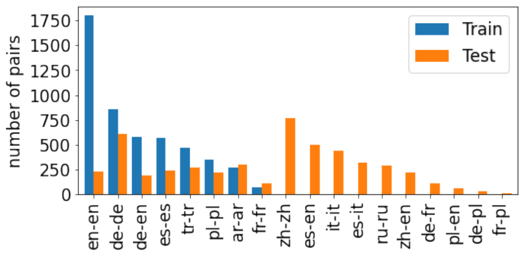


Figure 1: Distribution of language pairs in both datasets.

used in the training dataset (Chinese, Italian and Russian). Moreover, only 577 training pairs of news articles are in different languages (English and German). The performance of the models was computed using the Pearson Correlation Coefficient.

2.2 Related Work

We relied on transformers-based models and, in particular, we leverage the SentenceTransformers (Reimers and Gurevych, 2019) framework, which learns meaningful sentence Embeddings using Transformer-based Language Models (Vaswani et al., 2017). We build our solutions on top of the best pre-trained models provided and suggested by SentenceTransformers. Both models are based on Microsoft MPNet (Song et al., 2020), a pre-training approach that inherits the advantages of BERT's Masked Language Modeling (Devlin et al., 2019) and XLNet's Permuted Language Modeling (Yang et al., 2020) and avoids their limitations, providing better performances.

3 System overview

3.1 Data retrieval

To facilitate re-hydrating the textual content of the news articles, organizers provided a script[3] that downloads the earliest available version of each one of them from the Internet Archive and, only in case of problems, attempt to download them from the original site of publication using newspaper3k. For each article, we obtain the HTML content of the page and a JSON file containing additional information extracted from the page. We select the article's title, body and a brief description as features to feed our LMs.

In populating the dataset, we encountered two main challenges:

- We could not download the complete training dataset due to sites inaccessibility issues or anti-scraping systems. We did not encounter this issue on news articles from the Test dataset;

- The JSON files contains missing and noisy data. We believe that empty, unusable, or obviously incorrect fields are due to the low robustness of newspaper3k when applied to non-standard news websites.

While the first issue is hard to solve *a-posteriori* and could have been tackled by downloading the data as soon as the links were released, we improved the quality of the obtained dataset with Trafilatura[4] (Barbaresi, 2021), an alternative to newspaper3k. Trafilatura is an accurate web scraping tool for text discovery and retrieval that allows to prioritize the precision of the collection, i.e. yielding less but cleaner data.

3.2 Key algorithms

3.2.1 Single-field Language Models

As baseline models, we fine-tuned pre-trained Transformer-Encoder Language Model on a single field. We initialize the LMs with the pre-trained models suggested by SentenceTransformers (Reimers and Gurevych, 2019): a version of MPNet fine-tuned with self-supervised contrastive learning objective over a 1B sentences pairs obtained concatenating more than 20 datasets and trained using a self-supervised contrastive learning objective[5], and a similar multilingual alternative [6]. Every model was downloaded from Huggingface (Wolf et al., 2020).

The selected models are trained to generate, from variable-length input texts, fixed-size dense embeddings that encode semantic similarity: similar documents are mapped to vectors close to each other. We minimize the MSE loss between the cosine similarity of the embeddings obtained from the LMs and the rescaled labels (details about how and why we rescale the original labels in Section 5.1).

We trained the Single-field LMs on the three selected fields independently: Title (T), Description

[3]https://github.com/euagendas/semeval_8_2022_ia_downloader

[4]https://github.com/adbar/trafilatura

[5]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[6]https://huggingface.co/sentence-transformers/paraphrase-mpnet-base-v2

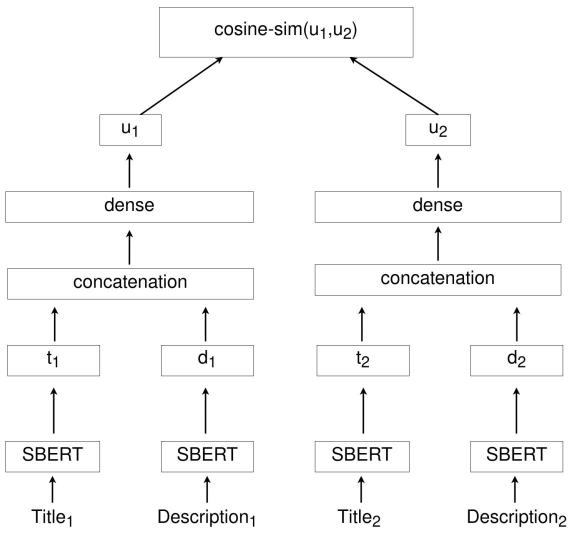


Figure 2: Schema of the two-fields (*title* and *description*) Language Model

(D) and Body (B). We obtain 6 different Single-field LMs, three of them fine-tuned on the original multilingual fields and three on the fields translated to English.

3.2.2 Multiple-fields Language Models

To improve the Single-Field LMs, we design an architecture that can handle multiple textual fields to generate semantically informative dense embeddings. Figure 2 shows the schema of such a model. Firstly, we select two fields (the alternative with the three fields is straightforward), and we feed them into a pre-trained Sentence-LM to generate two dense fixed-size embeddings for each news article (t_i and d_i). Then we concatenate the obtained embeddings, and we feed the result to a single learnable dense layer, initialized with Xavier (Glorot and Bengio, 2010), to generate a 768-dimensional global vector that includes information of both fields (u_i).

We train the model using every combination of two fields (TD, TB and DB) and also using all of them (TDB). We obtain 8 fine-tuned models, four of them fine-tuned on the original multilingual fields and four on the fields translated to English.

As for Single-field LMs, we minimize the MSE loss between the cosine similarity of the embeddings obtained from the LMs and the rescaled labels.

3.2.3 Stacking

Since ensemble learning has proven to be effective in competitions, we perform Stacking (Wolpert, 1992) on sets of the previously described models. We tested as final estimators simple regressors (implemented in scikit-learn (Pedregosa et al., 2011)) such as linear regressors regularized with Lasso (Tibshirani, 1996) or Ridge (Hoerl and Kennard, 2000), ElasticNet (Ela), Support Vector Machines (Cortes and Vapnik, 1995), KNN (Altman, 1992) and shallow Multi-layer Perceptrons (Hastie et al., 2001). Our best model is a MLP with 3 layers, 10 units per layer, trained with Adam (Kingma and Ba, 2015) and learning rate 3×10^{-2}, obtained using 5-fold cross-validation on the validation set.

4 Experimental setup

4.1 Data splitting, cleaning and preprocessing

We partitioned the available training data into two portions using an 80/20 training/test split stratified for language pairs and score (we approximate the labels to their integer part during this step).

We clean our dataset with the following preprocessing approaches:

- We remove duplicated rows. Some of the duplicates pairs obtained different similarity scores, so we replaced them with their mean;
- We detect the right character encoding with cChardet[7], a library to automatically detect the character encodings. Most of the files encoded with rare encodings were Turkish or Arabic news articles (encoded with legacy standards Windows-1254 and Windows-1256 respectively). When cChardet fails to detect the right encoding, we scrape the original website with Trafilatura;
- Some news articles had missing fields. We replace missing descriptions with bodies and missing titles with descriptions.

4.2 Translation

We fine-tuned our models on original data or translated data. Fine-tuning on original data requires a robust multi-lingual model that can compute semantic similarities regardless of the language of the documents. Fine-tuning on data translated to English requires an accurate Neural Machine Translation (NMT) model. We believe that Stacking both alternatives improves the overall performance since the final prediction relies on the strengths of both.

[7]https://github.com/PyYoshi/cChardet

We translated the whole dataset using EasyNMT[8] , a container of different NMT models. In particular, we used the Opus-MT (Tiedemann and Thottingal, 2020) model provided by the Helsinki NLP group.

4.3 Hyperparameters

Every LM was fine-tuned using the default optimizer (AdamW) with learning rate 2×10^{-5} and weight decay 0.01, using 10% of train iterations as warm-up. We trained the models for 3 epochs on our Training set, and we select the best model as the one that maximizes the Pearson Correlation Coefficient on the Validation set. Every experiment was performed using Colab (Bisong, 2019).

4.4 Evaluation measure

We evaluate our models using Pearson Correlation Coefficient of the cosine similarity between the embeddings generated by the models and the similarity manually scored by annotators. The scores ranges from 1 (perfect positive correlation) to -1 (perfect negative correlation), where 0 represents uncorrelated data.

5 Results and Ablation

Table 1 shows the results of our models. We obtain the best performing model on the Test set by Stacking 3 carefully selected Single-field Models fine-tuned respectively on translated titles, translated bodies and original bodies. The final estimator is a MLP with 3 layers and 10 units per layer as described above.

However, we obtained our best *submitted* prediction using our best performing model on the Validation set: a Stacking of all Single-field and Multiple-fields LMs, with the same final estimator as before. This model allowed us to score fourth in the original leaderboard that includes the best performing model for each team, and sixth on the official ranking[9].

Name	Field	Dev	Test
MPNet	T_t	75.1	63.7
M_MPNet	T	74.7	65.1
MPNet	D_t	69.9	57.2
M_MPNet	D	67.6	56.9
MPNet	B_{t512}	81.7	77.6
M_MPNet	B_{512}	78.2	73.4
MPNet	T_t,D_t	77.4	66.7
M_MPNet	T,D	76.7	67.9
MPNet	T_t,B_t	81.5	75.8
M_MPNet	T,B	80.1	74.2
MPNet	D_t,B_t	80.9	74.5
M_MPNet	D,B	77.7	71.7
MPNet	T_t,D_t,B_t	81.9	75.1
M_MPNet	T, D, B	79.7	73.6
Ridge	B_{t512},T_t	82.13†	78.07
SVR	B_{t512},B_{512}, T_t	82.99†	78.71
MLP	A	**83.7**†	79.0
MLP*	B_{t512},B_{512}, T_t	83.5†	**79.2**
Winner model	/	/	*81.8*

Table 1: Pearson's $r \times 100$ on Validation and Test Datasets. We indicate with T the title, D the description and B the body of the articles. A stands for title, description and body, both original and translated, both obtained with Single-field and Multiple-fields approaches. The subscript t indicates translated texts. Values marked with † refer to the models evaluated with 5-fold crossvalidation on the validation dataset. All fields are truncated at 256 tokens except when specified with a subscript. The model marked with * was not submitted before the end of the competition.

The first part of Table 1 reports the results of the six Single-field LMs. As expected, the model fine-tuned on the body (B) of the news articles, the longest, thus most informative field, obtained the best results, both in the Validation and Test set. Models fine-tuned on the description (D) perform worse, probably due to the higher noise of the field (i.e., sometimes the content was missing or contained general information about the journal or the website). English models (reported as MPNet to highlight their initialization model) trained on translated data generally perform better than multilingual models (reported as M_MPNet) trained on original data.

The second part of the table reports the results of the six Double-fields LMs and the two Triple-

[8]https://github.com/UKPLab/EasyNMT

[9]The final ranking was computed using a bootstrapping approach across teams' submissions. This approach assumes that multiple submissions by the same team represent an exploration of the hyper-parameter/model configuration space. This approach allows the organizers to estimate the general performance of the proposed models. However, the organizers announced this evaluation procedure after the end of the competition. We remark that the assumption does not hold for our team, and penalizes us on the final leaderboard. To get an idea of the overall test performance, several submissions from our team, especially at the beginning of the challenge, were generated with simple (sometimes even not fine-tuned) models. Moreover, due to technical problems of the challenge, we submitted opposite predictions from the same model to be sure that at least one scored correctly.

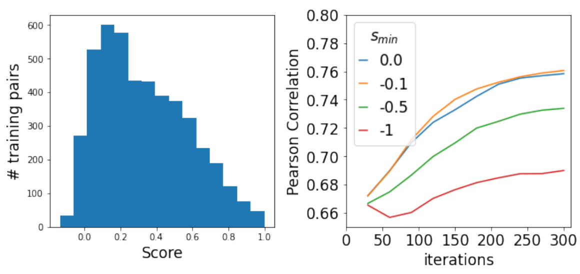


Figure 3: (*left*): Distribution of scores of pairs of titles from training set. (*right*): Pearson correlation during training for different values of s_{min}.

fields LMs. Combinations and interactions of fields give performances worse than our best performing Single-field LMs (MPNet on Body). We believe that this degradation of performance is due to two main factors. First, the noisiness of the title and description, as a higher percentage was missing or wrong. Second, due to memory issues, we had to perform Multiple-field LMs training reducing the maximum length of the body to 256 tokens instead of 512, as used when training Single-field LMs.

The third part of the table shows performances of our best four Stacking models. We evaluated many final estimators and combinations of first estimators and we submitted the scores from the models that performed better on our validation set. We computed the validation performance of stacking models using 5-fold cross-validation on the original validation dataset.

Finally we report the performance of the best team that participated to the competition. Up to now we do not know details about their approach.

5.1 Label rescaling

When we train our models we have to linearly scale the labels from the original range [1, 4]. While the straightforward choice of the final range could be [-1, 1] since our scores naturally fits that range due to the nature of the final cosine similarity computation, we observe that in practice, there are no pairs of titles from the training dataset that score less than -0.15 when we use a pre-trained model (see Figure 3 (left) for the complete distribution). Thus, we treat the lower bound of the transformed range s_{min} as an hyper-parameter to set. Figure 3 (right) shows values of Pearson Correlation on the validation dataset during the first training epoch for different choices of this parameter. We noticed that setting $s_{min} = -1$ as previously hypothesised leads to a slow training phase. We find $s_{min} = -0.1$ the best value among the tested ones. On the contrary, setting the higher bound $s_{max} = 1$ is optimal.

Language Pair	Test
ar-ar	66.2
de-pl	68.4
de-fr	71.3
pl-pl	71.5
ru-ru	72.9
zh-zh	76.8
es-it	77.3
de-de	78.0
tr-tr	78.3
de-en	82.2
it-it	82.4
zh-en	82.6
es-es	82.9
es-en	83.3
fr-fr	86.2
en-en	86.7
pl-en	87.1
fr-pl	88.3

Table 2: Pearson's $r \times 100$ of our best model on subsets of the Test dataset.

5.2 Error analysis

We report in Table 2 the performance of our best model for each language combination of the Test set. We believe that lower correlations are due to scraping issues, translation issues and to the different distribution of languages between the Training and Test sets.

6 Conclusion

To quantify the similarity between news articles, we propose an approach trained exclusively on the extracted textual data. We initialize our architecture with SOTA Semantic-Similarity Language Models, which we fine-tuned on titles, descriptions and texts of the articles. We also design a simple variant to process many textual fields at once. Finally, we perform stacking with a simple MLP, as it was proved to improve the overall performance of models trained on different features. Results show how the approach successfully estimated similarities since the main sources of error involved missing or wrong data.

References

N. S. Altman. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185.

Adrien Barbaresi. 2021. Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 122–131. Association for Computational Linguistics.

Ekaba Bisong. 2019. *Google Colaboratory*, pages 59–64. Apress, Berkeley, CA.

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. In *Machine Learning*, pages 273–297.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy. PMLR.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Arthur E. Hoerl and Robert W. Kennard. 2000. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86.

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

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

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

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *arXiv preprint arXiv:2004.09297*.

Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

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

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

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5(2):241–259.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

WueDevils at SemEval-2022 Task 8: Multilingual News Article Similarity via Pair-Wise Sentence Similarity Matrices

Dirk Wangsadirdja and **Felix Heinickel** and **Simon Trapp**
{name}.{surname}@stud-mail.uni-wuerzburg.de
Albin Zehe and **Konstantin Kobs** and **Andreas Hotho**
{surname}@informatik.uni-wuerzburg.de
University of Würzburg

Abstract

We present a system that creates pair-wise cosine and arccosine sentence similarity matrices using multilingual sentence embeddings obtained from pre-trained SBERT and Universal Sentence Encoder models respectively. For each news article sentence, it searches the most similar sentence from the other article and computes an average score. Further, a convolutional neural network calculates a total similarity score for the article pairs on these matrices. Finally, a random forest regressor merges the previous results to a final score that can optionally be extended with a publishing date score.

1 Introduction

The goal of the *Multilingual News Article Similarity* task (Chen et al., 2022) is to check pairs of multilingual news articles against each other in terms of similarity of their information content. The challenge focuses on *what* is talked about (time, geolocation, shared entities), not *how* the information is expressed (writing style, emotional tone, etc.), which is essential for applications such as analyzing the news coverage between different regions. The participants' objective is the creation of a model that rates the similarity of article pairs on a 4-point scale from most (1) to least (4) similar and achieves the highest possible Pearson correlation score compared to the gold standard. The languages covered by this competition are English (en), German (de), Spanish (es), Turkish (tr), Polish (pl), French (fr) and Arabic (ar) in the training, additionally Italian (it), Russian (ru) and Chinese (zh) in the evaluation.

Our system uses an ensemble approach to score pair-wise sentence similarity matrices of the article pairs, which are created with SBERT and Universal Sentence Encoder sentence embeddings. Scores are obtained through simple matrix operations and our convolutional neural network *SimCNN* based on the TextCNN by Kim (2014). Finally, a random forest regressor (Breiman, 2001) consolidates the individual scores into a final result, which we extend with a publishing date score.

The key challenge of this task is the usage of multilingual text pairs, which requires the application of less precise multilingual language models. Also the splitting of sentences and named entity recognition becomes hard to accomplish, since models for these tasks are still monolingual in most cases and we found the few multilingual ones to be unreliable. The article scraping tool provided by the authors also did not reveal meaningful features to work with besides the article title and text, since the scraped keywords, tags and publishing dates were not always available and in a utilizable format.

In the competition, we took 9th place with a Pearson correlation coefficient of 0.759 on the evaluation set, which differs from the observed performance on our own validation sets. This is due to a shift from Latin languages to more complex languages like Chinese in the evaluation data, where the performance of the multilingual models and sentence tokenizing algorithm decreases. Our code is publicly available[1].

2 Background

The starting point of the task is the training data — a CSV file — which contains a list of article pairs[2]. Each article pair consists of the languages, IDs and URLs for both articles and the gold standard similarity scores for multiple aspects of the articles (Geography, Entities, Time, Narrative, Style, Tone, Overall), of which only the overall score is relevant for our evaluation.

To get the content and metadata of the articles,

[1] https://github.com/simontrapp/semeval-22-task-8

[2] We used the most recent version 0.2 with 4,964 labeled article pairs.

the task authors provide a scraping tool[3] utilizing the *newspaper3k* python package that downloads the HTML page and creates a JSON file that contains the title, text, keywords, labels, date, and more properties of each news article. For the training of the prediction system on the JSON files, the results can be compared against the overall scores in the initial CSV file. The final evaluation data CSV, for which the results are submitted, does not provide scores to compare against.

3 Related Work

For the scoring of our similarity matrices in subsection 4.4, we use an idea from Ginzburg et al. (2021), who introduce Self-Supervised Document Similarity Ranking (SDR), an unsupervised approach built upon the RoBERTa language model to rank the semantic similarity of a collection of documents to a source (query) document. SDR captures the intuitive fact that for each sentence or paragraph in one document, there should be at least one similar one *anywhere* (obtained by a *max* operation on the rows and columns of the similarity matrix) in the other document if both deal with the same topic. Since SDR is monolingual and is only trained on English texts, we combine its scoring approach with multilingual embeddings obtained from SBERT (Reimers and Gurevych, 2019) and Universal Sentence Encoder (Cer et al., 2018) for this challenge.

4 System Overview

Our system (see Figure 1) first splits the article text into a list of sentences including the title. Then we compute embeddings with SBERT and Universal Sentence Encoder for all sentences and create the respective similarity matrices (similarity of all sentences of one article to all sentences of the other) of all article pairs. In the next step, we apply two different scoring approaches to these matrices: First, we apply simple maximum and average operations to the two matrices for four similarity scores, as proposed by Ginzburg et al. (2021). Second, we feed the matrices into our *SimCNN* to increase score accuracy (see Appendix C). Finally, we combine the five scores into a final score with a random forest regressor to get a stable prediction. Optionally, an additional score for the publishing date distance of both articles can be computed and merged with the random forest result to refine the prediction.

[3]`https://github.com/euagendas/semeval_8_2022_ia_downloader`

4.1 Preparation of Article Data

We decided that only the title and the text of the article should be relevant for our model, since other features such as keywords, tags or publishing date of the articles are not always available or feasibly retrievable. We use the `sent_tokenize` function of the `nltk` python package to split the text of both articles into a list of sentences and append their title strings to the respective list. This allows us to feed text of arbitrary length into the embedding models.

4.2 Creation of Embeddings and Similarity Matrices

For all sentences in the lists, we create two separate sets of sentence embeddings:

The Universal Sentence Encoder (Yang et al., 2019) always uses the same pre-trained model (see subsection 5.2) to create the embeddings, for which we calculate the recommended *arccos*-based text similarity (Yang et al., 2018), that converts the cosine values into angular distances in $[0, \pi]$ pair-wise between all sentences of both articles.

For SBERT, the model that is selected to create the embeddings depends on the languages of both articles: If both are the same and a model with better performance than the multilingual one is known (see subsection 5.2), this model computes the embeddings for both articles, otherwise the default multilingual model is used. The pair-wise similarity matrix between article sentences is created analogously to the Universal Sentence Encoder, but with cosine similarity instead of *arccos*, because SBERT was optimized for it.

4.3 Architecture of SimCNN

The architecture of our CNN is based on the architecture by Yoon Kim for CNNs for text processing (Kim, 2014) (detailed layer information in Appendix A). As input, we use both precomputed sentence embeddings from the SBERT and the Universal Sentence Encoder model of two articles with lengths x and y. Further, we calculate one similarity matrix from both the SBERT and the Universal Sentence Encoder embeddings since tests have indicated an increased performance using this input. Finally, the input of the CNN is an $x \times y \times 2$ matrix, generated by concatenating the SBERT and the Universal Sentence Encoder similarity matrices. Due to this input, we named our network *SimCNN*. However, the CNN requires

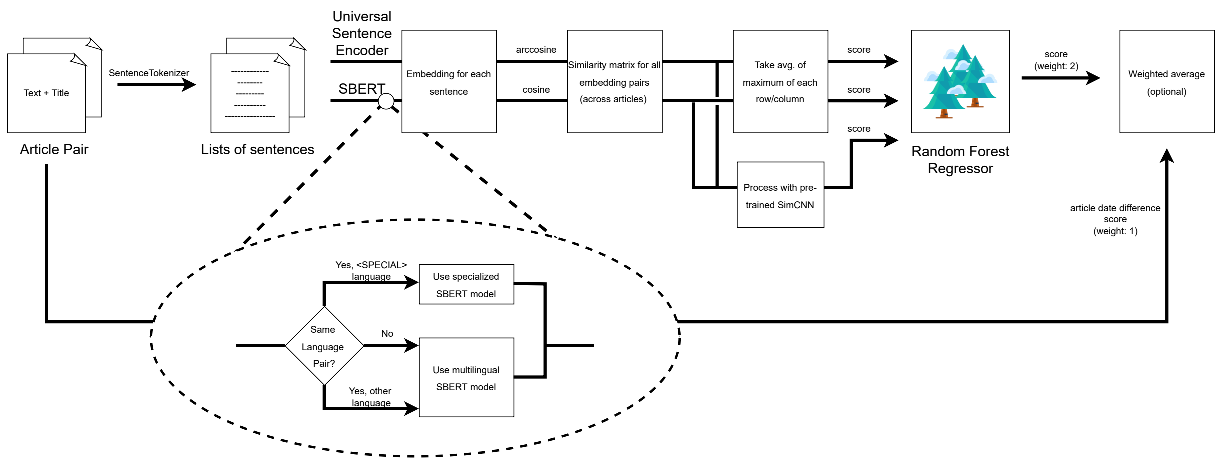

Text + Title
Article Pair
SentenceTokenizer
Lists of sentences
Universal Sentence Encoder
SBERT
Embedding for each sentence
arccosine
cosine
Similarity matrix for all embedding pairs (across articles)
Take avg. of maximum of each row/column
Process with pre-trained SimCNN
score
score
score
Random Forest Regressor
score (weight: 2)
Weighted average (optional)
article date difference score (weight: 1)
Same Language Pair?
Yes, <SPECIAL> language
No
Yes, other language
Use specialized SBERT model
Use multilingual SBERT model

Figure 1: The data flow of the article pairs from raw text to the final score. If both articles are written in the same language, we use an SBERT model that is better than the default multilingual one, if one exists (see subsection 5.2).

a fixed input size within the y-dimension, so we set y to 100 (using zero-padding if $y < 100$ and cropping if $y > 100$). We chose $y = 100$ based on analysis that indicates only a few articles have longer sentence lists and do not set it to the maximum sentence list length since broken lists with up to 1500 sentences can occur.

Adapted from the TextCNN of Yoon Kim, the network consists of seven different convolutions with kernel size $w \times 100$ with $w \in \{2, 3, \ldots, 9\}$, and 128 filters, named sliding window. For extended feature extraction, we added two convolutional blocks before each sliding window SW_w, consisting of five convolutions. Thereby, each convolution of a block SW_k uses a kernel size of $w \times w$, the same padding, and 32 filters within the first block and 64 within the second. Furthermore, each convolution follows a ReLU6 activation and a batch normalization layer and, additionally, a dropout layer ($probability = 0.25$) after the 2nd and 4th convolution. After each sliding window convolution, a max over time pooling extracts the best feature of each filter, so we get a output vector of size 128. Afterwards, a separate linear layer is applied to each vector(mapping to 128 features), followed by another dropout ($probability = 0.5$) and a ReLU6 layer.

For the final prediction, we concatenate all vectors of the different sliding windows to one vector of size 1024 that is fed into five consecutive linear layers. Thereby, the output size for each linear layer is half the input size and every layer uses a ReLU6 activation function and, additionally, a dropout layer ($probability = 0.5$) every second time. Finally, a linear layer with an input size of 32 and output size of 1 predicts a score $s \in [0, 1]$ using a sigmoid activation function. This score is scaled to our target values $s \in [1, 4]$ subsequently.

4.4 Ensemble Scoring of Article Similarity

In addition to being fed into the SimCNN, the *arccos* and *cosine* similarity matrices of the article pairs are processed by taking the average over the maximum value of each row/column of the matrix, similar to the approach by Ginzburg et al. (2021) in section 3. The maximum values yield the most similar sentence in the other article for each sentence and the average operation acts as a proportion of how many sentences of one article share statements with the other article. By doing this for both rows and columns, we obtain two scores for both articles respectively. Figure 2 visualizes this process.

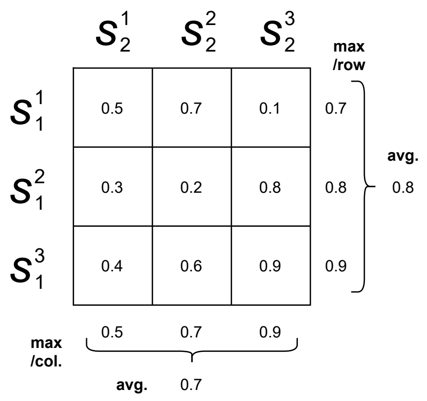


Figure 2: Example of the scoring operation on the *arccos* and *cosine* sentence similarity matrices. The superscript number denotes the number of the sentence in the document, the subscript number marks the document number of the sentence.

Finally, the four matrix scores (article similarity SBERT 1-to-2 and 2-to-1, Universal Sentence Encoder 1-to-2 and 2-to-1) and the SimCNN score are fed into a random forest (Breiman, 2001) regressor from the `scikit-learn` Python package[4]. The forest uses the results of 100 different trained decision trees and takes the average of their predictions, which is the final score reported back by the model.

4.5 Integration of a Publishing Date Score

We also compute scores based on the publishing date distances of each article pair (if available) for the reason that the further the publishing dates are apart, the more likely the articles are about different topics. In accordance with the 4-point score used for the text scoring, we use 10, 20, and 50 days as the boundaries, meaning the score would be ignored if the difference is less than 10 days, between 2 and 3 if the difference is between 10 and 20 days, between 3 and 4 if the difference is between 20 and 50 days, and a hard 4 for a difference of more than 50 days.

If a date score can be calculated and is not ignored, a weighted average of the random forest score (weight 2) and the date score (weight 1) is returned as a final result, otherwise the random forest score is returned. The 10, 20, and 50 day boundaries, as well as the weights, were determined based on experiments on the training data.

5 Experimental Setup

After outlining the model components and their interactions in the previous section, here we cover the data sets, experiments and training processes used to configure the separate parts of the system. The sole evaluation metric for this competition is the Pearson correlation coefficient r (Pearson, 1896; Lee Rodgers and Nicewander, 1988), which describes the linear association between two related variables X and Y. A score close to -1 or 1 implies that a linear equation can express the relationship between the two variables almost perfectly, while a score of 0 indicates no correlation.

5.1 Validation Data Sets

The only labeled data available for training are the 4,964 article pairs provided by the task authors. We reserve a static 10% subset of the training data consisting of 470 pairs of diverse language combinations for our model validation in Table 2, Appendix D and the following experiments. The other 90% of the data are used for the actual training.

5.2 Selection of Pre-Trained Models

The pre-trained multilingual SBERT model *paraphrase-multilingual-mpnet-base-v2* performed best overall on the validation set, so it is used as the default model. We could improve the accuracy for some same-language article pairs by using specialized pre-trained models (**en-en**: *all-mpnet-base-v2*, **es-es**: *distiluse-base-multilingual-cased-v1*, and **fr-fr**: *sentence-transformers/LaBSE*). For Universal Sentence Encoder we used version 3 of the model *multilingual-large*.

5.3 SimCNN Pre-Training

The presented model was implemented in python using `PyTorch` (Paszke et al., 2019) and trained on a consumer graphics card. For updating the parameters in the network, we employed the Stochastic Gradient Decent optimization algorithm using a learning rate of 0.05 with a batch size of 8. Mean squared error was used as the loss function, and additionally, we monitored the mean average error and the Pearson correlation coefficient for performance evaluation. We used early stopping with a patience of 20 epochs to prevent overfitting, so the network was saved when no longer improving in terms of the Person correlation coefficient. Eventually, our network was trained around 30 epochs before overfitting.

5.4 Random Forest Regressor Pre-Training

After the training of the SimCNN model, five similarity scores per article pair of the training data are available through our model pipeline: Two scores for both the SBERT and Universal Sentence Encoder matrices and the SimCNN score. These scores are fed into a random forest (Breiman, 2001) regressor[5] with the provided *Overall* scores of the training data as labels. The random forest is populated with 100 decision trees and uses the squared error as the optimization criterion. Appendix C shows how the performance of the model increases as we provide more data.

[4]`https://scikit-learn.org/stable/modules/ensemble.html#forest`

[5]RandomForestRegressor of the python package `scikit-learn` with version 1.0.2.

Data Set	Language Combination								
	pl-pl	de-de	de-en	fr-fr	ar-ar	en-en	tr-tr	es-es	zh-zh
Validation	0.80	0.75	0.80	0.91	0.76	0.79	0.89	0.81	-
Evaluation	0.63	0.67	0.77	0.74	0.59	0.85	0.66	0.74	0.64

Table 1: Performance of the random forest with date score on selected language combinations. The full table with all pairs is in Appendix D.

Model	Pearson r	
	Validation	Evaluation
SimCNN	0.800	0.699
RF	0.797	0.702
RF + Date	0.800	0.715

Table 2: Performance of our models on our validation set and the final evaluation data. The random forest (RF) is able to improve over just the SimCNN on the evaluation data when combined with the date score.

6 Results

Our system ranked 9th place in the competition with a Pearson score of 0.759.

To find the best models for submission, we tested the three last stages of our system on the aforementioned validation set separately: Just the SimCNN score, the prediction of the random forest regressor and the weighted average of the random forest score and the publishing date score.

The results in Table 2 indicate that the random forest regressor replicates the results of the SimCNN and barely considers the additional information provided. The combination of the random forest with the publishing date score on the other hand improves the results. Further, our models generally predict perceptibly worse scores on the evaluation data than on the training data split.

The language distribution in Appendix B shows a shift from Latin languages and article pairs to vastly different and more complex languages: Chinese-Chinese article pairs account for over 15% of article pairs in the evaluation data. Russian, Polish and Arabic articles also occur often, frequently in combination with other article languages such as English, French and German. This leads to problems in the preparation of the article texts for our system because our sentence tokenizer only supports English or similar texts and many of the new languages have a different structure and alphabet.

When taking a look at the performance per language pair of our model in Table 1, another reason for the score drop-off between training split and evaluation data becomes apparent: The commonly occurring Chinese-Chinese article pairs perform bad with a Pearson score of 0.64. Also, other combinations which previously did well on the split of the training data gave significantly worse results on the evaluation set, indicating that maybe the quality or structure of the new data differs from the earlier samples. Interestingly, our system improved on the English-English evaluation pairs.

All things considered, we achieved satisfying results with just slightly modified publicly available models to create sentence embeddings and a CNN to work with them. The system only depends on an article's text, title and sometimes the publishing date, if it is available, and is still able to achieve a Pearson correlation score of about 0.6 even for the most difficult examined language.

7 Conclusion

Pair-wise sentence comparison is a simple way to calculate the similarity of texts of arbitrary length if suitable multilingual models for sentence embeddings are available. With simple matrix operations like taking the maximum or average and a random forest regression algorithm, good results can be achieved. After introducing the more complex SimCNN and combining it with a score of the publishing dates of the article pairs, our model surpassed a Pearson correlation coefficient of 0.8 in some conditions.

Nevertheless, the current state of the system leaves many things to be improved: The sentence tokenizing currently only reliably works for English and similar languages. With a sentence splitting algorithm, that is capable of differently structured languages like Chinese or Japanese, results on such articles could be greatly improved. Further, our pair-wise sentence similarity matrix approach could be extended to named entities like locations and persons, which we think would also greatly improve accuracy, but for that, a more sophisticated multilingual algorithm for named entity recognition would be needed.

References

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

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

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Dvir Ginzburg, Itzik Malkiel, Oren Barkan, Avi Caciularu, and Noam Koenigstein. 2021. Self-supervised document similarity ranking via contextualized language models and hierarchical inference. *arXiv preprint arXiv:2106.01186*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification.

Joseph Lee Rodgers and W Alan Nicewander. 1988. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

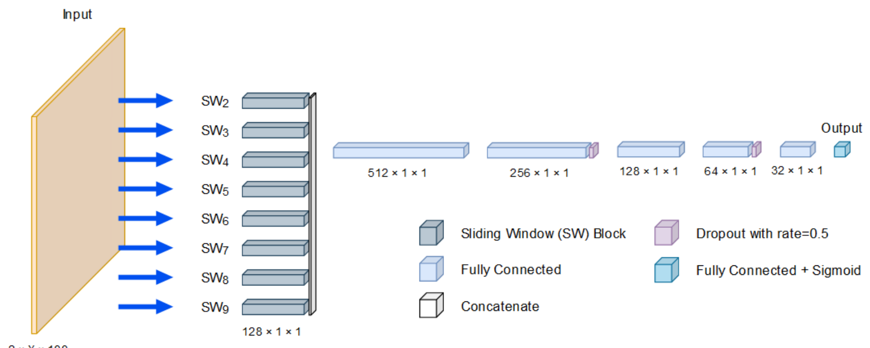
Karl Pearson. 1896. Vii. mathematical contributions to the theory of evolution.—iii. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 187:253–318.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

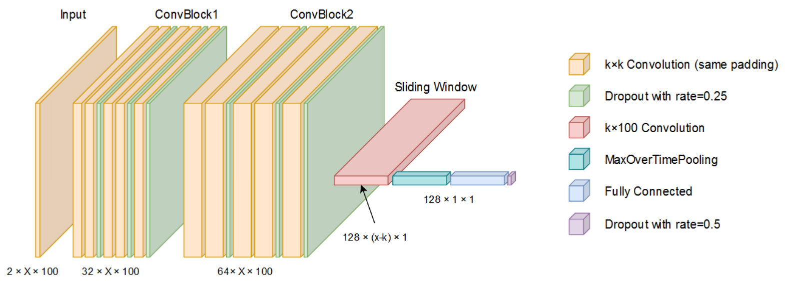
Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Multilingual universal sentence encoder for semantic retrieval. *CoRR*, abs/1907.04307.

Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning semantic textual similarity from conversations. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 164–174, Melbourne, Australia. Association for Computational Linguistics.

A Architecture of SimCNN



(a) SimCNN



(b) Sliding Window Block with size k

Figure 3: Architecture and methods of the SimCNN based on the TextCNN described by Kim (2014).

Figure 3a shows the architecture of the SimCNN. The SimCNN is based on the architecture by Yoon Kim for CNNs for text processing (TextCNN) (Kim, 2014). It consists of seven sliding window blocks SW_w with window size $w \in \{2, \dots, 9\}$. Each block SW_w receives the input of size $2 \times x \times 100$ and is structured as illustrated in graphic 3b. First, two convolution blocks, with five convolutions each, are applied to the input. Each convolution of SW_w has a kernel size of $w \times w$ and uses same padding. The convolutions of the first block have 32 filters and the ones of the second block 64. After a convolution, first, a ReLU6 activation layer is applied and, subsequently, a batch normalization layer. Furthermore, after the second, fourth and last convolution, a dropout is executed with a probability of 0.25. Afterwards, the sliding window convolution is applied, using a kernel size of $w \times 100$ and 128 filters, followed by a ReLU6 and a batch normalization layer. Next, a MaxOverTime pooling layer extracts the best feature of each filter. Last of the sliding window block, a fully connected layer using a Relu6 activation function, followed by a dropout with a probability of 0.5, maps to a feature vector of 128 features.

After all sliding window blocks $SW_2, \dots, SW_9$, the outputs are concatenated to a feature vector of 1024 features. Five fully connected layers with output sizes 512, 256, 128, 64 and 32 are applied to this vector. A Relu6 is used as activation function after these layers, and a dropout is performed after the second and fourth layer with a probability of 0.5. Finally, a fully connected layer maps the feature vector to one number, that is processed in a sigmoid function, to get the score $s \in [0, 1]$

B Language Distributions of Data Sets

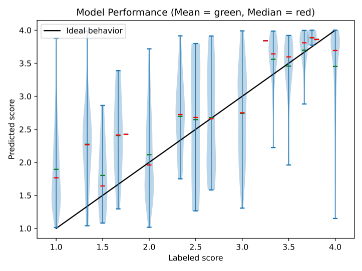
	en-en	de-de	de-en	es-es	tr-tr	pl-pl	ar-ar	fr-fr
Absolute Count	1800	857	577	570	465	349	274	72
Percentage [%]	36.26	17.26	11.62	11.48	9.37	7.03	5.52	1.45

Table 3: In the distribution of training data language pairs, English-to-English is the prevalent combination, with other similar European languages following. With our specialized English-to-English SBERT model, we therefore achieve very good results.

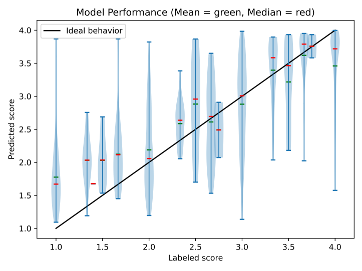
	zh-zh	de-de	es-en	it-it	es-it	ar-ar	ru-ru	tr-tr	es-es
Absolute Count	769	608	496	411	320	298	287	275	243
Percentage [%]	15.69	12.4	10.12	8.38	6.53	6.08	5.85	5.61	4.96
	en-en	pl-pl	zh-en	de-en	de-fr	fr-fr	pl-en	de-pl	fr-pl
Absolute Count	236	224	213	185	116	111	64	35	11
Percentage [%]	4.81	4.57	4.35	3.77	2.37	2.26	1.31	0.71	0.22

Table 4: In the evaluation set, the use of languages is vastly different: Previously unseen combinations (gray), often with complex languages like Chinese, make up a large part of the article pairs the models are scored upon.

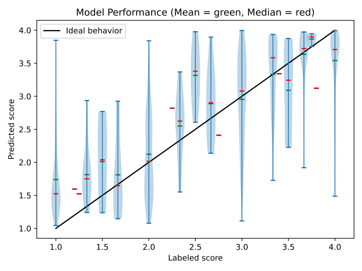
C Random Forest Performance with Increasing Amount of Training Data



(a) SBERT (cosine) scores only ($r = 0.703$)



(b) SBERT + Universal Sentence Encoder scores ($r = 0.748$)



(c) SBERT + Universal Sentence Encoder + SimCNN scores ($r = 0.77$)

Figure 4: Performance of the random forest regressor with different inputs on a random train-test-split with 80% training and 20% test data. The more data is provided, the better our Pearson correlation score r gets.

D Model Performance per Language

Model	Language Combination								
	pl-pl	de-de	de-en	fr-fr	ar-ar	en-en	tr-tr	es-es	es-it
V SimCNN	0.89	0.75	0.82	0.83	0.77	0.80	0.81	0.80	-
V Random Forest	0.90	0.75	0.81	0.88	0.76	0.79	0.83	0.80	-
V Publish Date	0.80	0.75	0.80	0.91	0.76	0.79	0.89	0.81	-
E SimCNN	0.61	0.67	0.76	0.69	0.56	0.86	0.65	0.73	0.73
E Random Forest	0.61	0.66	0.76	0.72	0.58	0.85	0.67	0.73	0.73
E Publish Date	0.63	0.67	0.77	0.74	0.59	0.85	0.66	0.74	0.73
	fr-pl	pl-en	de-pl	zh-en	it-it	ru-ru	de-fr	zh-zh	es-en
E SimCNN	0.71	0.77	0.62	0.76	0.76	0.73	0.60	0.64	0.77
E Random Forest	0.70	0.77	0.62	0.75	0.76	0.73	0.58	0.63	0.77
E Publish Date	0.70	0.78	0.62	0.78	0.79	0.74	0.59	0.64	0.80

Table 5: Pearson correlation score of our model configurations on different language pairs. They often do perform significantly better on the validation data (V) than on the evaluation data (E) and seldom vice versa. The additional languages in the evaluation data do not perform noticeably worse than some of the languages already seen in the training set.

SemEval-2022 Task 9: R2VQ – Competence-based Multimodal Question Answering

Jingxuan Tu, Eben Holderness, Kyeongmin Rim, Kelley Lynch, Richard Brutti, James Pustejovsky
Lab for Linguistics & Computation
Department of Computer Science
Brandeis University
{jxtu,egh,krim,kmlynch, brutti,jamesp}@brandeis.edu

Marco Maru[1], Simone Conia[1], Roberto Navigli[2]
Sapienza NLP Group
[1]Department of Computer Science
[2]Department of Computer, Control and Management Engineering
Sapienza University of Rome
first.lastname@uniroma1.it

Abstract

In this task, we identify a challenge that is reflective of linguistic and cognitive competencies that humans have when speaking and reasoning. Particularly, given the intuition that textual and visual information mutually inform each other for semantic reasoning, we formulate a Competence-based Question Answering challenge, designed to involve rich semantic annotation and aligned text-video objects. The task is to answer questions from a collection of English language cooking recipes and videos, where each question belongs to a "question family" reflecting a specific reasoning competence. The data and task result is publicly available. [1]

1 Introduction

One of the fundamental goals of Artificial Intelligence (AI) has been to create systems that interact with human users fluently and intelligently, by demonstrating inferencing and reasoning capabilities that would be expected of a human partner. This includes a growing interest in posing larger challenges to end-to-end systems employing architectures with deep neural networks (DNNs) (Ribeiro et al., 2020; Prabhumoye et al., 2020; Rogers et al., 2021; Minaee et al., 2021). Here we argue that we should start focusing on linguistic *competencies*, and not just on Question Answering (QA) skills or "challenge checklisting". There are some moves in this direction already (Johnson et al., 2017), but there is still no generally accepted distinction in current Natural Language Processing (NLP) between challenge-based tasks and competence-based performance (Bentivogli et al., 2017). Analogous to human cognitive competencies, there is both a methodological and modeling advantage to focusing a system's performance on competence-based learning rather than a narrowly defined task or challenge checklist.

First we define competence-based knowledge, and then the questions that can be generated from such knowledge. While Chomsky (1965)'s distinction between competence and performance has long been debated in linguistics, the term *competence-based* has been applied to a number of different concepts in both the science of learning and educational communities (Bechtel et al., 1999; Voorhees, 2001; Chyung et al., 2006; Platanios et al., 2019; Hsiao et al., 2020). The common core to both is a concept capturing a coherent set of abilities that an individual has in a specific domain (Doignon and Falmagne, 1985; Heller et al., 2013).

Here we focus on *lexical competence* as deployed in both single and multiple sentence composition (Pustejovsky, 1995; Marconi, 1997; Geeraerts, 2009; Asher, 2011). A competence-based question will query competence-based knowledge structures. For this task, lexical competence will involve the following:

- Understanding implicit arguments that are not present (due to syntactic ellipsis or semantic defaulting or shadowing), and being able to use this (missing) information to formulate knowledge about the event or situation (Malmaud et al., 2014; Kiddon et al., 2015);
- Understanding the dynamics of the text or narrative and how events can change an object or contribute to new properties (and subsequent descriptions) of objects in the text (Tandon et al., 2018; Das et al., 2018; Brown et al., 2018).

It is clearly the case that these two phenomena require non-extractive QA capabilities of some sort. We describe our dataset, Recipe-to-Video Questions (R2VQ), and summarize the procedures implemented by task participants for answering such

[1]https://competitions.codalab.org/competitions/34056

questions in the remainder of the paper.

2 Overview

2.1 Summary of the task

The task is structured as QA pairs, querying how well a system understands the semantics of English language recipes.

We hope that this task will help move NLP system design and evaluation towards the construction of meaning representations involving linguistic and multimodal situated grounding. In the present context, this involves identifying cooking entities and activities from recipe text, as well as linking them to videos of related recipes, entities, and activities.

Participants are provided with a multimodal training set, and are asked to provide answers to unseen queries. These questions can be answered using a unimodal dataset of text recipes and associated annotations. Participants are also encouraged to explore the full multimodal training set with additional cooking videos to potentially improve the results from the unimodal models. Following SemEval guidelines, the R2VQ dataset is publicly available[2] in CONLL-U format, with annotations encoded in plain text files.

2.2 Impact of the task

When we apply our existing knowledge to new situations, we demonstrate a kind of understanding of how the knowledge (through tasks) is applied. When viewed over a conceptual domain, this constitutes what we will refer to as a *competence*, and the corresponding challenge can be called a competence-based challenge. Competence-based evaluations can be seen as a new approach for designing NLP challenges, in order to better characterize the underlying operational knowledge that a system has for a conceptual domain, rather than focusing on individual tasks.

3 Related Work

NLP challenges have helped drive progress in the field recently. These challenges in part have been framed as specific tasks, and advances are largely driven by leaderboards on benchmark datasets or model comparison on individual datasets. Common benchmarks such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) have been used widely. They contain several language understanding tasks such as Winograd Natural Language Inference (WNLI) (Levesque et al., 2011) as an inference task, and Winograd Schema Challenge (WSC) (Levesque et al., 2011) as a coreference resolution task. A survey (Rogers et al., 2021) showed the recent trend to measure various machine reasoning capabilities using different designs of QA tasks.

While all the tasks aim to advance the research towards corresponding NLP challenges, whether these reflect human competencies remains a question, especially in recent years with the success of transformers (Devlin et al., 2019; Yang et al., 2019; Liu et al., 2019). Many top-ranked NLP models that have shown better performance than humans on benchmarks may have come from overfitting to the dataset rather than addressing the challenge (Rogers, 2019). Current pre-training paradigms may also tune models towards capturing merely statistical patterns, so datasets should be designed to align the model's ability with human expectations (Linzen, 2020). Sugawara et al. (2020) found that most of the questions from common QA and reading comprehension datasets can be correctly answered by models without complex reasoning.

Recent work has been trying to identify and evaluate the tasks that are reflective of human linguistic and reasoning competencies. For example, Kim and Linzen (2020) proposed a semantic parsing dataset that evaluates the human-like compositional generalization of models. Ribeiro et al. (2020) designed three test types that can be used to test various linguistic capabilities of NLP models. More closely related to our work, QA-SRL (He et al., 2015) use predicate-argument structure to represent QA pairs. SynQG (Dhole and Manning, 2021) and RoleQ (Pyatkin et al., 2021) try to incorporate existing semantic annotations to generate comprehension questions.

4 Task Description

We formulate the task as competence-based QA, designed to involve rich semantic annotation and aligned text-video objects. The goal of this task is to answer questions from a collection of cooking recipes and images. Each question belongs to a "question family" that characterizes a specific reasoning competence to be tested. These competencies include abilities such as spatial and temporal reasoning, semantic role assignment, and object

[2] https://competitions.codalab.org/competitions/34056#participate

Recipe Title: **Appelkoek** *Passage:* Peel and cut apples into eighths (wedges). Sift together flour, baking powder and salt with 4 tablespoons of the sugar. Cut in butter. Combine egg and milk and add to flour mixture. Turn batter into greased 8 inch square cake pan. Press apple wedges partly into batter. Combine remaining 2 tbsp sugar and cinnamon. Sprinkle over apple. Bake at 425 degF for 25 to 30 minutes.

IMPLICIT	How do you cut apples into wedges? - by using a knife
ELISION	What should be sprinkled over apple wedges? - cinnamon sugar
LOC. CHANGE	Where was the batter when you press apple wedges? - in the pan
OBJ. LIFESPAN	What's in the appelkoek? - apples
SRL-TIME	For how long should you bake appelkoek? - 20 to 35 minutes
SRL-VALUE	How do you bake appelkoek? - bake at 425 degF

Table 1: Example competence-based questions. Color-coded text spans represent how information has been collected and generated in the questions.

cardinality and counting.

We adopt the concept of "question families" as outlined in the CLEVR dataset (Johnson et al., 2017). While some question families (e.g., integer comparison, counting) naturally transfer over from the Visual Question Answering (VQA) domain (Antol et al., 2015; Zhu et al., 2016), other concepts such as ellipsis and object lifespan must be employed to cover the full extent of competency within procedural texts. On the basis of the aforementioned competencies, we categorize the questions into five question families. Table 1 shows the definition of each question family as well as sample questions.

The question families are defined as follows:

- *Cardinality*: covers concepts of integer comparison and counting.
- *Elision*: deals with identifying arguments (ingredients in most cases) that are omitted from a text, but can be understood from context.
- *Implicit*: covers both implicit tools and habitats introduced in the text. This is distinct from elision, as these are not solved merely through contextual clues. Instead, they require general competence; applying world knowledge of an action and its requirements to a novel situation.
- *Obj. Lifespan*: covers different states of an object in a cooking event.
- *Semantic Role Labeling (SRL)* covers semantic roles that are modifiers to a cooking event.

5 Data and Resources

The textual component of our dataset consists of a collection of English language recipes sourced from two open-source recipe wikis, Recipe Fandom[3] and Foodista[4], and is labeled according to three distinct annotation layers: (i) Cooking Role Labeling (CRL), (ii) Semantic Role Labeling (SRL), and (iii) aligned key frames image triples taken from creative commons cooking videos downloaded from YouTube.

Compared to text of news or narratives, procedural text such as recipes and user manuals tend to be task-oriented, and the main content is split into steps that describe small goals to accomplish the final task. We believe such texts are a good fit for our task, as it involves the understanding of how to reach the goal locally for each step, as well as how each step contributes to the final task globally. Further, the step-wise progression inherent in the goal-oriented narrative contributes both an interpretative dynamics as well as contextualized elision of arguments.

5.1 Train/Dev/Test Datasets

There are 1,000 recipes released as part of the task (800 for training and 100 each for validation and testing). Table 2 shows the basic statistics of the dataset. We exclude any "less informative" recipe that has less than 4 sentences from our dataset. For each recipe, there are an average of 35 questions (5 from each question family). Each recipe is also paired with an additional set of 10 "unanswerable" questions (answers that cannot be found in a given recipe) as negative samples.

5.2 Cooking Role Labeling

Cooking Role Labeling (CRL) is a domain-specific dependency relation annotation for the cooking domain. CRL is done via a two-phase annotation. First, to identify mentions of cooking events and

[3]https://recipes.fandom.com/
[4]http://foodista.com/

	Train	Dev	Test
# of recipes	800	100	100
Avg. # of sentences per recipe	8	7.9	7.8
Max. # of sentences	26	16	31
Min. # of sentences	4	4	4
Avg. sentence length per recipe	12.5	13.4	12.5
Max. sentence length	32	25	19
Min. sentence length	6	6	7

Table 2: Statistics of the train, dev and test subsets of the R2VQ dataset.

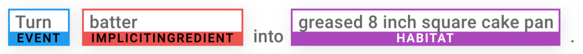


Figure 1: Docanno environment for event and entity annotation.

entities and put labels on them, and then to establish relations between those mentions.

Each step in a given recipe is annotated for cooking-related *events* and the associated *entities* (ingredients and props such as tools, containers, and habitats). The ingredients can be either labeled as explicit (those listed in the ingredients section of the recipe) or implicit (intermediate outputs of applying a cooking action to a set of explicit ingredients).

We post-process the data by running the Stanza pipeline (Qi et al., 2020) on the raw text of each recipe to get tokenization and other basic linguistic features including word lemmas, part-of-speech tags. We took a semi-automated approach to performing the span-level entity annotations. First, using a small labeled dataset as seed training data, we trained a character-level named entity recognition (NER) model using Flair embeddings (Akbik et al., 2019) to pre-annotate the recipe text. We then validated the model predictions using the Docanno annotation tool (Nakayama et al., 2018) to create our gold set of event and entity mentions. Figure 1 shows an example from the Docanno environment, with annotations for Event, Implicit Ingredient, and Habitat.

For the next phase of annotation, we developed Deep Event & Entity Palette or DEEP, a specialized annotation environment to manually annotate cooking role relations. Annotators start from documents that are already annotated with span-level entity tagging from Docanno. The primary job of annotators is to draw links between entities (coreference) or between an entity and an event (participant). DEEP provides an intuitive and easy interface for pairwise linking annotation, as well as a holistic view of the document-level context using color coding of tokens related to the selected events or entities, as shown in Figure 2. All annotation is done at document-level, namely, annotators can create long distance links. For example, a food entity from a previous step can be linked to an event in the next step even if the direct object of the event is omitted on surface (or "hidden"). And finally, DEEP also provides an interface to add such hidden entities with a free-text identifier and immediately link it to an event.

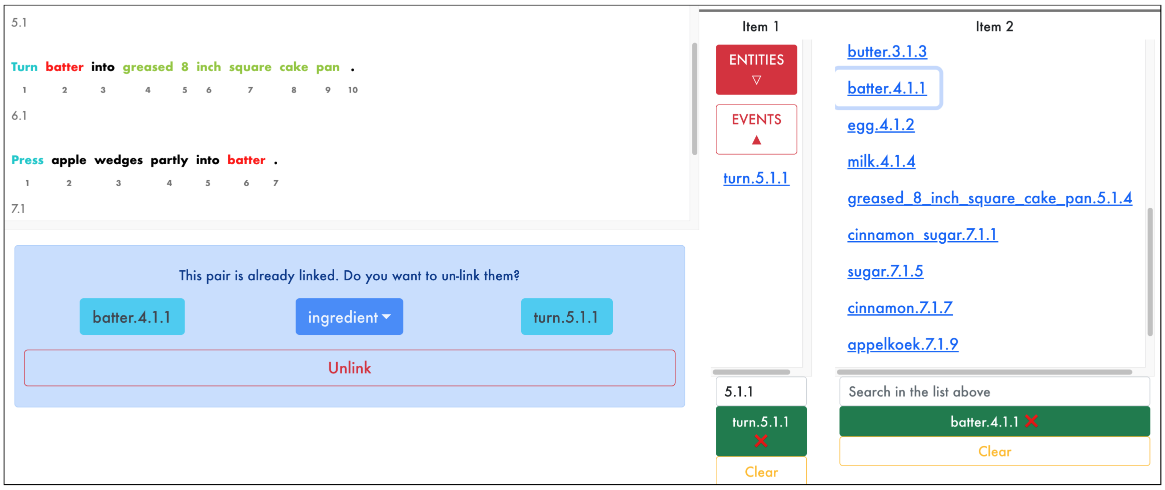
More specifically, event-entity links can be one of several possible link tags, which can be made between explicit spans of text or between an event and a hidden entity that does not explicitly appear in the recipe text. These relations are:

- **Ingredient**: identifies the food material that participates in cooking events.
- **Result**: identifies entities produced as the output of an event.
- **Tool**: relates objects with the events they are used in. Tools may appear in the text ("Cut the pear with a sharp knife"), or they may be hidden ("Cut an apple" requires an unmentioned knife).
- **Habitat**: links events with the objects in which they take place. Habitats may appear in the text ("Bake in a preheated oven"), or they may be hidden ("Saute the onion" requires an unmentioned pan).

Table 3 shows the statistics of cooking role annotation on the dataset. EVENT should always be explicit, while the other cooking roles can be either explicit text spans or hidden entities. We hired 8 student annotators for the CRL annotation work. All annotators were students at Brandeis University, ranging from undergraduate to master's level.

5.3 Semantic Role Labeling

Aside from the above-described annotation layer, which is tailored to highlight domain-specific events and entities, each step in the recipes featured in R2VQ is automatically tagged and manually validated according to the predicates and constituents identified at the Semantic Role Labeling (SRL) level, i.e., the task of identifying and labeling predicate-argument structures within a sentence (Gildea and Jurafsky, 2002). More specifically,


5.1

Turn batter into greased 8 inch square cake pan .
1 2 3 4 5 6 7 8 9 10

6.1

Press apple wedges partly into batter .
1 2 3 4 5 6 7

7.1

This pair is already linked. Do you want to un-link them?

batter.4.1.1 ingredient turn.5.1.1

Unlink

Figure 2: DEEP environment for CRL entity linking.

	Train		Dev		Test	
Avg. # of entities per recipe	Exp.	Hidden	Exp.	Hidden	Exp.	Hidden
EVENT	14.0	N/A	13.6	N/A	13.3	N/A
INGREDIENT	13.0	6.9	14.0	10.8	12.5	8.6
RESULT	0.2	1.5	0.2	1.4	0.3	1.7
TOOL	0.6	2.1	0.7	2.2	0.6	2.0
HABITAT	2.8	4.8	2.5	6.2	2.5	4.0

Table 3: Statistics of cooking role annotation on R2VQ.

each recipe step is semantically enriched by (i) identifying all its predicates, i.e., those words or multi-word expressions that denote an event or an action, (ii) assigning the most appropriate sense label to each identified predicate according to a predefined inventory, (iii) detecting all the arguments, i.e., the parts of the text that are semantically related to each predicate, and (iv) choosing the most fitting semantic role for each predicate-argument pair. Let's consider the example "John bakes potatoes". In this case, SRL consists of (i) identifying "bake" as a predicate, that is, something that denotes an action or an event; (ii) disambiguating the predicate, that is, assigning the most appropriate sense for "bakes" in this context; (iii) identifying the arguments of each predicate, that is, those parts of the text, "John" and "potatoes" that are semantically linked to "bakes"; and (iv) assigning a semantic role to each predicate-argument pair, e.g., "John" is the *Agent* of the predicate "bakes", whereas "potatoes" is the *Patient*.

In SRL, there are two main annotation formalisms for tagging arguments: span-based and dependency-based. We adopted the former; the core and only difference between the two lies in the fact that, in the span-based SRL, semantic role labels are applied to the whole span of a given argument, whereas, in dependency-based SRL, the label is only applied to the argument's head (e.g., we label "the broccoli" and not "the").

The SRL task is often tied to a linguistic resource, which defines the inventory of predicate senses and semantic roles. For this task, we chose VerbAtlas[5] (Di Fabio et al., 2019) as our inventory of predicate senses and semantic roles given its high coverage in terms of verbal lexicon[6], the informativeness of its human-readable roles (e.g., *Agent*, *Patient*, *Instrument*), and its mapping to the PropBank frame inventory (Palmer et al., 2005) and to the BabelNet multilingual knowledge base (Navigli and Ponzetto, 2012; Navigli et al., 2021).

The annotation process for the SRL layer featured three distinct stages. In detail, we first employed the Stanza toolkit (Qi et al., 2020) to perform PoS tagging over the R2VQ corpus so as to

[5]VerbAtlas is freely available for research purposes at http://verbatlas.org/.

[6]VerbAtlas covers all the verbal senses defined in WordNet, and clusters them into predicate frames.

identify verbal predicates, and proceeded to manually include predicates that were not discovered automatically (e.g., *season* was often incorrectly labeled as a noun), as well as fixing instances erroneously labeled as predicates (such as adjectival or prenominal predicates, as well as predicates appearing within ill-formed sentences).[7] Secondly, we employed a state-of-the-art system (Conia and Navigli, 2020) to automatically label recipes in a span-based fashion, concurrently assigning VerbAtlas frames and arguments to recipes, and manually validating the whole corpus once more in order to verify the automatically-generated outputs, fixing errors and inconsistencies.[8]

We used BabelNet 5.0 as the inventory to validate predicates, first picking the most suitable word sense to disambiguate a given verb, and then selecting the relative frame in VerbAtlas according to its original mapping. As our final step, we instructed annotators to manually tag as many arguments as possible for each predicate (adding arguments where needed and removing additional arguments such as *Negation* in the process), first, referring to the predicates' prototypical arguments according to VerbAtlas, and then, providing additional arguments. We used VerbNet (Schuler, 2006) argument descriptions and examples along with in-house argument descriptions for ambiguous argument assignments (e.g., "in the oven" in "Jennifer baked the potatoes IN THE OVEN" is not a *Agent*, but rather an *Instrument* with respect to to the predicate "bake").

With respect to the SRL layer annotators, in order to make use of the Mechanical Turk platform already employed in the context of the aligned image frame annotation, we initially devised HITs for both predicate sense disambiguation and argument labeling. Though, independently of the rates and templates employed, we kept collecting low-quality or suboptimal data, likely, due to the background knowledge needed to perform such tasks in an adequate fashion. In light of this, after several attempts, we eventually decided to have one in-house annotator with extensive experience in SRL validate the whole corpus at all stages required, and asked a second annotator to review the validation instances,

[7]We also labeled word forms with typos in the original recipes as predicates (e.g. *prehet* as *preheat*). Additionally, we labeled as multi-word predicates those predicates whose form was featured as a compound in BabelNet.

[8]See Appendix A for details about the SRL annotations' format.

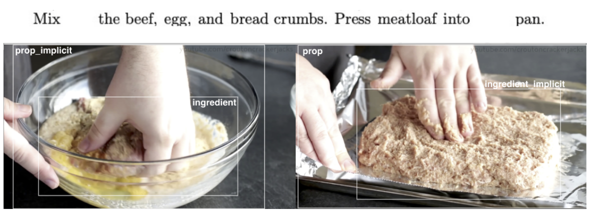


Figure 3: An aligned CRL-Image Frame annotation.

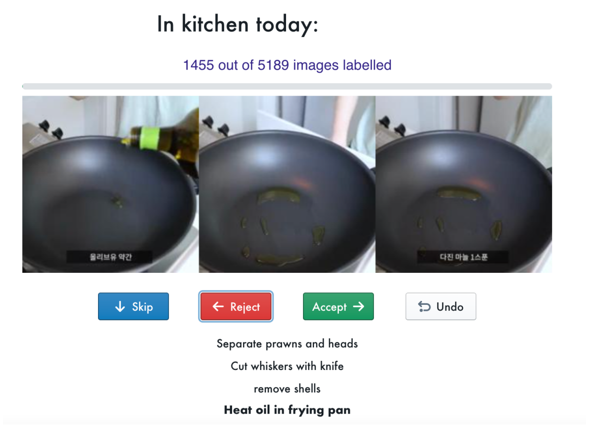
seeking agreement in case of discrepancies. As an additional step to ensure data quality, a third, external annotator was assigned with the task of reviewing recipes in order to look for potential formatting issues.[9]

5.4 Aligned Image Frame Annotation

Accompanying each recipe is a series of images extracted from YouTube videos that are associated with a particular event in the recipe. We pulled the images from a set of YouTube videos that were selected by querying YouTube for recipe titles. For each recipe title, we downloaded 5 Creative Commons licensed videos. These videos were indexed by generating an embedding using the Tensorflow implementation of the S3D Text-Video model trained on HowTo100M using MIL-NCE (Miech et al., 2020, 2019). For each cooking event in the recipes, the 5 closest clips as scored by L2 distance were selected from the YouTube videos we downloaded. We showed the annotators the first, middle and last frame from each 4 second clip alongside a list of the CRL representations of the events in the recipe. We asked the annotators to rank the match of the image and the cooking event as a good match, a partial match, or not a match. The Swipe Labeler (Peterson Jenessa, 2021) tool was used to conduct the annotation. The tool was modified to included the recipe event text, with the full recipe displayed and the current step in bold text. An example of the frame annotation is shown in Figure 4.

Due to complex combinations of ingredients in many of the recipes and the limitation of considering only Creative Commons videos, many events did not match with any of the detected segments. Partial matches were included in order to increase the total number of events represented. Importantly, the action represented in the image clips does nec-

[9]All annotators employed in the SRL layer have effective operational proficiency in English and received a wage in line with their country of residence. Annotation has been carried out by means of user-friendly shared worksheets.



Separate prawns and heads
Cut whiskers with knife
remove shells
Heat oil in frying pan

Figure 4: Swipe Labeler Annotation Tool

essarily include the exact same ingredients of as those used in the recipe. The videos were chosen based on the similarity of the cooking event described in the sentence. In total, 1927 events were matched with images across 655 recipes.

5.5 Generating Competence-based Questions

We first design text templates for each type of question. Then we generate QA pairs by populating the templates in a cloze test style with the data annotated in CRL. Table 4 shows the text templates for two types of questions we want to use for the QA task. ELISION identifies arguments (ingredients in most cases) that are omitted from a text, but can be understood from context. IMPLICIT covers both implicit tools and habitats introduced in the text. This is distinct from ELISION, as these are not solved merely through contextual clues. Each text template has several slots that can be filled with corresponding entities from CRL.

To increase the variety of questions, we also include adjunct slots into the templates. As shown in Table 4, adjunct slots include tool or habitat phrases and SRL modifiers. SRL modifiers are any semantic roles that are not claimed by CRL entities such as `TIME` and `VALUE`. For example, one ELISION question can be as short as *What should be cut?* or *What should be cut on the board with a knife into eighths?* with all the adjunct slots. We argue that it is helpful to generate questions more challenging to the systems. Adding more adjunct slots completes the context for the question, but also introduces unseen context if the slots contain hidden entities.

These slotted templates are further processed to improve the readability of generated questions. We change word inflections and insert articles and agreements. For the templates with [habitat_phrase] and [tool_phrase] slots, we fill those with corresponding `LOCATION` or `INSTRUMENT` spans from SRL. If a slot is filled with a hidden entity that has no associated semantic roles, we run a BERT-based model (Devlin et al., 2019) to get the most likely preposition given the sentence as context through the masked language modeling task. SRL modifiers are populated in the same order as they were in the original sentence.

5.6 Details of copyright

All recipes are distributed under Creative Commons license. The YouTube videos queried were limited to Creative Commons videos only. No personally identifying information is included in either the text or visual components of the dataset.

6 Participation

We discuss the baseline system and the systems from participants in this section.

6.1 Evaluation Metrics

All systems are asked to provide answers to the open-ended questions based on the textual and visual information encoded in the dataset. The results are evaluated using exact match (EM) and token-level F1 score (F1) following Rajpurkar et al. (2018).

6.2 Baselines

To build a model that is reflective of the nature of the abstractive question answering task and benefits from the aligned key frames to the text, we adopt a vision-and-language text genertation model as the baseline for our task. We build the baseline with the model framework that is proposed by Cho et al. (2021). They propose the model VL-T5 based on T5 text generation model (Raffel et al., 2020) by extending the original T5 text encoder to a multimodal encoder that can take both textual and visual embeddings as the input.

Following closely the VL-T5 work (Cho et al., 2021), we prepare the key frames as model input by encoding them into visual embeddings using Faster-R-CNN. We prepare the text input by appending the task-specific prefix to the question and context text: `"question: {question_str} context: {recipe_str}"`. The `recipe_str` is the concatenation of the text of all cooking steps from the recipe the question is generated from. We fine-tune the VL-T5 model for our QA task on the

QUESTION TYPE	TEXT TEMPLATE	QUESTION-ANSWER PAIR
Elision	What should be verb [habitat_phrase] [tool_phrase] [modifiers]? — ingredient_obj	What should be cut on the board with a knife into eighths? — apples
Implicit	What do you use to verb obj [habitat_phrase] [modifiers]? — tool Where do you verb obj [tool_phrase] [modifiers]? — habitat_phrase	What do you use to sauté the onions [in the pan]? — spatula Where do you arrange the slices [into rounds]? — in the casserole

Table 4: Text templates and example of generated questions. The squared brackets ([...]) in the templates indicates adjunct slots.

training set, and run the fine-tuned model on the test set. As a comparison, we also fine-tuned the T5 model with text input only. Baseline results are shown in Table 5 along with other results from participants.

	EM	F1	Key Frames?
SRPOL	92.53	94.34	
ITNLP&QMUL	91.33	94.23	
PINGAN_AI	78.21	82.62	
Slug	69.49	77.37	
BASELINE (VL-T5)	69.37	77.77	✔
BASELINE (T5)	65.34	75.22	
ych	10.23	10.23	
UoR	5.90	15.78	✔
CLT6	0.0	0.0	

Table 5: Task results from participant teams and the baseline. The ranking is based on EM score. The last column indicates whether the system uses key frames for training.

6.3 Description of team submissions

We collect successful submissions from 8 participating teams (including the baseline), as well as one participating team that did not submit predictions that passed our automated evaluation script. The results and final ranking are shown in Table 5. We summarize their work below:

- **SRPOL**: This system attains the highest scores in this task by adopting a hybrid approach. The system includes a rule-based system for intent identification and finding N/A questions. It also applies a transformer-based model ELECTRA for generating extractive answers.

- **ITNLP&QMUL**: This system attains the second highest scores in this task. The system adapts a T5 model to the task by altering the input to include semantic and cooking role labels that are provided in the data.

- **PINGAN_AI**: This system attains the third highest scores in this task. The system uses the BERT model as the backbone, and enhances the model by incorporating additional knowledge about cooking entities and part-of-speech tags in the format of plain text and embeddings.

- **Slug**: Semantic labels were preprocessed using BERT and handmade rules, with hidden roles infused into the recipe. A task-finetuned T5 model was then used for question answering.

- **UoR**: The only submission that exploited the visual information provided in the dataset, this system used an Inception V3 model (pre-trained on ImageNet), to extract image features that were used to train an image captioning model on the MS-COCO dataset. These captions were included alongside the recipe text in a Retrieval-Augmented Generation model for question answering.

7 Discussion

In this paper we have described the new task of Competence-based Multimodal Question Answering. In this task, we extended the traditional question answering by providing text-visual aligned data as the context, and asking questions that reflects reasoning competences over the question context. To create the dataset for our task, we proposed and applied a rich annotation of semantic role labels, cooking role labels and aligned video key frames to a set of cooking recipes.

A criticism of the approach we adopted to create annotated dataset is that the video key frames are not well aligned with the text, thus making it difficult to include those into the modeling training. Although with the full awareness of this, video annotation and alignment remains a very difficult task. Copyright issues also make it challenging for us to get enough video sources to work with. Future work to improve the key frame annotation may include utilizing entity recognition so that more accurate alignment to text can be made. We will also consider reusing the key frames and adding static images to represent similar events from different recipes to increase the coverage of annotation. Another criticism of the data is the semantic ambiguity and loose definition of certain questions. For example, the same How-to question can have multiple reasonable answers, but only one is considered as the gold answer. Although this is the semantic ambiguity as it is, we intend to improve it by replacing the question phrase "How to ..." to more specific phrase like "What tool ..." based on the answer it is inquiring about.

An analysis of the systems that participated in our task showed the major improvement over the evaluation scores is achieved by making the hidden information appear on the surface. In general, two approaches are proven to be useful for this purpose by the participating systems. One is to train an end-to-end system to generate text that contains CRL-SRL annotation, so that the hidden information is expressed explicitly in the generated text. Then an extractive QA system can be adopted to identify text spans as answers. The second approach involves rules and heuristics to identify question intents, and get auxiliary knowledge. Intent identification can help classify questions into different categories. Each question category is associated with a rather fixed set of answer templates and possible entity types to be filled in. Auxiliary knowledge is generated by associating specific entities with their co-referred mentions or result ingredients (e.g. "small balls" to "flour mixture").

The analysis of the results from participating systems also reveals some interesting characteristics about the dataset and is useful for future task design. Despite the error rate of the top-performing systems such as SRPOL and ITNLP&QMUL is only 8%, the cardinality questions and How-to questions solely contribute the majority of the errors. As it is mentioned above, the innate ambiguity of How-to questions makes it difficult for both humans and systems to get a single correct answer. The poor performance on cardinality questions shows that the "counting reasoning" remains a big challenge to current transformer-based systems. In the R2VQ dataset specifically, the mentions of the entity involved a cardinality question can scatter over the whole recipe, which requires a larger context to answer such questions. Due to nature of "constant ingredient transformation" in cooking recipes, the mentions of the same entity could vary in our definition. For example, in the appelkoek recipe (Table 1), *apples*, *peeled apples*, *apple wedges*, *apples with batter* all refer to the same entity *Apple*. This characteristic of cardinality questions also hinders the systems from counting the mentions of the entity properly.

The human benchmark created by the SRPOL team provides useful insights on our future QA task design. They asked six linguists to answer $2,000$ questions selected randomly from the validation set. By examining the manual annotation on the questions, they found that although 73% of the annotated QA pairs have the same meaning as the gold answers, the EM score is quite low. This reveals the fact that traditional QA metrics that focus on string match might be too strict in our task. For example, from the analysis of the human benchmark, for the question *What's in the mixture?*, both the gold answer *the egg and mixture* and the human answer *the butter, sugar, tangerine zest, vanilla, baking powder, salt and egg* can be considered correct. Other metrics like BERTScore (Zhang et al., 2019) might be a good compliment to account for the syntactic and semantic variance between the model inference and the gold answer.

8 Conclusion

In this paper we described *SemEval-2022 Task 9: R2VQ – Competence-based Multimodal Question Answering*. The task is to answer questions from a collection of cooking recipes and videos, where each question belongs to a "question family" reflecting a specific reasoning competence. We developed a new dataset of cooking recipes with rich annotation for cooking roles, semantic roles and aligned video key frames. We collected 8 result submissions and analyzed the participating systems by highlighting and summarizing their findings to help future research pertaining the topic of our task.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and

Devi Parikh. 2015. VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Nicholas Asher. 2011. *Lexical meaning in context: A web of words*. Cambridge University Press.

Gregory A Bechtel, Ruth Davidhizar, and Martha J Bradshaw. 1999. Problem-based learning in a competency-based world. *Nurse Education Today*, 19(3):182–187.

Luisa Bentivogli, Ido Dagan, and Bernardo Magnini. 2017. The recognizing textual entailment challenges: Datasets and methodologies. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, pages 1119–1147. Springer.

Susan Windisch Brown, James Pustejovsky, Annie Zaenen, and Martha Palmer. 2018. Integrating generative lexicon event structures into verbnet. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jaemin Cho, Jie Lei, Haochen Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. *ArXiv*, abs/2102.02779.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*, volume 11. MIT Press.

Seung Youn Chyung, Donald Stepich, and David Cox. 2006. Building a competency-based curriculum architecture to educate 21st-century business practitioners. *Journal of Education for Business*, 81(6):307–314.

Simone Conia and Roberto Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1396–1410, Barcelona, Spain (Online).

Rajarshi Das, Tsendsuren Munkhdalai, Xingdi Yuan, Adam Trischler, and Andrew McCallum. 2018. Building dynamic knowledge graphs from text using machine reading comprehension. In *International Conference on Learning Representations*.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Kaustubh D. Dhole and Christopher D. Manning. 2021. Syn-qg: Syntactic and shallow semantic rules for question generation.

Andrea Di Fabio, Simone Conia, and Roberto Navigli. 2019. VerbAtlas: a novel large-scale verbal semantic resource and its application to semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 627–637, Hong Kong, China.

Jean-Paul Doignon and Jean-Claude Falmagne. 1985. Spaces for the assessment of knowledge. *International journal of man-machine studies*, 23(2):175–196.

Dirk Geeraerts. 2009. *Theories of lexical semantics*. OUP Oxford.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Jürgen Heller, Thomas Augustin, Cord Hockemeyer, Luca Stefanutti, and Dietrich Albert. 2013. Recent developments in competence-based knowledge space theory. In *Knowledge spaces*, pages 243–286. Springer.

Cheng-Ting Hsiao, Fremen ChihChen Chou, Chih-Cheng Hsieh, Li Chun Chang, and Chih-Ming Hsu. 2020. Developing a competency-based learning and assessment system for residency training: analysis study of user requirements and acceptance. *Journal of medical Internet research*, 22(4):e15655.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910.

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. 2015. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 982–992.

Najoung Kim and Tal Linzen. 2020. Cogs: A compositional generalization challenge based on semantic interpretation. In *EMNLP*.

H. Levesque, E. Davis, and L. Morgenstern. 2011. The winograd schema challenge. In *KR*.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *ACL*.

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

Jonathan Malmaud, Earl Wagner, Nancy Chang, and Kevin Murphy. 2014. Cooking with semantics. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, pages 33–38.

Diego Marconi. 1997. *Lexical competence*. MIT press.

Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. 2020. End-to-End Learning of Visual Representations from Uncurated Instructional Videos. In *CVPR*.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning based text classification: A comprehensive review.

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

Roberto Navigli, Michele Bevilacqua, Simone Conia, Dario Montagnini, and Francesco Cecconi. 2021. Ten years of BabelNet: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4559–4567. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial intelligence*, 193:217–250.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Weerasinghege Udara Peterson Jenessa, Ramesh Sumanth. 2021. Swipe-labeler.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M Mitchell. 2019. Competence-based curriculum learning for neural machine translation. *arXiv preprint arXiv:1903.09848*.

Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W Black. 2020. Topological sort for sentence ordering. *arXiv preprint arXiv:2005.00432*.

James Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge, MA.

Valentina Pyatkin, Paul Roit, Julian Michael, Reut Tsarfaty, Yoav Goldberg, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

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

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *ArXiv*, abs/1806.03822.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Anna Rogers. 2019. How the transformers broke nlp leaderboards.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension.

Karin Kipper Schuler. 2006. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and A. Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. *ArXiv*, abs/1911.09241.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 57–66.

Richard A Voorhees. 2001. Competency-based learning models: A necessary future. *New directions for institutional research*, 2001(110):5–13.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP@EMNLP*.

Z. Yang, Zihang Dai, Yiming Yang, J. Carbonell, R. Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. 2016. Visual7W: Grounded question answering in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4995–5004.

A Reading SRL Annotations in R2VQ

Predicate frames. Each predicate is labeled according to its VerbAtlas sense/frame. A value of "_" means that the corresponding word is not a predicate.

In the example below, there is only one predicate, "Cut" with the corresponding sense/frame "CUT" in position 1.

1	Cut	[...]	CUT	B-V
2	the	[...]	_	B-Patient
3	broccoli	[...]	_	I-Patient
4	into	[...]	_	B-Result
5	flowerets	[...]	_	I-Result
6	.	[...]	_	_

Semantic roles. For each predicate, we provide its semantic roles in BIO format (B - Beginning, I - Inside, O - Outside). Note that, for this dataset, we only use B and I to indicate the first token of a span and the rest of the tokens in the same span, respectively. In the example above, "the broccoli" is a *Patient* of the predicate CUT, with the token "the" as the Beginning of the span (B-Patient) and the token "broccoli" as the Inside of the span (I-Patient). Note that the predicate that refers to a specific column of semantic roles is always labeled with the notation B-V. Should the predicate consist of a multi-word expression, the other tokens apart from the first are labeled as I-V:

1	Deep	[...]	COOK	B-V
2	-	[...]	_	I-V
3	fry	[...]	_	I-V
4	till	[...]	_	B-Result
5	crispy	[...]	_	I-Result
6	&	[...]	_	I-Result
7	golden	[...]	_	I-Result
8	brown	[...]	_	I-Result

Should the multi-word expression be made of non-adjacent words, tokens apart from the first are instead labeled as D-V:

1	Bring	[...]	CHANGE_APP./STATE	B-V
2	the	[...]	_	B-Patient
3	water	[...]	_	I-Patient
4	to	[...]	_	D-V
5	boil	[...]	_	D-V
6	.	[...]	_	_

In the case of multiple predicates in the same sentence, there will be multiple semantic role columns, one for each predicate. For example, if there are two predicates in the sentence, one column will indicate the semantic roles for the first predicate, and the following will show the semantic roles for the second predicate.

1	Reduce	[...]	REDUCE_D.	B-V	_
2	heat	[...]	_	B-Attr.	_
3	,	[...]	_	_	_
4	and	[...]	_	_	_
5	simmer	[...]	COOK	_	B-V
6	for	[...]	_	_	B-Time
7	1	[...]	_	_	I-Time
8	hour	[...]	_	_	I-Time
9	.	[...]	_	_	_

HIT&QMUL at SemEval-2022 Task 9: Label-Enclosed Generative Question Answering (LEG-QA)

Weihe Zhai*[1,2], Mingqiang Feng*[1], Arkaitz Zubiaga[2], Bingquan Liu†[1]

[1]Harbin Institute of Technology, China
[2]Queen Mary University of London, UK
{weihezhai, mqfeng}@insun.hit.edu.cn
a.zubiaga@qmul.ac.uk
liubq@hit.edu.cn

Abstract

This paper presents the second place system for the R2VQ: competence-based multimodal question answering shared task. The task consisted in building question answering systems that could process procedural recipes involving both text and image, and enriched with semantic and cooking roles. We tackled the task by using a text-to-text generative model based on the transformer architecture, with the aim of generalising across different question types. Our proposed architecture incorporates a novel approach for enriching input texts by incorporating semantic and cooking role labels through what we call Label-Enclosed Generative Question Answering (LEG-QA). Our model achieves a score of 91.3, with a significant improvement over the baseline (65.34) and close to the top-ranked system ((92.5). After describing the submitted system, we analyse the impact of the different components of LEG-QA as well as perform an error analysis.

1 Introduction

The objective of text-and-image multimodal question answering (QA) is to jointly leverage both textual and visual information to mutually inform each other for semantic reasoning (Ben-Younes et al., 2017). A SemEval 2022 shared task titled Competence-based Multimodal Question Answering (R2VQ) focuses on this task. In the R2VQ task, participants were invited to develop QA models to resolve questions associated with procedural recipe instructions. The corpus provided for the task is made of recipes, which include rich semantic annotations and where textual instructions are aligned with images illustrating them. The authors proposed to address a set of 18 question families, for which participants could develop and evaluate their own proposed solutions.

This paper describes the participation of the HIT&QMUL team in the R2VQ task, for which we proposed a methodology that we call label-enclosed generative question answering (**LEG-QA**). Through this methodology, we proposed enclosing labels providing semantic information embedded in the input texts. This methodology has proven competitive by achieving a score of 91.3% in exact match accuracy, ranking 2nd overall in the competition.

Our code is available at `https://github.com/weihezhai/HIT-QMUL-at-SemEval-2022-Task-9`.

2 Task and System Description

Building on the transformer architecture (Vaswani et al., 2017), we use T5 an encoder-decoder model (Raffel et al., 2020) implemented using Hugging Face[1]. We chose T5 given its reasonably good general language learning abilities, and provided that the downstream task of R2VQ covers a diverse set of task types that are also shared by T5. Intuitively, a task-agnostic model like T5 would be expected to perform well on R2VQ. We further adapt the T5 model to the task by altering the input to include semantic and cooking role labels enclosed in the textual recipe instructions.

2.1 Task and Data

The R2VQ (Tu et al., 2022) task proposed the use of multimodal models to leverage both text and image for QA in the context of recipes. The R2VQ task adopts the definition of 'Question Family' from the CLEVR dataset (Johnson et al., 2017), where each type of question-answer pair comes from a template identified by task organisers. Each of these question type is meant to evaluate a different ability for reasoning.

The R2VQ dataset consists of a collection of 1,000 recipes, of which 800 are used for training

*contribute equally
†corresponding author

[1]`https://huggingface.co/`

and two sets of 100 recipes are used for validation and testing. These recipes involve more than 30K question-answer pairs, where each recipe consists of texts with procedural instructions as well as associated images. Questions are intended to require both visual and textual information jointly to produce effective answers. However, based on our initial explorations of the dataset provided and after considering the use of multimodal models, we found that the majority of the questions could be answered through the sole use of text. Hence, we thought that the use of an image processing component could be avoided while producing an accurate answer generation model for the dataset at hand. Further improvement of the model through the use of an image processing component is therefore left for future work.

For in-depth analysis of the system results, we grouped the 11 question types[2] provided by the organisers into 4 categories, i.e. generative, number reasoning, classification and extractive (see Table 5 in the Appendix for category details). Note that each of these classes comprises questions intended to evaluate different abilities of models.

2.2 Soft Constrained Generative QA as Multitask Transfer Learning

Soft Constrained Text-to-Text Generation Mainstream text-to-text generation methods mostly aim to learn meaningful mappings between input and output sequences. This is particularly the case for the recent pre-trained language models (Lewis et al., 2020; Raffel et al., 2020), where the model is expected to identify what to attend to in the source input and what to include in the model output. Models like UniLM, T5 and GPT2 unify the generation and understanding tasks within a single model, but none of them investigates the model's ability of generating free-form answers which includes both generative and discriminative tasks.

In R2VQ, all questions are created through a semi-automated method. Generative questions such as implicit argument identification (e.g. how do you drain the pasta?) cannot be answered independently by an extractive question answering approach, hence some question types should be treated as a soft constrained generation (SCGen) problem (See et al., 2017; Dou et al., 2021). SCGen implicitly specifies token constraints that the model needs to focus on in the answer output. For instance, in the question example above, one of the following words must be present in the answer: "by", "using" (gerund of a verb), "in/on/at", "with". There are many variants of models considering soft constraints, but usually they are achieved by adding an attention mechanism to the source keywords (Yao et al., 2019).

[2]To be clear, 11 types of questions are defined on the R2VQ homepage but not the types of question-id in the dataset.

Multitask Transfer Learning Multitask learning consists in training the model on multiple tasks at a time. This means the model has an objective of simultaneously taking on more than one task. In LEG-QA, we leverage and transfer the prior knowledge from T5 and apply to our R2VQ tasks to solve the unseen SCGen problems. The reason why we do not train separate models for each of 4 categories in Table 5 is that we observe non-negligible accuracy drop when missing some of the question families.

2.3 Label-Enclosed Input

The key modification we made on the T5 model to prepare our submitted system is a rarely adopted heuristic method for embedding label information. Instead of appending supportive external features to the text sequence (He et al., 2017, 2018), we fuse the hidden cooking entities into the original text through our proposed approach called **label-enclosed input**. Typically, to grammatically and syntactically maintain the textual structure, the text should not be broken down into pieces by inserting annotations. However, we noticed that with the knowledge of labels appearing in the enclosed form, a pretrained text-to-text multitask model (in our case T5) can effectively process the enclosed noise from external information. In turn, the model using this enriched input behaves better than using a clean text input, when generatively answering questions with soft constraints.

2.4 Input Format

Figure 1 depicts the pipeline for the input data preprocessing through which the attributes of cooking roles are transformed and enclosed into the input sequences. We employed different processing approaches for each type of semantic and cooking roles. Through close observation, the most frequent attributes that take place in the answers are the 'Hidden' labels which consist of multiple values and keywords. As shown in the example, after text regularisation and reorganisation, attributes

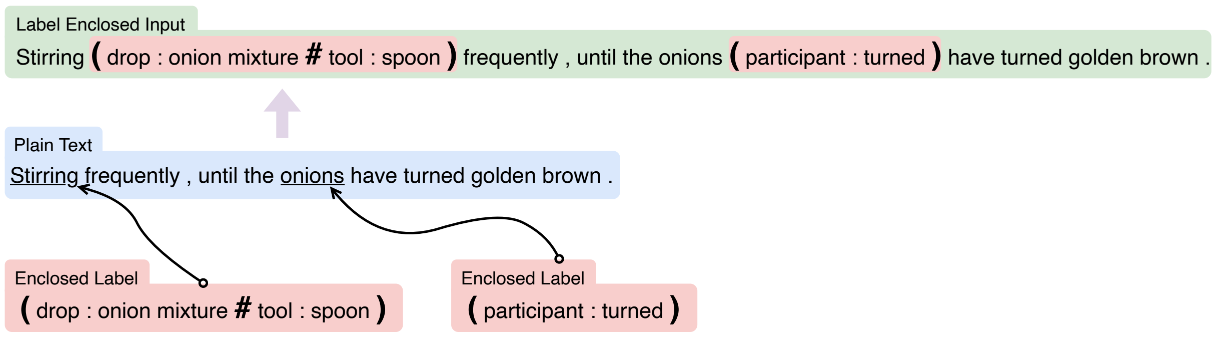

Label Enclosed Input
Stirring (drop : onion mixture # tool : spoon) frequently , until the onions (participant : turned) have turned golden brown .
Plain Text
Stirring frequently , until the onions have turned golden brown .
Enclosed Label
(drop : onion mixture # tool : spoon)
Enclosed Label
(participant : turned)

Figure 1: Pipeline for creating label-enclosed inputs. In this case, labels are wrapped in braces, and labels for the same event but are of different categories are separated by hashtag.

(e.g. stirring) are enclosed in special tokens before embedding them into the input. While the example shows the use of brackets for enclosing labels, we tested different approaches, which we discuss in more detail in Section 3.

3 Experiments

Since different formats of enclosing special tokens can result in considerably different scores, we compare a range of experiment settings to evaluate their performance. For more detail on the different enclosing methods tested, see Table 4 in Appendix A.

We conduct a series of experiments to answer the following questions:

Q1: Which way of enclosing special tokens performs best? Is there any big difference between them, and any measurable explanation for the gap?

Q2: Which label combos achieve the best accuracy?

Q3: How significant is the difference between results generated by size-equivalent models?

3.1 Experimental Settings

For the purpose of, as much as possible, controlling variables unrelated to the pre-trained language model, we experiment with five input variants on T5-large (our suboptimal result model). For comparison, we also show results for models based on different architectures. In this case, we use BART(Lewis et al., 2020), a denoising text-to-text model.

All the performance scores we show indicate the exact match (EM) accuracy on the development data.

	Enclosing Method	GEN	CLS	NUM	EXT	Overall
T5-Large	brackets	87.3	97.0	**82.6**	93.4	89.9
	hash	86.9	**97.4**	80.6	93.6	89.5
	dollar sign	**88.2**	96.9	81.0	93.8	**90.2**
	[BOL] [EOL]	87.0	97.2	81.0	93.4	89.5
	parallel	81.2	97.0	79.7	**94.1**	86.5

Table 1: Detailed results for different enclosing methods. The default method is "( )" which is used for our final submission. The dollar sign enclosing method is evaluated after the final submission so is not reported as the best practice during the competition.

3.2 Comparative study

To answer Q1, we report EM score for every 2K steps up to a total 26K steps. This is equivalent to approximately 10 epochs. Table 1 summarises scores achieved by the T5-Large model with the 5 different label-enclosing methods. The *Hidden, Part and Event* labels are examined together, given for that our best-performing model so far follows this paradigm. The special token "$" dollar sign has a noticeable positive impact on the GEN task, leading to the overall best performance. The "Parallel" enclosing method refers to directly attaching all labels horizontally aligned with plain text. Additionally, the [BOL] and [EOL] are special symbols inherent to BERT-like models. Unsurprisingly, joining labels in parallel with text without breaking sentence syntax helps achieve better EXT score. The reason for ultimately choosing "( )" is that we believe that its directional attribute could help the model parsing the label structure to some extent. A further combination comparison is discussed when answering Q2.

To answer Q2, we carry out label combination experiments to compare the effectiveness of different roles when contributing to the 4 question categories. Table 2 analyses the benefit of gradually

	Labels Combo	GEN	CLS	NUM	EXT	Overall
T5-Large	HIDDEN	86.5	**97.2**	81.7	94.3	89.5
	HIDDEN + PART	87.2	**97.2**	80.4	**94.6**	89.8
	HIDDEN + PART + EVENT	**87.3**	97.0	**82.6**	93.4	**89.9**
	Text Only	46.8	93.8	66.4	93.9	66.7

Table 2: Scores achieved with ensembles of selected labels, each of which is picked out as a result of benefiting certain types of questions.

appending additional roles to each identified entity, i.e. adding more extrinsic information which frequently matches with answers. Note that the *Hidden* label is the dominant among participants, as it appears approximately in all generative questions, and compared with the text-only method, there is a noticeable improvement after adding *Hidden* to it. First, introducing more applicable labels has a positive impact on the overall accuracy. We observe the triplet leads to overall best performance. Second, plain text behaves consistently to what we find in answering Q1, which performs worse in tasks like generative questions, however achieving strong performance in the extractive questions.

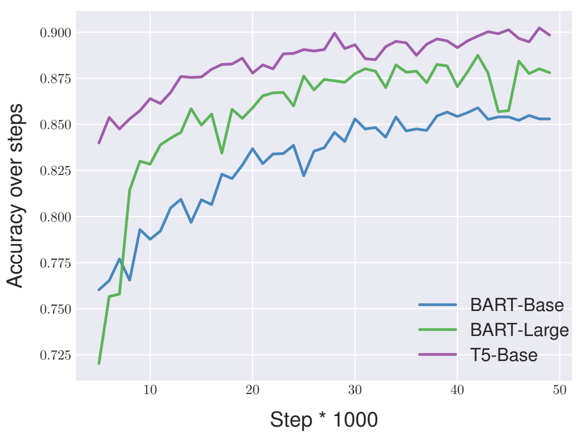


Figure 2: Models accuracy on validation dataset report every 1k step for three representative models.

To answer Q3, the tested models need to be comparable. We could say if two models (1) are roughly of the same amount of parameters (i.e. the same size). (2) share similar architectural design methodology (e.g. transformer-based), they would be considered comparable. We list the performance comparison of five models for 26k training steps in Table 3. We see that BART-Large is on a par with T5-Base from many aspects but performs poorly on a number of reasoning and generative questions. This is likely because T5 is fine-tuned on a more diverse mixture of tasks along with a very large pre-training dataset. Figure 2 demonstrates that BART-Large performs poorly at the beginning, taking off at around the 10k steps i.e. the second epoch. Moreover, later in the training stage, T5-Base has a slight advantage in achieving low variance. By contrast, the performance of BART-Large drops.

		Model	GEN	CLS	NUM	EXT	Overall
Enclosed by ( )	HD + PT + ET	BART-Base	82.9	94.2	74.7	92.7	86.0
		BART-Large (850 MB)	85.6	96.8	78.5	93.0	88.4
		T5-Base (950 MB)	87.0	96.4	81.7	**93.9**	89.6
		T5-Large	87.3	97.0	**82.6**	93.4	89.9
		T5-3B *	**88.6**	**97.4**	81.5	93.6	**90.5**

Table 3: Models comparison between T5 and BART of multiple sizes, result of T5-3B without error correction is used as our final prediction model. Note that T5-3B is 4 times the size of T5-Large. HD, PT and ET are short for HIDDEN, PART and ENTITY labels. Note that those labels are defined by the R2VQ dataset.

3.3 Error Analysis

Table 6 gives detailed examples including denoting mistakes of our suboptimal models i.e. T5-Large, which is slightly worse than our best T5-3B. As shown in questions 1 and 3, the model has a tendency to include more unrelated label information when answering generative questions. However, in question 4, it ignores some non-label important ingredients. That is in part because the transformer's attention mechanism sometimes fails at choosing whether to attend to labels or not when filling the answer template. Additionally, the worst performnace is for the 'number reasoning' question type, which is very challenging given that it needs to pay attention to multiple labels in combination. This could possibly be improved by further re-designing the transformer block or by including a memory block over the context.

3.4 Best-performing Submission

Even though we submitted multiple systems throughout the evaluation phase, our best-performing submission is the model that uses T5-3B and integrates the label-enclosing approach based on round brackets "( )". This system achieved an overall 91.3 in the test set, attaining the 2nd position in the competition.

4 Conclusion

In this paper, we describe the participation of the HIT&QMUL team in the R2VQ shared task, where we ranked second. Our model is based on a unified generative text-to-text approach, in which we propose a novel label-enclosed input technique to include annotation labels to include semantic and cooking role labels. Our model achieved an exact match accuracy of 91.3, well over the baseline model (65.3) and only slightly behind the top-ranked system (92.53). Table 7 lists the top five final results on the R2VQ test set from all user submissions ordered by Exact Match score.

Through our comparative study, "$" enclosed labels proved to be best, with the most effective generative answering ability. A combination of HIDDEN, PART and ENTITY provides the best set of labels. Our study of the label-enclosing approach has some limitations given our focus on a small number of experimental label combinations. In future work, more analysis can be conducted exploring other label combinations potentially leading to further improved performance. In addition, the error analysis reveals that the model sometimes lacks the ability to attend to related labels possibly due to attention decay. Deeper investigation of this is left for future work.

References

Hedi Ben-Younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. 2017. Mutan: Multimodal tucker fusion for visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2612–2620.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and whats next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910.

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

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

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

Jingxuan Tu, Eben Holderness, Marco Maru, Simone Conia, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Roberto Navigli, and James Pustejovsky. 2022. Semeval-2022 task 9: R2VQ – competence-based multimodal question answering. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

Appendix

A Label-enclosed Example

Enclosing Method	Example
( )	Stirring (drop : onion mixture # tool : spoon) frequently, until the onions (participant : turned) have turned golden brown.
#	Stirring # drop : onion mixture # tool : spoon # frequently , until the onions # participant : turned # have turned golden brown .
$	Stirring $ drop : onion mixture # tool : spoon $ frequently, until the onions $ participant : turned $ have turned golden brown.
[BOL] [EOL]	Stirring [BOL] drop : onion mixture # tool : spoon [EOL] frequently, until the onions [BOL] participant : turned [EOL] have turned golden brown.
Parallel	Stirring frequently, until the onions have turned golden brown. [space] drop : onion mixture tool : spoon [space] [space] [space] [space] [space] participant : turned [space] [space] [space] [space] [space]

Table 4

B Question Family Categorisation

Catagory	Question Family
Number Reasoning (NUM)	Cardinality
Classification (CLS)	Event Ordering, Unanswerable
Generative (GEN)	Implicit Argument Identification, Ellipsis, Object Lifespan
Extractive (EXT)	Coreference Location Change, Attribute, Temporal, Result, Cause, Co-Patient

Table 5

C Error Example

Context
1. Mash eggs & mix with salad cream or mayonnaise. If you prefer a sweeter taste, go with salad cream. I like mine with mayo. 2. Clean & devein prawns; separate the heads. Blanch prawns & heads. 3. Drain & transfer to ice water to prevent them from over-cooking. Dice prawns & set aside. 4. Remove apple & mango skin & dice fruits into small cubes. Soak apple in salt water, lemon juice or cider vinegar to prevent it from browning. I usually use the traditional method of soaking in salt water handed down by my mom. Add them into egg mixture, together with the diced prawns. 5. Remove excess sauce from beancurd skin & stuff mixture into the pockets. You may have to cut & adjust the pocket height. 6. Cut up watermelon & start plating your dish. Chill your appetiser & youre ready impress your guests with this Inari Age Laughing Prawns Salad.

Question	T5-Large	Gold Answer
1. How did you get the mixture?	by adding the diced prawns, apple, mango and prawns to the bowl	by adding the apple, mango and prawns to the bowl
2. How do you soak apple to prevent it from browning?	soak apple in salt water, lemon juice or cider vinegar	soak apple in salt water , lemon juice or cider vinegar
3. What's in the inari prawn salad?	the pocket height and pockets	the pockets
4. What should be diced?	the apple mango	the fruits, apple and mango
5. How many times is the pot used?	2	3

Table 6

D Leader Board

Username	EM	F1
t.dryjanski	92.53	94.34
weihezhai 🐇	91.34	94.23
ruan	78.21	82.62
kartikaggarwal98	69.49	77.37
r2vq (baseline from organizers)	65.34	75.22

Table 7

Samsung Research Poland (SRPOL) at SemEval-2022 Task 9: Hybrid Question Answering Using Semantic Roles

Tomasz Dryjański[1], Monika Zaleska[1], Bartłomiej Kuźma[1], Artur Błażejewski[1], Zuzanna Bordzicka[1], Klaudia Firląg[1], Christian Goltz[1], Maciej Grabowski[1], Jakub Jończyk[1], Grzegorz Kłosiński[1], Natalia Paszkiewicz[1], Bartłomiej Paziewski[1], Jarosław Piersa[1], Paweł Bujnowski[1], Piotr Andruszkiewicz[1,2]

{t.dryjanski; m.zaleska; b.kuzma; a.blazejewsk; z.bordzicka; k.firlag; c.goltz; m.grabowski2; j.jonczyk; g.klosinski; n.paszkiewic; b.paziewski; j.piersa; p.bujnowski; p.andruszki2}@samsung.com

[1] Samsung Research Poland, Warsaw

[2] Warsaw University of Technology, Warsaw

Abstract

In this work we present an overview of our winning system for the R2VQ — Competence-based Multimodal Question Answering task, with the final exact match score of 92.53%. The task is structured as question-answer pairs, querying how well a system is capable of competence-based comprehension of recipes. We propose a hybrid of a rule-based system, Question Answering Transformer, and a neural classifier for N/A answers recognition. The rule-based system focuses on intent identification, data extraction and response generation.

1 Introduction

The goal of the task[1] was to develop a system applying existing knowledge to new situations, demonstrating a kind of understanding of a real-world domain. The competition presents a QA[2] challenge requiring linguistic and cognitive competencies that humans have while speaking and reasoning (Tu et al., 2022).

The task dataset contains questions belonging to "question families" based on CLEVR (Johnson et al., 2016), reflecting specific reasoning competences. These families were explicitly marked as 19 categories, the last one having no answer (N/A), but direct reference to these categories was prohibited by the task requirements.

The cooking recipes included in the dataset were provided with exceptionally extensive annotations containing semantic information. The authors applied CRL and span-based SRL using VerbAtlas (Di Fabio et al., 2019) for the reference inventory of frames and semantic roles. Subsequently, human annotators were asked to validate and correct frames and argument labels.

The dataset was split into training (26,526), validation (3,829) and test set (3,442 questions). At the competition evaluation stage, the answers to the latter were not revealed, but the annotations were retained.

Our source code is available on GitHub[3].

2 Related Work

In the recent years, deep learning systems trained on large datasets began to outperform humans and other algorithms in the whole QA discipline. Challenges presented in works such as SQuAD (Rajpurkar et al., 2018), MS MARCO (Nguyen et al., 2016), CoQA (Reddy et al., 2019), multilingual MLQA (Lewis et al., 2020) and others popularized various machine learning models for extractive QA. Meanwhile, visual and multimodal QA contests started to appear, e.g. VQA (Antol et al., 2015) or Audio-Visual Scene-Aware Dialog (Alamri et al., 2019). They require understanding of images, natural language and their mutual relations to produce answers. One should not overlook QA systems applying SRL annotations used in advanced answer and question generation, such as Fitzgerald et al. (2018).

QA systems were proposed for open domains as well as specific ones, including medicine, education, tourism, weather forecasting, etc. One of the most popular yet challenging topics for QA is cooking. The system in Khilji et al. (2021) required preparing a cooking-related ontology, categorizing questions and extracting potential answers.

[1]https://competitions.codalab.org/competitions/34056 (access Apr 28th, 2022).

[2]Abbreviations used in the text: QA: Question Answering; SRL: Semantic Role Labeling (or Labels); CRL: Cooking Role Labeling; EM: Exact Match; RC: Reading Comprehension; DNN: Deep Neural Networks.

[3]https://github.com/samsungnlp/semeval2022-task9

Haussmann et al. (2019) focused on a knowledge graph used to answer a range of questions related to healthy diet.

Furthermore, food recognition could be perceived as a part or a preliminary step for cooking QA. Mohanty et al. (2021) and Akhi et al. (2018) seek for deep learning classifiers to properly identify food from real images.

3 System Overview

3.1 Questions Categorization

Because using original question categories was not allowed, we started with building a categorization solution. We based it on syntactic and lexical structure of the questions and used regular expressions as a way of distinguishing them; details can be found in Appendix C. Subsequently, to discover relationships between resulting question groups, as well as within them, we took SRLs and CRLs into account. They allowed us to determine a word or a phrase that should be included in the answer to a given question. Finally, we distinguished 17 question categories. To match the answers more effectively, some of them were later divided into subcategories:

1. COUNTING TIMES — counting how many times a given TOOL or HABITAT is used.
2. COUNTING USES — counting how many TOOLS or HABITATS are used.
3. COUNTING ACTIONS — counting how many actions it takes to do something.
4. ELLIPSIS — searching for direct object(s) which has undergone a certain process.
5. LOCATION (CRL) — searching for the place to which something is being transferred or in which it is located (a CRL is returned).
6. LOCATION (SRL) — similar to the above, but an SRL is returned.
7. METHOD — searching for a way of performing an action, with four subcategories according to which a CRL or an SRL is returned as an answer:
   - Question about a TOOL,
   - Question about an INSTRUMENT — objects or forces (such as heat, cold) that come in contact with an object and cause a change in it,
   - Question about an ATTRIBUTE — a property that a direct or indirect object possesses,
   - Question about a GOAL — the point to which something (e.g. temperature/heat/flame, consistency, thickness) needs to be brought.
8. LIFESPAN (HOW) — searching for a result of a process; a related action and its objects are returned as the answer.
9. LIFESPAN (WHAT) — similar to the above, but only related objects are inserted into the answer (without the action).
10. EVENT ORDERING — checking which action should be performed first.
11. RESULT — searching for expressions determining to what point a condition has changed.
12. TIME — searching for a specific expression relating to time.
13. EXTENT — searching for expressions specifying the range or degree of change.
14. PURPOSE — searching for expressions describing why an action needs to be performed.
15. CO-PATIENT — searching for indirect objects that undergo a process, are affected in a certain way, are situated in a particular location or are transferred to a different location.
16. SOURCE — searching for a starting point of a motion.
17. LOCATION CHANGE — searching for previous location of an object.

3.2 Approach Based on Semantic Roles

The system uses the following three-step path to find the answer: intent identification, data extraction and response generation.

Having the intent predicted, the question is dispatched to one of the per-category handlers. We designed the system to use a separate answerer for each category. LOCATION, RESULT, TIME, EXTENT, PURPOSE, CO-PATIENT and SOURCE share the same code after some parametrization, see Table 4. For the remaining categories (COUNTING, ELLIPSIS, LOCATION CHANGE, EVENT ORDERING, METHOD and both LIFESPANS) we use separate sub-engines, as we need to perform diverse tasks.

The implementations (except for METHOD) are pretty straightforward and obey the general rule:

- identify a reference verb and / or object in the question,
- search for a relevant sentence using the same verb / object in the given role (category-dependent) and extract relevant informa-

tion from the sentence using semantic roles (category-dependent again),
- if necessary, rephrase the information to form the answer.

Since METHOD contains four original categories (2, 6, 10 and 14) and direct use of the category ID was prohibited, the sub-engine for METHOD runs the above steps multiple times for: ATTRIBUTE, INSTRUMENT, GOAL and TOOL, returning the first found answer. The exact order of the labels was found empirically, by minimizing the number of category mismatches on the validation set.

In following sections we discuss the details.

Intent Identification

In almost every category the key to answering a question is identifying the verb and the object associated to it (jointly referred to as *intent*) and then finding the answer in the annotation.

First, a question classifier (see Section 3.1) is used to assign the question to the relevant category. Then, the analysis of the recipe is performed in an iterative way. We start with a small chunk (sentence or paragraph) to prevent mismatches resulting from looking too broadly. Verbs from the analyzed part are collected using either SRL (more specifically, tokens labeled as B-V), or a CRL and SRL combination, namely finding B-EVENT (CRL) with corresponding SRL (I-V or D-V). A detailed description of the annotation system is presented in Tu et al. (2022).

The next step of the intent identification requires iteration over the collected verbs to find a related object for each of them. The objects may be annotated in numerous ways:

- using SRL (e.g., PATIENT, THEME)
- using CRL (e.g., TOOL, HABITAT, EXPLICIT-INGREDIENT)
- using HIDDEN ROLES (e.g. DROP, HABITAT)

When both the verb and the associated object occur in the question, the system is ready to utilize this information to search for the answer in the recipe. More details of our algorithm can be found in Appendix C.

Data Extraction

Depending on the identified intent, the answer may appear in the passage either explicitly or implicitly. In the first case, the data essential for generating the answer is a direct span from the recipe and the system only needs to find an appropriate SRL and return it as the answer. However, for question categories where the answer is not explicitly mentioned in the passage, the process of data extraction is far more complicated. It requires calculating the actions, tracking object position, or collecting parts of the answer using all the information available in the annotation part: SRL, CRL, HIDDEN ROLES and the relations between them.

Response Generation

Generation of the final response is category-specific. In some cases, the gold answer contains only words annotated as a specific SRL (e.g. LOCATION, TIME). In other categories, the gold response contains the verb and the object from the question. There are categories where the system has to count occurrences of the object and return the number, as well as ones where the phrase *by using* is required at the beginning. See Appendix C for details.

3.3 DNN-Based Systems

QA is a well-established NLP task, mainly thanks to the advancements in attention-based DNN models. Thanks to fine-tuning, pre-trained BERT (Devlin et al., 2018) and its successors may be employed for downstream tasks, such as RC.

To examine how successful RC models could be for the competition, we tested the following ones: BERT, RoBERTa (Liu et al., 2019) and ELECTRA (Clark et al., 2020) in the extractive setup, i.e. taking text spans as predicted answers. We fine-tuned them on the SQuAD dataset and then on the task recipes from the train set. We used large versions of the models, and trained them for 5 (BERT), 12 (RoBERTa) and 15 (ELECTRA) epochs. The other hyperparameters used were: batch size: 8, learning rate: 1.5e-5, and max token sequence length: 512. Notably, we did not use provided annotations so that the solution was based solely on raw recipe texts.

N/A Classifier

An important part of the task is the correct identification of N/A answers in the QA pairs. The dataset contains about 9% of such, spread across all categories quite evenly. In most cases, the rule-based system was enough to identify the missing response in a cooking recipe. For more problematic situations we reached the best classification results by fine-tuning the *bert-base-multilingual-uncased* model (Wolf et al., 2020) taken from the PyTorch Hugging Face repository[4].

4 Results

Our end-to-end hybrid system reached EM scores between 80% and 100% per question category and the official result overall amounted to 92.53%. Details can be found in Table 1.

Our pipeline starts with the semantic-based system. If no answer is returned, the RC system is used if its confidence threshold exceeds 98%. This fallback mechanism produced any significant improvement only for the LOCATION (SRL) category. We additionally consider the N/A Classifier result: if it exceeds the 99% certainty threshold, the N/A answer is returned. This operation enhances the results for the EVENT ORDERING category. Adding the DNN systems in such a way leads to a 0.145 pp result increase.

In the post-evaluation phase we made further improvements, mainly in the rule-based system, and we ultimately reached the 92.969% EM result.

Human annotators reached notably low EM score of 52%. It is mainly due to the fact that the exact match metric leaves no room for human creativity. A manual review of the semantic validity of the responses gave us 73% alignment with the gold answers. This is discussed further in Section 4.2.

Manual analysis revealed that only some elements of the images associated with the recipes relate directly to recipe content. This was also mentioned by task organizers. Only 62 from 500 analyzed pictures were considered helpful in answering questions by our human evaluators. They also reported that they were often assigned to a different recipe step. For these reasons we further disregarded the images and focused only on the textual part of the data.

4.1 DNN-based Systems

Table 2 shows RC models comparison. As expected, the best Exact Match score was achieved by ELECTRA, which is currently the top-performing model on the SQuAD benchmark.

Table 3 presents the percentage of test set questions that could be answered by an oracle extractive answering system, i.e. where the answer either can be found as a span from the recipe text or it is N/A. Such examples cover 35% of the test set, meaning that this is the upper limit for any extractive QA system. The result achieved by ELECTRA (EM equal to 31%) is in line with this estimation. Another 34.6% EM could be achieved with an extractive QA system by using additional post-processing.

This leaves out 30.4% examples, mainly from categories COUNTING, LIFESPAN and EVENT ORDERING, that require non-trivial processing (e.g. rephrasing) and/or aggregation of information from various parts of the recipe.

Based on these results we claim that ELECTRA or other BERT-based systems can be considered applicable for this type of task, yet they should be able to generate answers beyond plain span extraction. It would require improvements, such as making use of a generative model, feeding the semantic annotations along with recipe texts, and perhaps adjusting the models to specific question categories.

The N/A Classifier worked better when fed with the full recipe passage (i.e. *Ingredients* and *Directions*; $F1 = 82.7\%$). If provided only with *Directions*, the result dropped to $F1 = 76.6\%$. It shows that the *Ingredients* information plays an important role in the solution.

4.2 Human benchmark

The aim of creating the human benchmark described in this subsection has no other purpose but to measure to what extent our results (as well as the gold answers) are close to human reasoning, i.e. to the answers provided by an actual person. We did so as we did not find any information on human performance in materials provided by the organizers.

We asked a group of six linguists to answer 2,000 questions selected randomly from the validation set. We maintained similar percentages of each question category for the sample to be representative. Before starting their task, the linguists had become familiar with the train dataset to grasp the main idea and the structure of questions and answers. Importantly, they did not have access to the annotation so that they based their answers solely on the recipe texts and related pictures. We decided to take this approach assuming that the semantic annotation of the recipes serves as a partial equivalent of the general knowledge that AI lacks.

As already mentioned, the manual review of the human answers revealed that 73% of them have the same meaning as the gold answers. Other re-

[4]`https://huggingface.co/models` (access Feb 20th, 2022).

Table 1: Exact Match percentage per category. For the training (**Train**), validation (**Val**) and test (**Test**) sets we present the results of our hybrid system. **Post-Eval** shows our final post-evaluation results if different than **Test**. **Size** is a percentage of the given question category in the whole validation set. **Human** results were calculated based on a sample from the validation set as described in Section 4.2. For **Electra** we took the full test set.

Category	**Size**	**Train**	**Val**	**Test**	**Post-Eval**	**Human**	**Electra**
COUNTING TIMES	2.3	80.6	95.3	88.5		41.9	9.0
COUNTING ACTIONS	6.2	89.7	88.4	87.8		52.7	8.9
COUNTING USES	5.4	98.1	97.5	98.4		77.1	10.2
ELLIPSIS	13.8	89.2	89.3	89.5		20.9	22.7
LOCATION (CRL)	9.4	98.4	97.5	98.4		51.0	47.2
LOCATION (SRL)	8.0	95.6	96.5	95.3		69.5	80.1
METHOD	13.4	86.4	87.9	87.0	88.0	37.1	23.9
LIFESPAN (HOW)	5.4	89.1	91.6	88.7		5.1	10.8
LIFESPAN (WHAT)	5.1	93.7	93.9	92.6		15.6	21.1
EVENT ORDERING	15.8	97.1	97.8	96.7	97.2	93.4	9.8
RESULT	2.5	95.9	97.9	96.5		96.2	83.5
TIME	3.0	87.8	94.2	90.3		74.7	73.8
EXTENT	0.3	100.0	100.0	88.9		0.0	88.9
PURPOSE	1.2	98.2	100.0	97.6		81.8	82.9
CO-PATIENT	0.6	88 .4	95.8	85.0		64.3	90.0
SOURCE	0.6	96.4	100.0	100.0		68.4	31.0
LOCATION CHANGE	7.2	93.9	97.0	91.5	93.9	40.8	40
Total		**92.7**	**93.9**	**92.5**	**93.0**	**52.0**	**31.0**

Table 2: Reading Comprehension models results for the test set (0 - 100 range). **EM** — Exact Match score.

Model	**F1**	**EM**
BERT	36.9	30.7
RoBERTa	37.7	30.7
ELECTRA	**38.5**	**31.0**

sponses are semantically close, yet not identical. However, they often differ lexically from gold answers, resulting in the low overall EM score.

It is particularly visible in the ELLIPSIS category:

> **Question:** What should be tossed?
> **Gold answer:** the rice mixture and yogurt mixture
> **Human answer:** yogurt, sour cream, mustard, sugar, salt, pepper and rice mixture

The linguists also failed to return the gold answer when the question itself was semantically ambiguous. It was mostly applicable to the METHOD category and to both LIFESPAN categories. In the former, it results from various possible ways of understanding the English word *how*:

> **Question:** How do you slice the tomatoes?
> **Gold answer:** by using a knife
> **Human answer:** slice the tomatoes thinly

The LIFESPAN questions require listing ingredients needed to obtain something. The discrepancies between the gold answer and the human answer often resulted from a different nouns ordering or using a synonym:

> **Question:** How did you get the hot chocolate?
> **Gold answer:** by mixing the hot water, milk and mixture in the mug
> **Human answer:** by mixing the mixture with hot water or milk in a mug

As linguists did not see the annotation, their proposals were often different from the gold answer in categories where it was taken from SRL or CRL, such as METHOD subcategory concerning tools or habitats, or the COUNTING categories. Moreover, e.g. in both LIFESPAN categories, gold answers either listed the ingredients explicitly or returned the DROP value (one of the HIDDEN ROLES), such as "mixture", "soup" or "dough". From the human point of view those two kinds of responses would be equally correct:

> **Question:** What's in the mixture?

Table 3: Extractive Answering usability on the task. **EA** — answers present in source texts as non-empty spans. **NA** — N/A answers in the test set. **AQ** — Answerable Questions (EA + NA); i.e. an oracle system result. **EM** — Exact Match ($\leq$ AQ) actually achieved by our ELECTRA system. All results are provided as percentage and based on the test set.

Category	**EA**	**NA**	**AQ**	**EM**
Counting Times	0	9	9	9
Counting Actions	0	9	9	9
Counting Uses	0	10	10	10
Ellipsis	12	10	22	21
Location (CRL)	50	10	60	47
Location (SRL)	75	9	84	81
Method	13	12	25	24
Lifespan (how)	0	11	11	11
Lifespan (what)	16	11	27	21
Event Ordering	0	10	10	10
Result	79	5	84	83
Time	67	13	80	74
Extent	78	11	89	89
Purpose	78	10	88	83
Co-Patient	80	10	90	90
Source	81	5	86	31
Location Change	48	14	62	40
Total	25	10	35	31

Gold answer: the egg and mixture

Human answer: the butter, sugar, tangerine zest, vanilla, baking powder, salt and egg

The linguists obtained the best results in the Event Ordering, Result, Time and Purpose categories. Apart from the last one, those are closed-form questions that leave little room for semantic ambiguity.

We treated human benchmark as an interesting experiment that confirmed two hypotheses we had. Firstly, the answers provided by our model are often semantically close to the gold answers, as stated above. The scoring criteria reject any answer that is not identical to the gold one, which leads to allegedly poor human performance and makes the answer post-processing a daunting but crucial step. Secondly, there are some patterns in the task data that are remote from human thinking. The result of the experiment did not affect the final score — it served solely for analytic purposes.

5 Conclusion and future work

Our main contribution is the hybrid system for the cooking-related QA. While we are satisfied with the result, the ~7% error rate still leaves some room for improvement.

The most challenging task for our system was the correct intent identification. This is visible in the fairly low results in the Method category. It may relate to four different intents, and we did not always distinguish them properly. Other problematic aspects were counting actions and objects and generating answers that contain all required items in the right order. These issues solely contribute to as much as 5.5 out of 7 pp constituting the whole the error rate.

The obvious question left unanswered is the possibility of SRL/CRL annotation automation, also for other competence domains. This is a missing component in a full end-to-end application of our solution.

References

Amatul Akhi, Farzana Akter, Tania Khatun, Mohammad Uddin, Mohammad, and Shorif Uddin. 2018. Recognition and Classification of Fast Food Images. *Global Journal of Computer Science and Technology*, 18.

Huda Alamri, Vincent Cartillier, Abhishek Das, Jue Wang, Anoop Cherian, Irfan Essa, Dhruv Batra, Tim K. Marks, Chiori Hori, Peter Anderson, Stefan Lee, and Devi Parikh. 2019. Audio-visual scene-aware dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

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

Andrea Di Fabio, Simone Conia, and Roberto Navigli. 2019. VerbAtlas: A novel large-scale verbal semantic resource and its application to semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 627–637, Hong

Kong, China. Association for Computational Linguistics.

Nicholas Fitzgerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060, Melbourne, Australia. Association for Computational Linguistics.

Steven Haussmann, Oshani Wasana Seneviratne, Yu Chen, Yarden Ne'eman, James Codella, Ching-Hua Chen, Deborah L. McGuinness, and Mohammed J. Zaki. 2019. Foodkg: A semantics-driven knowledge graph for food recommendation. In *SEMWEB*.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. 2016. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. *CoRR*, abs/1612.06890.

Abdullah Faiz Ur Rahman Khilji, Riyanka Manna, Sahinur Rahman Laskar, Partha Pakray, Dipankar Das, Sivaji Bandyopadhyay, and Alexander F. Gelbukh. 2021. CookingQA: Answering Questions and Recommending Recipes Based on Ingredients. *Arabian Journal for Science and Engineering*, pages 1–12.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics.

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

Sharada Prasanna Mohanty, Gaurav Singhal, Eric Antoine Scuccimarra, Djilani Kebaili, Harris Héritier, Victor Boulanger, and Marcel Salathé. 2021. The food recognition benchmark: Using deeplearning to recognize food on images. *arXiv preprint arXiv:2106.14977*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *CoCo@NIPS*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Jingxuan Tu, Eben Holderness, Marco Maru, Simone Conia, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Roberto Navigli, and James Pustejovsky. 2022. Semeval-2022 task 9: R2VQ – competence-based multimodal question answering. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

A Dataset Analysis

One of the major problems we encountered during the linguistic analysis was question duplicates:

Semantically justified

- Semantic ambiguity resulting from the specific characteristics of the English language, so that it is not possible to distinguish question types by their syntactic structure or by other elements. For example, four groups of semantic roles as possible answers can appear in the same recipe
  Duplicated question: How do you mix the shrimp, pasta, butter and parsley?
  Answer 1: mix the shrimp, pasta, butter and parsley well
  Answer 2: by using a spatula

- With the same question structure, semantic ambiguity resulting from the content of the recipe, e.g. the same verb appearing twice in the text.
  Duplicated question: What should be added to the pan?
  Answer 1: the string beans and dressing
  Answer 2: the sauteed garlic, onions, ginger and string beans

In such cases it might be a better solution to list the correct answers instead of giving just one.

Semantically unjustified

- With the same question structure, referring to the same object in the recipe; these appeared mainly in the COUNTING category.
  Duplicated question: How many bowls are used?
  Answer 1: N/A
  Answer 2: 1

The aforementioned example of unjustified duplicates is associated with another problem. Namely, for many questions marked as unanswerable it was actually possible to find an answer in the recipe. We suppose that this was caused by selecting random questions from other recipes and assuming that they could not be answered based on the content of another recipe. Unfortunately, due to the relatively small variety of vocabulary related to cooking, this assumption was misleading. This can be seen especially in the categories: COUNTING, LIFESPAN and ELLIPSIS.

B Additional Experiments

B.1 Applicability to Another Domain QA

In this experiment we checked whether our system would work for other domains. We chose four instruction texts that are related to make-up techniques, furniture assembling and handmade Christmas decorations. We labeled these texts manually and asked linguists to write questions and answers bearing in mind the structure of questions and answers proposed by the organizers. They created 20 QA pairs for each text on average. The system achieved results in the range of 40%-50% EM (if we additionally included responses that are semantically correct, but not fully consistent with the answer suggested by the question authors, we reached approximately 60% EM). It is worth emphasizing that this was possible without any changes in our system.

B.2 Completeness and Correctness of Question Intents

The second experiment checked whether the questions provided by the organizers were semantically diverse and to what extent they corresponded to potential human intentions. We asked linguists to write questions related to five recipes from the validation set. Importantly, for the sake of an unbiased experiment, those were not the same people who worked on the human benchmark. The linguists engaged in this experiment had not seen questions and answers provided by the organizers, so the structure of independently written questions is not influenced by the existing dataset. They prepared about 100 question-answer pairs (20 for each recipe). After comparing the questions provided by the organizers to the ones created by our linguists, we concluded that some question types have not been included in the competition dataset:

- questions related to the amount of ingredients, e.g *How many tablespoons of vinegar should I add?*
- questions about the type of ingredients, e.g. *What kind of oil should I use for this recipe?*
- yes-no questions, e.g. *Is spinach required for this recipe?*
- questions about name or type of the dish, e.g. *What is this recipe for?*

Therefore, we have four extra categories not mentioned by task organizers. On the other hand, every category in Section 3.1 was covered by at least one question.

It is also noteworthy that most of the questions formulated by linguists are in the first person singular instead of the second person, as the organizers propose. Also, they respond using a whole sentence rather than a single word or a short phrase. The remaining questions written by the linguists correspond to the questions categories proposed by the task organizers. This proves that the proposed Question categories are valid and reflect real human intentions. It should be emphasized that the structure of human-written questions and answers is much more varied, but they still contain keywords that can be used without problems in our question classifier.

It must be stressed that our manual annotation concerned entirely new texts, only for the purpose of these experiments. We did not use the any of the additionally annotated data to augment the datasets provided by task organizers. Therefore, the experiments did not affect out final score.

C Implementation Details

The process of searching for information in a recipe and generating answers is presented in the Algorithm 1. It utilizes information such as types of semantic labels playing the crucial role while answering a given question category. It is summarized in Table 4, which also shows regular expressions used by our classification system.

- By *event* we usually mean a verb annotated as EVENT which should match the verb from the question. If the question also includes an adverbial, it can be used to distinguish the correct *event* in the recipe.
- By *object* we mean a word or phrase, which is annotated as DROP, PATIENT or THEME and matches the *object* from the question. In some cases no *object* is provided. Then the system relies on *event* matching.
- In the COUNTING category we need to search directly for TOOLS, HABITATS or RESULTS.

If no matching *event*, *object*, HABITAT, TOOL or RESULT can be found within the recipe, the system concludes that the question is not answerable.

Example of answering can be seen in Fig 1.

Additional Remarks

To ensure higher accuracy of the results, the system has to take into account several characteristics of

Algorithm 1: Answer generation process

```
Input    : question, recipe
Output   : generated answer
question category ← predict category using
  regex from TABLE 4 COLUMN 2
question details ← extract details from the
  question (see COLUMN 3)
relevant information ← search for relevant
  part in the recipe using question details
RC threshold ← 0.98
NA threshold ← 0.99
if relevant information was found then
    answer ← generate answer for given
      question category according to
      COLUMN 4
else
    answer ← use answer predicted by
      Electra Extractive QA[1]
    if confidence < RC threshold then
        answer ← N/A
if N/A Classifier[1,2] output = N/A and
  confidence >= NA threshold then
    answer ← N/A
return answer
```

[1] Electra Extractive QA and N/A Classifier are used only for some categories

[2] N/A Classifier was added after competition end

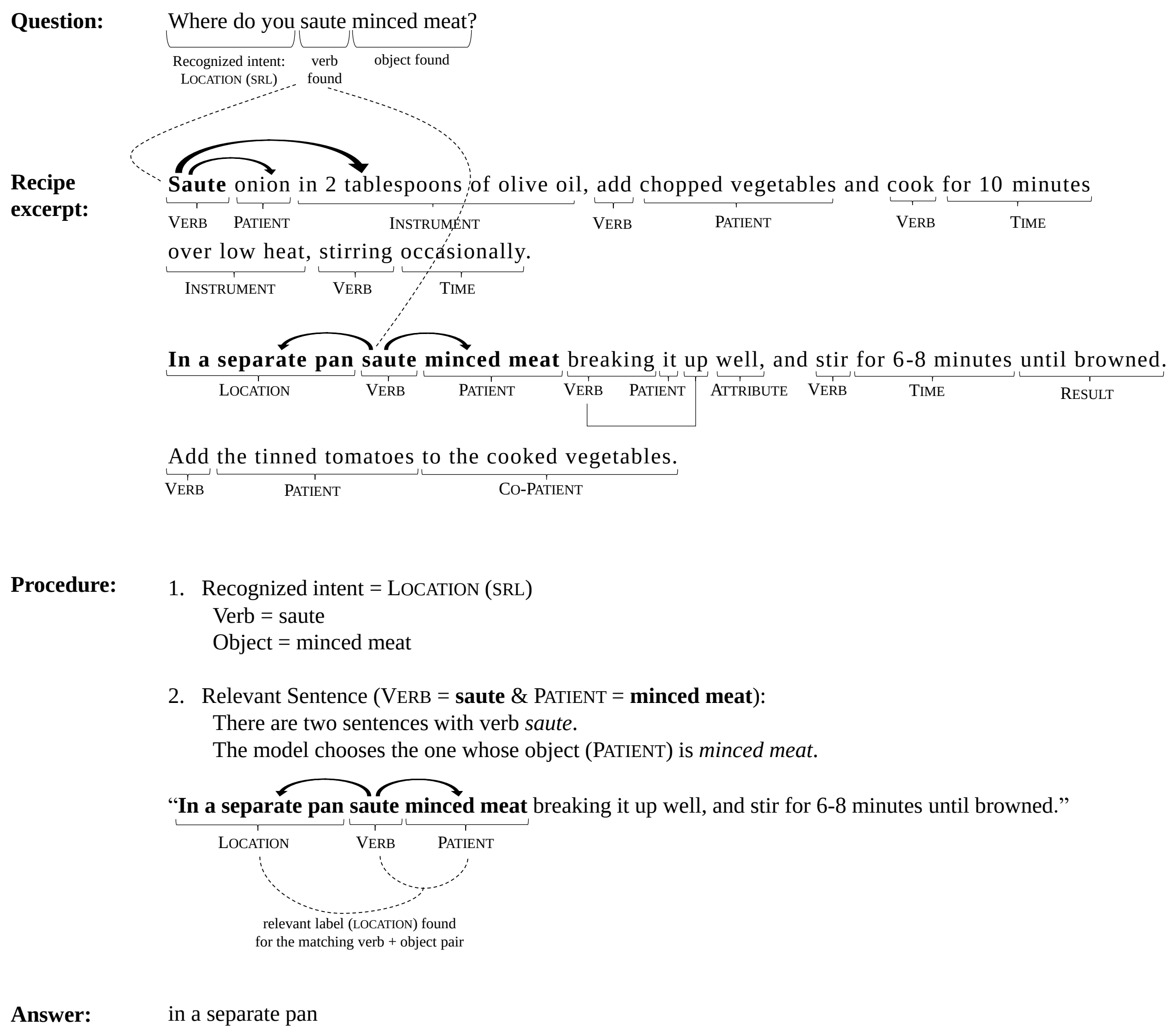
Table 4: Summary of question handling. Columns left-to-right: category, regex used for initial classification, semantic information used to search for the answer in the recipe, information used to generate the final response.

Category	Regex Pattern	Searched Label	Answer Generation
COUNTING TIMES	How many times	tools or habitats	count found occurrences
COUNTING ACTIONS	How many actions	result and corresponding event	count found occurrences
COUNTING USES	How many .* are used	tools or habitats	count found occurrences
ELLIPSIS	What should	event and (tool or habitat)	drops, ingredients
LOCATION (CRL)	Where should you	event and object	habitat
LOCATION (SRL)	Where do you	event and object	location, destination, co-patient or co-theme
METHOD	How do you	event and (object or ingredients)	verb, object, one of: tool, instrument, attribute, goal
LIFESPAN (HOW)	How did you get	result and corresponding event	verb, drops, patients, tools, habitats
LIFESPAN (WHAT)	What's in	result and corresponding event	ingredients (if patient or theme), drops
EVENT ORDERING	.* which comes first	both events	use the preceeding one
RESULT	To what extent	event and object	result
TIME	For how long	event and (object, attribute or purpose)	time
EXTENT	By how much	event and object	extent
PURPOSE	Why do you	event and object	cause or purpose
CO-PATIENT	What do you .* with	event and object	co-patient or co-theme
SOURCE	From where	event and object	source
LOCATION CHANGE	Where was .* before	event and object, and all previous events for the same object	previous habitat different from the one in the starting event

the R2VQ dataset:

- SRLs are represented as columns, within which objects are connected to the verb. Each subsequent verb within a sentence has its own column with corresponding objects. Iterating over each column separately appears to be very helpful in terms of associating verbs with proper objects.
- Each SRL starts with the head (the label starts with the letter B). If the phrase contains multiple words, the head is followed by the body (the label starts with the letter I or D). We found that concatenation of the full-length expression (using B and I as indicators) improves the quality of the identification process.
- Tokens whose CRL is TOOL, HABITAT, EXPLICITINGREDIENT or IMPLICITINGREDIENT are supplied with the index of the verb to which they relate. In some cases, using that information is extremely helpful as it allows for unambiguous identification of the relationship between the verb, the object and the answer.
- While an object in the question is created using a HIDDEN ROLE, it is needed to singularize each part of it. For example, if Drop="**limes**.3.1.9:**ginger**.3.1.1:**onions**.2.1.7" there is a great chance that it will appear in the question in the form of *the lime, ginger and onion*. On the contrary, when CRL or SRL were used to create the question, they will most likely appear as an unchanged span from the passage.

Figure 1: Example of the sematic-role-based answer generation.



Question: Where do you saute minced meat?

Recipe excerpt:

Saute onion in 2 tablespoons of olive oil, add chopped vegetables and cook for 10 minutes over low heat, stirring occasionally.

In a separate pan saute minced meat breaking it up well, and stir for 6-8 minutes until browned.

Add the tinned tomatoes to the cooked vegetables.

Procedure:

1. Recognized intent = LOCATION (SRL)
   Verb = saute
   Object = minced meat

2. Relevant Sentence (VERB = **saute** & PATIENT = **minced meat**):
   There are two sentences with verb *saute*.
   The model chooses the one whose object (PATIENT) is *minced meat*.

"**In a separate pan saute minced meat** breaking it up well, and stir for 6-8 minutes until browned."

Answer: in a separate pan

PINGAN_AI at SemEval-2022 Task 9: Recipe knowledge enhanced model applied in Competence-based Multimodal Question Answering

Zhihao Ruan, Xiaolong Hou, Lianxin Jiang
Ping An Life Insurance Company of China
{ruanzhihao322, houxiaolong430, jianglianxin769}@pingan.com.cn

Abstract

This paper describes our system used in the SemEval-2022 Task 09: R2VQ - Competence-based Multimodal Question Answering. We propose a knowledge-enhanced model for predicting answer in QA task, this model use BERT as the backbone. We adopted two knowledge-enhanced methods in this model: the knowledge auxiliary text method and the knowledge embedding method. We also design an answer extraction task pipeline, which contains an extraction-based model, an automatic keyword labeling module, and an answer generation module. Our system ranked 3rd in task 9 and achieved an exact match score of 78.21 and a word-level F1 score of 82.62.

1 Introduction

In this paper, we discuss an approach to the Question Answering (QA) task for SemEval-2022 Task9(Tu et al., 2022). This task is structured as question answering pairs, querying how well a system understands the semantics of recipes derived from a collection of English cooking recipes and videos, which involve rich semantic annotation and aligned text-video objects.

In this task, a large proportion of answers can't be derived directly from the original recipe text and the information of these answers is hidden in annotated knowledge data. So we adopted two knowledge-enhanced methods in this model: the knowledge auxiliary text method and the knowledge embedding method. The knowledge auxiliary text method incorporates the hidden-roles knowledge and co-reference knowledge in generating auxiliary text. The knowledge embedding method encodes u-pos knowledge and entity knowledge into knowledge embedding.

A key evaluation measure in this task is the exact match score. Because the extraction-based model is more robust than the generative-based model, we design an answer extraction task pipeline. The pipeline contains an extraction-based model, an automatic keyword labeling module, an answer restructures module. In the training phase, we locate keywords of answers in recipe text and provide training data by labeling the keywords. In the prediction phase, we collect output keywords of the model and generate answer text by the keywords.

Our system ranked 3rd in task 9 and achieved an exact match score of 78.21 and a word-level F1 score of 82.62. We make our code publicly available on Github[1].

2 Related Work

Question Answering (QA) Liu et al. (2019) described the Span Extraction task in MRC (Machine Reading Comprehension) and Mervin (2013) described the extraction-based question answering task. BERT(Devlin et al., 2019) displayed a general extraction-based approach for the QA task.

Knowledge enhanced CoLAKE(Sun et al., 2020), ERNIE 3.0(Sun et al., 2021) and K-BERT(Liu et al., 2020) shown the method of constructing a new context by using knowledge information. Know-BERT (Peters et al., 2019) used the knowledge embedding method and Zhang et al. make some improvements.

Prompt In generating knowledge auxiliary text, we are inspired by prompt(Liu et al., 2021) learning. We focus on prompt engineering in this paper. Yuan et al. (2021) rewrite the context by replacing phrase. Davison et al. (2019) use a unidirectional LM to score the prompt patterns. Gao et al. (2021) use the T5 model to generate a template.

3 System overview

In this paper, we discuss an approach to the Question Answering (QA) task for SemEval-2022 Task9.

[1]https://github.com/archfool/SemEval2022_Task09

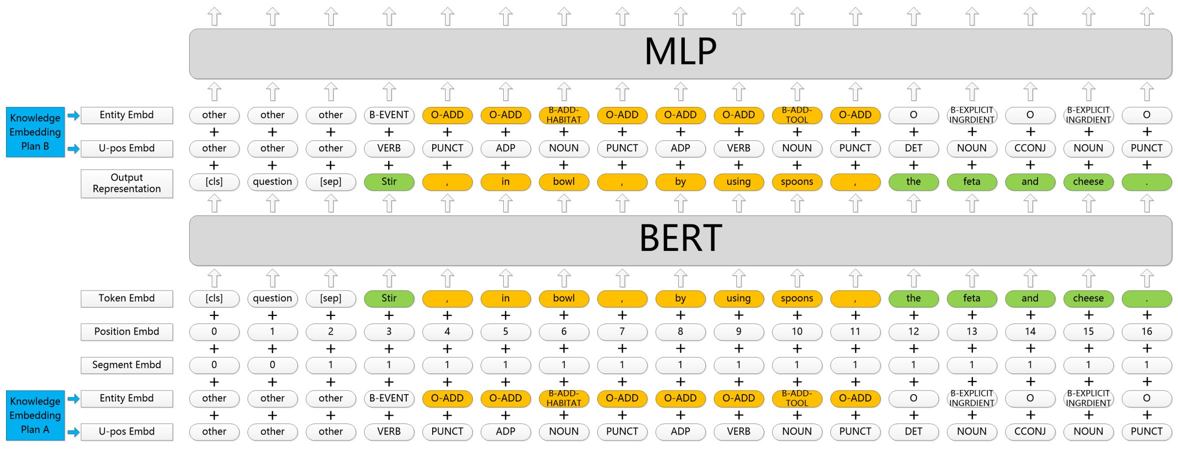


Figure 1: knowledge enhanced model

We divided the samples into 18 categories after analyzing the patterns of QA pairs. We predict answers of 6 categories by rule-based method and 12 categories by deep-learning model.

We chose BERT(Devlin et al., 2019) as the backbone of our model and chose the answer extraction method for this QA task.

Knowledge information is significant in this task, so we adopted two knowledge-enhanced methods: the knowledge auxiliary text method and the knowledge embedding method.

Knowledge auxiliary text method generating expanded text that contains original text and knowledge auxiliary text. The knowledge auxiliary text incorporates the hidden roles and co-reference knowledge.

Knowledge embedding method encoding u-pos (universal-part-of-speech) knowledge and entity knowledge into knowledge embedding.

3.1 QA datasets analysis

Following the design ideas of the FAQ task, we analyzed the patterns of QA pairs first. And we summarize QA pairs into 18 pattern categories.

With the help of divided pattern categories, we fed certain categories of data into a rule-based module. For example, some questions are about counting tasks, the rule-based method is better for these tasks than the deep-learning model method. We fed the rest QA categories into the deep-learning model module. The proportion of samples using the rule-based method was 39.87%, and the proportion of samples using the deep-learning model method was 60.13%.

The QA-pair pattern categories' example is shown in Table 2 of Appendix A.

Some more dataset information is shown in Section 4.1.

3.2 Knowledge auxiliary text

By analyzing the QA cases of the datasets, we found that a large proportion of answers can't be derived directly from the original recipe text. And the information of the answers is in the Cooking Role Labeling (CRL) annotations. So we generate knowledge auxiliary text by introducing the knowledge of the co-reference column and hidden-role column in CRL annotations. We add knowledge auxiliary text to the original text as the input of the model.

Co-reference knowledge can be regarded as a kind of entity link between the current entity token and the token where this entity is mentioned the first time. In this way, readers can search and locate the item unambiguously even though different items have the same name or one item uses different names. Knowledge auxiliary text of co-reference information is generated by easily adding the alias in the co-reference column within a couple of brackets.

For example, if the original text is "mixture" and the co-reference column has the value "small balls". Then we generate knowledge auxiliary text "(small balls)". Combined knowledge auxiliary text to the original text, and get the final text: "mixture (small balls)".

Hidden-role knowledge is a supplement to the action token that appears in the original text when some elements of the action are hidden. Inspired by prompt(Liu et al., 2021) learning, we generate knowledge auxiliary text following the pattern of the answer text from the QA-pairs. For example, we use 'by using a knife' as knowledge auxiliary text when the hidden role column has a key value of 'TOOL: knife'.

The knowledge auxiliary text method can be shown in Figure 1. In the token embedding layer, the tokens of green grids are the original text of the recipe, and the tokens of orange grids are the knowledge auxiliary text. We combined the original text and the knowledge auxiliary text as the input of the model.

3.3 Knowledge embedding

Entity knowledge and u-pos knowledge is useful in predicting answers in QA task. For example, a QA pair is: "How do you cut the carrots? by using a knife". In this case, the token of a knife is labeled as B-TOOL in the entity column. It would help predict the answer if the model gets the token's entity type information.

In our model, we adopt the knowledge embedding method to incorporate the knowledge of the u-pos column and entity column. Similar to the token embedding representation, the knowledge embedding method maps values in the u-pos column to embeddings. The size of the knowledge embedding is the same as token embedding. The entity column's knowledge embedding method is the same.

As shown in Figure 1, we have two places options to apply knowledge embedding: the embedding layer (knowledge embedding plan A in Figure 1) or the header (knowledge embedding plan B in Figure 1).

In plan A, we add knowledge embedding to the embedding layer of the model. Compared with BERT's embedding layer whose output embedding is the sum of token embedding, position embedding, and segment embedding, our model adds entity knowledge embedding and u-pos knowledge embedding to the original BERT's embedding layer.

In plan B, we add knowledge embedding to the header of the model. We first sum up the output representation of BERT's last layer, the entity knowledge embedding, and the u-pos knowledge embedding. Then fed it into a small network such as MLP(Rosenblatt, 1957).

3.4 Answer extraction method for QA task

Two common methods of predicting answers in QA tasks are the extraction-based method and the generative-based method. In our model, we chose the answer extraction method for the QA task. Here are the steps to label the data for extracting answers:

First, we filtered out keywords from the questions and answers by deleting words whose u-pos are article, preposition, pronoun, and so on.

Second, we located the keywords in the text. According to the statistics, 88.5% of the QA samples can be located successfully in one sentence, and 0.5% in two adjacent sentences. 11% samples' keywords can't be located in texts, these samples were regarded as low confidence samples and would not be used.

Third, we labeled the answers' keywords in the recipe texts. Concatenate the question and the labeled recipe texts as the input for the model.

Forth, predicting answer's keywords by model.

Fifth, generate answers based on the keywords.

3.5 Rule-based answer generation method

Some QA categories in Table 2 are more suitable for predicting using rule-based methods. For example, some categories are about counting tasks.

By analyzing the QA patterns, we assigned 6 categories to the rule-base module: cat 1, cat 6, cat 7, cat 8, cat 9, and cat 10. These samples make up 44.36% of the total samples.

A simple rule description of the 6 categories is shown in Appendix B.

4 Experimental setup

4.1 Dataset

The organizer of SemEval-2022 Task9 provided R2VQ (Recipe Reading and Video Question Answering) dataset.

R2VQ dataset is a collection of cooking recipes and videos. The R2VQ dataset provided to us has 1000 samples, and the organizer split the dataset into 3 parts: training dataset (800 samples), validation dataset (100 samples), and testing dataset (100 samples). The training dataset and validate dataset have corresponding answers to the questions while the test dataset hasn't.

Data of each sample in the R2VQ dataset is from two sources: cooking recipe text and screenshots from the videos. Each context of the recipes is consist of ingredients and directions whose labels have two annotation layers: Cooking Role Labeling (CRL) and Semantic Role Labeling (SRL). In the model of this paper, we only use the directions of recipes as the model's input.

position	**entity**	**upos**	**f1**
None	False	False	88.78
embed_layer	False	True	88.93
embed_layer	True	False	89.00
embed_layer	**True**	**True**	**89.23**
header	False	True	88.47
header	True	False	88.76
header	True	True	88.03
embed_layer	⋆	⋆	89.05
header	⋆	⋆	88.42
⋆	True	⋆	88.76
⋆	⋆	True	88.66

Table 1: Squad f1 for different knowledge-enhanced methods. Column Position means which place we add knowledge embedding to. Column Entity means whether we use entity knowledge or not. Column Upos means whether we use universal-pos knowledge or not. Column F1 means the word-level F1 score of the method.

4.2 Training Details

We implemented our model using the pre-trained language model BERT(Devlin et al., 2019) as the backbone and chose Adam-W(Loshchilov and Hutter, 2018) as the optimizer. In train phrase, the batch size is 8 and the optimizer's learning rate is 3e-5. We ran the experiment on an NVIDIA GeForce RTX 3090 GPU.

As like in the MRC task, the article is a long text while the question is a short text. We spilt the long recipe text into short texts. Concatenate the short recipe text and the question by the special token [SEP] as a tokens sequence. And added [CLS] token to the beginning of the tokens sequence. Every tokens sequence is an input of the model. We set the max sequence length of the tokens sequence as 512. We set the configuration of doc stride as 128 which indicates the token length of adjacent sentences' overlap.

5 Results

We ranked 3rd in the competition of SemEval-2022 task 9. Our model's word-level F1 score is 82.62 and the exact match score is 78.21.

In our system, parts of questions got answers by rule module while others by deep-learning model module, which is mentioned in Section 3. In this paper, we mainly focus on the deep-learning model module and word-level F1 score. The oblation experiment is about knowledge enhanced method.

The baseline of our original model is 61 of squad word-level F1 score, without knowledge auxiliary text method and knowledge embedding method. Its corresponding model is BERT and its input is the original text of the recipes without any annotation of SRL and CRL.

Then we adopt the methods of knowledge enhancement including knowledge auxiliary text and knowledge embedding. We got a squad word-level F1 score of 88.78 when we used the knowledge auxiliary text method and an F1 score of 89.23 when using both of the two knowledge enhanced methods. The ablation experiment results are shown in Table 1.

The record of knowledge auxiliary text method is shown in the first line (so as the first section) of Table 1, and the squad word-level f1 score is 88.78. In this method, we used a text which contains both original text and knowledge auxiliary text as the model's input data.

The second part of Table 1 is the records of the complete knowledge enhanced method containing both knowledge auxiliary text and knowledge embedding. We have done 6 groups of experiments. They are different in the place we put the knowledge embedding in and the specific embedding terms (entity knowledge or upos knowledge).

The best score for the second part is 89.23, its corresponding parameters are: adding knowledge embedding in the embedding layer, using both entity knowledge and upos knowledge.

The third part of Table 1 is a summary pivot table of the second part.

Some conclusions could be drawn from Table 1. 1) Knowledge embedding in the embedding layer improves the model's performance. 2) Both entity knowledge and upos knowledge benefit the performance and entity knowledge is a little more important. 3) The performance dropped if we place the knowledge embedding in the header of the model. Perhaps the parameters we set in experiments are inappropriate or the model's header we designed is too simple.

6 Conclusion

We present two knowledge-enhanced methods in this model: the knowledge auxiliary text method and the knowledge embedding method. We design an answer extraction task pipeline to accommodate SemEval-2022 Task 09. Future work will involve incorporating the video data and predicting the answer of all pattern categories by model.

References

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pre-trained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178.

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

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL/IJCNLP (1)*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Shanshan Liu, Xin Zhang, Sheng Zhang, Hui Wang, and Weiming Zhang. 2019. Neural machine reading comprehension: Methods and trends. *Applied Sciences*, 9(18):3698.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

R Mervin. 2013. An overview of question answering system. *International Journal Of Research In Advance Technology In Engineering (IJRATE)*, 1:11–14.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Frank Rosenblatt. 1957. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. Colake: Contextualized language and knowledge embedding. In *COLING*.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Jingxuan Tu, Eben Holderness, Marco Maru, Simone Conia, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Roberto Navigli, and James Pustejovsky. 2022. Semeval-2022 task 9: R2VQ – competence-based multimodal question answering. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Ningyu Zhang, Shumin Deng, Xu Cheng, Xi Chen, Yichi Zhang, Wei Zhang, Huajun Chen, and Hangzhou Innovation Center. Drop redundant, shrink irrelevant: Selective knowledge injection for language pretraining.

Appendix

A QA-pair pattern categories

The QA pattern categories are shown in Table 1.

B Rule-based method descriptions

Follows are brief descriptions of the some categories' rule-based methods. The QA-pair pattern categories are described in Table 1.

Cat 1: locate two sentences in the text and judge whose position is in front.

Cat 6: count how many times the tool or ingredient, is mentioned in the question, appears in the text.

Cat 7: get ingredient and action information from the question, locate the ingredient before the action occurs, and get the habitat of the ingredient.

Cat 8: count how many times the habitat, which is mentioned in the question, appears in the text.

Cat 9: locate the RESULT target in the context and extract the whole sentence. Reorganize the text by the components from original text and knowledge auxiliary text.

Cat 10: locate the HABITAT target in the context and extract relevant ingredients.

ID	Question	Answer	Pct.
01	Pouring batter in and baking other side, which comes first?	the first event	15.9
02	How do you cut the stalks?	by using a knife	13.7
03	What should be added to the pan?	the kale	12.9
04	Where should you divide the dough?	floured surface	10.8
05	Where do you transfer the garlic?	to a paper towel	8.5
06	How many times is the spoon used?	2	8.3
07	Where was the quinoum before it was mixed into the wok?	bowl	5.8
08	How many teaspoons are used?	2	5.6
09	How did you get the aromatic mixture?	by adding the shallot and garlic	4.5
10	What's in the gratin?	the cheese	4.2
11	For how long do you add diced mushroom pieces?	after a few minutes	3.5
12	To what extent do you stir the pie	til syrup thickens	3.4
13	Why do you whip the egg?	to mix well	1.3
14	From where do you drain water?	potatoes	0.8
15	What do you mix sweetener with?	with 3/4 cup water	0.6
16	By how much do you cover the beans with water in a pot?	by 2 inches	0.1
17	What do you cut the rectangle into?	into 6 squares	0.05
18	How would you reduce oven temp?	slightly	0.01

Table 2: QA-pair pattern categories

SemEval 2022 Task 10: Structured Sentiment Analysis

Jeremy Barnes[1], **Laura Oberländer**[2], **Enrica Troiano**[2], **Andrey Kutuzov**[3]
Jan Buchmannn[4], **Rodrigo Agerri**[1], **Lilja Øvrelid**[3], **Erik Velldal**[3]

[1]HiTZ Center - Ixa, University of the Basque Country UPV/EHU
[2]IMS, University of Stuttgart
[3]LTG, University of Oslo
[4]Technical University of Darmstadt

{jeremy.barnes, rodrigo.agerri}@ehu.eus
{laura.oberlaender, enrica.troiano}@ims.uni-stuttgart.de
{andreku, liljao, erikve}@ifi.uio.no
buchmann@ukp.informatik.tu-darmstadt.de

Abstract

In this paper, we introduce the first SemEval shared task on *Structured Sentiment Analysis*, for which participants are required to predict all sentiment graphs in a text, where a single sentiment graph is composed of a sentiment holder, target, expression and polarity. This new shared task includes two subtracks (monolingual and cross-lingual) with seven datasets available in five languages, namely Norwegian, Catalan, Basque, Spanish and English. Participants submitted their predictions on a held-out test set and were evaluated on Sentiment Graph F_1. Overall, the task received over 200 submissions from 32 participating teams. We present the results of the 16 teams that provided system descriptions and our own expanded analysis of the test predictions.

1 Introduction

Affective computing is a fundamental step towards enabling human computer interaction (Picard, 1997), as human communication is filled with affective content which conveys a speaker's private state, *i.e.* their current mood, their emotional experiences, or their attitude towards a certain object of conversation. Along with emotion detection, sentiment analysis (Pang et al., 2002; Turney, 2002; Wiebe et al., 2005) is an important stepping stone towards this goal. On a more practical level, being able to automatically determine what people think about an idea, product, or policy is of interest to companies, governments, and private citizens.

In this paper, we describe the SemEval-2022 shared task on *Structured Sentiment Analysis*, which can be thought of as an information extraction problem in which one attempts to find all of the opinion tuples $O = O_i, \ldots, O_n$ in a text. Each opinion O_i is a tuple (h, t, e, p) where h is a **holder** who expresses a **polarity** p towards a **target** t through a **sentiment expression** e, implicitly defining pairwise relationships between elements of the same tuple. Liu (2012) argues that all of these elements are essential to fully resolve the sentiment analysis problem. Although early annotation efforts in sentiment analysis annotated for fine-grained sentiment (Wiebe et al., 2005; Toprak et al., 2010), most research on modeling sentiment *focuses either on a variety of sub-tasks* which avoid performing the full task, *e.g.* targeted (Hu and Liu, 2004), aspect-based (Pontiki et al., 2014), or end-2-end sentiment (Wang et al., 2016), *or instead relies on simplified and idealized tasks*, *e.g.* sentence-level binary polarity classification (Pang et al., 2002).

We argue that the division of fine-grained sentiment into these sub-tasks has become counterproductive, as reported experiments are often not sensitive to whether a given addition to the pipeline improves the overall resolution of sentiment, nor do they take into account the inter-dependencies of the various sub-tasks.

Motivated by this, we present the SemEval-2022 shared task on **Structured Sentiment Analysis**, which jointly predicts all elements of an opinion tuple and their relations.

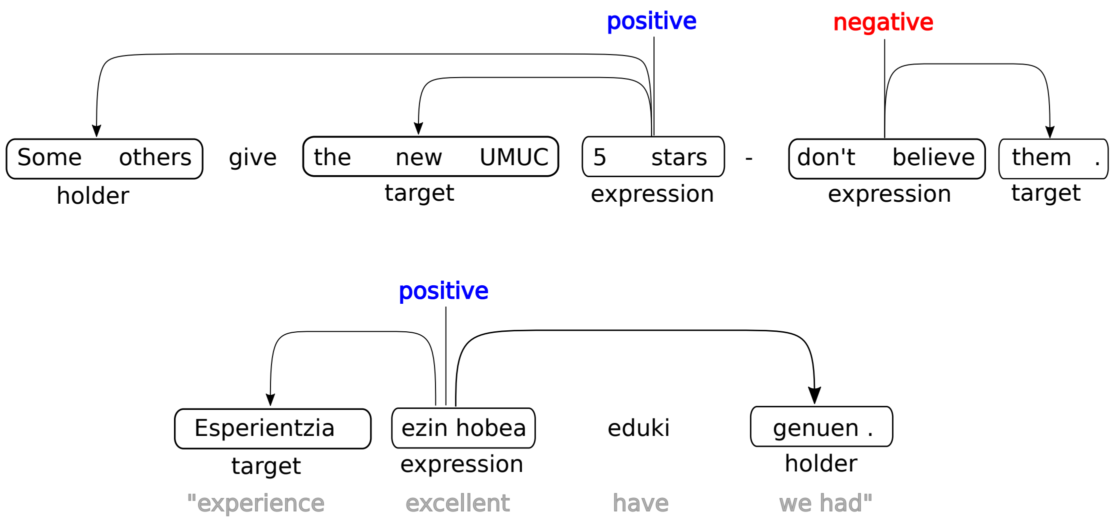


Figure 1: A structured sentiment graph (shown in English and Basque) is composed of a holder, target, sentiment expression, their relationships and a polarity attribute. Holders and targets can be null.

2 Related Work

The conceptual roots of Structured Sentiment Analysis can be found in early computational work on sentiment (Hu and Liu, 2004; Wiebe et al., 2005). Much research in the field has been motivated by the corpus compiled by Wiebe et al. (2005), who annotated English news wire documents with sentiment holders, targets, expressions, intensities, and other variables of interest. Subsets of these variables have been detected with linear (Choi et al., 2006; Yang and Cardie, 2012) and neural models (Katiyar and Cardie, 2016; Zhang et al., 2019), but the full task has never been performed simultaneously.

For instance, various SemEval shared tasks on *Aspect-Based Sentiment Analysis* (ABSA) (Pontiki et al., 2014, 2015, 2016) have focused on target extraction and polarity classification, and there have been research efforts to predict sentiment expressions as well (Wang et al., 2017). The models implemented for that purpose, however, neither resolve relations between expressions nor predict their polarity. The combination of targets, expressions and their polarity have been addressed for the recent task of *aspect sentiment triplet extraction* (Peng et al., 2019; Xu et al., 2020), but the resources that are used for this goal, which typically augment existing targeted datasets with polar expressions, suffer from a major limitation: they do not report annotation guidelines, procedures, or inter-annotator agreement, leaving the final quality of the data unclear.

To solve these issues and integrate all such perspectives, Barnes et al. (2021) proposed a holistic approach to sentiment. They cast the problem of structured sentiment as one of dependency parsing, they introduced specific metrics to evaluate automatic performance on this task, and developed a state-of-the-art structure-aware model. Further improvements were reported by Peng et al. (2021), who proposed a sparse fuzzy attention mechanism to deal with the sparseness of dependency arcs in the dependency models.

3 Task Description

The aim of this shared task is to predict all sentiment graphs in a text (see Figure 1), where a graph includes the elements of an opinion tuple (h, t, e, p). We proposed two subtasks, corresponding to monolingual structured sentiment and cross-lingual structured sentiment. Participants were free to participate in one or both setups, and they had the opportunity to submit a single run on each dataset.

Subtask 1: Monolingual structured sentiment. Models implemented for the first setup had to be trained and tested on the same language. We did not include a closed track, but we asked participants to detail all data used to train models and to make their training reproducible. This also allowed the teams to train multi-lingual models on all of the available training data for structured sentiment – a choice that was made by some of them.

Subtask 2: Cross-lingual structured sentiment. The second task required participants to train on other languages' train set and test on Spanish, Catalan, and Basque. Again, we allowed any further resource besides the train/dev sets for the test language provided with the shared task.

4 Data

The task contained seven datasets in five languages: Basque (**MultiBooked**$_{\text{EU}}$), Catalan (**MultiBooked**$_{\text{CA}}$) (Barnes et al., 2018), English, with data from **OpeNER**$_{EN}$ (Agerri et al., 2013), Multi-Perspective Question Answer corpus (Wiebe et al., 2005, **MPQA**) and Darmstadt Universities corpus (Toprak et al., 2010, **DS**$_{\text{Unis}}$), Norwegian (Øvrelid et al., 2020, **NoReC**$_{\text{Fine}}$) and Spanish (**OpeNER**$_{ES}$) (Agerri et al., 2013). We directly distributed the datasets in json format, with the exception of the last two corpora, as they have more restrictive licensing. For these, we provided links to the data such that participants could acknowledge their respective terms of use and include scripts to preprocess the data in a uniform manner. Table 1 provides an overview of the datasets and their sentiment annotations.

MultiBooked is a collection of hotel reviews in Basque and Catalan, written by users and collected from `booking.com`. The data was annotated for structured sentiment with polar expressions, targets, and holder labels, as well as polarity and intensity. Each dataset contains around 1,500 sentences, making them the smallest datasets in the shared task. However, these sentences are more densely annotated than some of the larger corpora. For guidelines and inter-annotator agreement, see Barnes et al. (2018).

OpeNER contains an opinion mining corpus of hotel reviews for six languages (de, en, es, fr, it, nl)[1]. For the purposes of the shared task, we only used the English and Spanish data (Agerri et al., 2013). The reviews were extracted from different booking sites from November 2012 to November 2013. Data collection was designed to ensure that different ratings and languages were included for a given hotel review.

The annotations regard opinion expressions, their respective holders and targets, their polarity and opinion strength. The guidelines were based on the work by Wiebe et al. (2005), which defines an opinion expression as a word (or combination of words) that expresses an attitude of the opinion holder towards a target. The corpus also specifies the relations between holders, targets and opinion expressions in opinion triplets[2].

MPQA annotates English news wire texts with a complex set of annotation types, *i.e.* *agent*, *expressive-subjectivity*, *direct-subjective*, *objective-speech-event*, *attitude*, and *target*. These types are also associated to a number of features and relations between one another. For the purpose of the shared task, we keep only the labels of *agent*, *target*, *direct-subjective*, as well as the *polarity* feature of direct-subjective, which respectively map to our holder, target, polar expression and polarity. We further normalize the polarities such that we have only *positive*, *negative*, and *neutral*. This is the second largest dataset, with a large number of holders, but relatively fewer targets and expressions.

DS$_{\text{Unis}}$ was initially published as part of the Darmstadt Service Review Corpus (DSRC) (Toprak et al., 2010). The DSRC contains reviews of online universities and services annotated with sentiment on the sentence level and fine-grained sentiment on the expression level. The **DS**$_{\text{Unis}}$ data that is part of the task contains only the university reviews, and discards the sentence-level annotations. Toprak et al. (2010) distinguish between polar facts and opinions in their annotation scheme. In order to map the data to the format of the shared task, this distinction is resolved.

NoReC$_{\text{Fine}}$ annotates a subset of the Norwegian Review Corpus (Velldal et al., 2018) for fine-grained sentiment, i.e., including polar expressions, targets and holders, as well as their polarity and intensity. The corpus contains annotations for more than 11k sentences (both subjective sentences and fact-implied ones) taken from professional reviews from a number of different domains, such as screen, music, literature, products and games (Mæhlum et al., 2019). Further details on the annotation procedure, guidelines and inter-annotator agreement can be found in Øvrelid et al. (2020).

5 Evaluation

The main metric for the task is Sentiment Graph F$_1$ (**SF**$_1$), which attempts to quantify how well a

[1]`https://github.com/opener-project`

[2]For more information about the annotation guidelines: `https://github.com/opener-project/opinion-domain-lexicon-acquisition/blob/master/annotation_guidelines/WP5-guidelinesReviews.pdf`

	sentences		holders			targets			expressions			polarity		
	#	avg.	#	avg.	max	#	avg.	max	#	avg.	max	+	neu	−
MultiBooked$_{\text{EU}}$	1,520	10.6	296	1.1	6	1,760	1.4	9	2,319	2.2	10	1,940	0	379
MultiBooked$_{\text{CA}}$	1,676	15.2	237	1.1	7	2,350	2.4	18	2,770	2.6	19	1,743	0	1,027
OpeNER$_{EN}$	2,492	14.8	413	1.0	3	3,843	1.8	21	4,149	2.4	21	2,981	0	1,168
OpeNER$_{ES}$	2,054	17.4	225	1.0	2	3,960	2.2	12	4,386	2.2	15	3,557	0	829
MPQA	10,048	23.3	2,265	2.7	40	2,437	6.3	50	2,794	2.0	14	1,082	465	1,059
DS$_{\text{Unis}}$	2,803	20.0	94	1.2	4	1,601	1.2	6	1,082	1.9	9	612	186	805
NoReC$_{\text{Fine}}$	11,437	16.9	1,128	1.0	12	8,923	2.0	35	11,115	5.0	40	7,547	0	3,557

Table 1: Statistics of the datasets, including number of sentences and average length (in tokens), as well as average and max lengths (in tokens) for holder, target, and expression annotations. Additionally, we include the distribution of polarity – restricted to positive, neutral, and negative – in each dataset.

model captures the full sentiment graph (see Figure 1). For **SF**$_1$ each sentiment graph is a tuple of (holder, target, expression, polarity). A true positive is defined as an exact match at graph-level, weighting the overlap between the predicted and gold spans for each element, averaged across all three h, t, e spans. We therefore allow some variability at the token-level (properly weighted), as long as the sentiment graph is predicted.

For precision, we weight the number of correctly predicted tokens divided by the total number of predicted tokens (for recall, we divide instead by the number of gold tokens). Correctly predicted tokens can also consist of empty holders and targets.

6 Baselines

We provided participants with two strong baselines: 1) a dependency graph prediction model, and 2) a sequence-labeling pipeline.

Dependency Graph The first baseline approaches sentiment graph prediction as a dependency graph prediction task, following Barnes et al. (2021). Each sentiment graph is converted to a head-final dependency representation, where we set the final token of the sentiment expression as a root node, the final token in each holder and token span as the head of the span, with all other tokens within that span as dependents. The labels simply denote the type of relation (target/holder) and for sentiment expressions, they additionally encode the polarity. After converting the data, we use a neural graph parsing model (Dozat and Manning, 2018), which learns to score each possible arc to predict the output structure simply as a collection of all positively scored arcs. The base of the network structure is a BiLSTM that creates contextualized representations $c_1, \ldots, c_n = \text{BiLSTM}(w_1, \ldots, w_n)$ where w_i is the concatenation of a word embedding, POS tag embedding, lemma embedding, and character embedding created by a character-based LSTM for the ith token. The contextualized embeddings are then processed by two feedforward neural networks (FNN), creating specialized representations for potential heads and dependents, and the scores for each possible arc-label combination are computed by a final bilinear transformation.

Sequence Labeling We also include a sequence labeling baseline. Specifically, this approach first trains three separate BiLSTM models to extract holders, targets, and expressions, respectively. It then trains a relation prediction model, which uses a BiLSTM with max pooling to create contextualized representations of 1) the full text, 2) the first element (either a holder or target) and 3) the sentiment expression. These three representations are then concatenated and passed to a linear layer followed by a sigmoid function. The training consists of predicting whether two elements have a relationship or not, converting the problem into a binary classification. During inference, the model starts by predicting all sub-elements. Next, it decides if these have a relationship (prediction > 0.5). Finally, the predictions are combined to form full sentiment graphs.

7 Results and Discussion

32 teams participated, with over 200 submissions in total for the evaluation period. The top 10 results for **Subtask 1** are shown in Table 2 (for **Subtask 2** in Table 3). Nearly all teams perform better than the baselines and the performance of the winning teams constitutes the new state of the art.

	NoReC	**MultiBooked**		**OpeNER**		**MPQA**	**DS**	
Team	NO	CA	EU	ES	EN	EN	EN	Avg.
zhixiaobao	52.9	72.8	73.9	72.2	76.0	44.7	49.4	63.1
MT-speech	52.4	72.8	73.9	74.2	76.3	41.6	48.5	62.8
Hitachi	53.3	70.9	71.5	73.2	75.6	40.2	46.3	61.6
SLPL	50.4	68.1	72.3	73.5	74.7	37.5	41.0	59.6
sixsixsix	48.3	71.1	68.1	68.6	72.7	37.9	37.3	57.7
KE_AI	48.3	71.1	68.1	68.6	72.7	36.4	37.3	57.5
SeqL	48.4	70.4	70.3	69.8	72.5	25.4	42.0	57.0
LyS_ACoruña	46.2	65.3	68.0	69.2	69.8	34.9	41.4	56.4
ECNU_ICA	49.6	68.4	68.6	62.3	67.6	35.1	49.0	56.1
ohhhmygosh	48.7	65.8	65.1	66.9	71.0	26.9	41.6	55.1
graph baseline	27.2	51.6	54.5	49.5	52.1	12.5	20.4	38.3
seq baseline	12.3	33.8	36.5	24.0	32.9	0.02	0.06	19.9

Table 2: Top 10 systems for the monolingual Sub-task 1 according to Sentiment Graph F_1.

Team	ES	CA	EU	Avg.
MT-speech	64.4	64.3	63.2	64.0
SLPL	61.8	56.2	58.4	58.8
Hitachi	62.8	60.7	52.7	58.7
sixsixsix	60.4	59.6	51.2	57.1
SeqL	58.9	59.3	51.6	56.6
ECNU_ICA	55.1	61.5	53.0	56.6
Mirs	61.7	54.4	52.2	56.1
LyS_ACoruña	57.0	55.4	50.9	54.4
OPI	56.4	58.6	44.4	53.1
KE_AI	56.1	55.2	46.3	52.5

Table 3: Top 10 systems for the cross-lingual Sub-task 2 according to Sentiment Graph F_1.

8 Summary of Participating Systems

In this section, we summarize the top three approaches and then further discuss some commonalities among the remaining teams.

8.1 The ZHIXIAOBAO submission

The best performing team on **Subtask 1** formulated the task similar to Barnes et al. (2021), using a dependency graph parsing approach. They deviate from the original in several important ways. First, they use either RoBERTa_Large (Liu et al., 2019) (for the English datasets) or XLM-RoBERTA_-Large (Conneau et al., 2020) (for non-English) as a feature extractor, rather than multilingual BERT base (Devlin et al., 2019) and further fine-tune the parameters of this model, rather than freezing it. Secondly, they introduce a new attention mechanism to help differentiate 'in span' and 'out of span' tokens, which helps dealing with tokens that are not a part of any sentiment span. Finally, they also use suffix masking for tokens which are broken into subtokens when computing the edge scores.

During experimentation, they found that removing the final LSTM layer from the Barnes et al. (2021) model gave improved performance. They also found that the 'in-label' approach was beneficial. Interestingly, they also found that XLM-RoBERTa often performed better than similarly sized monolingual BERT models, e.g. for Basque, Catalan, and Spanish.

8.2 The MT-Speech submission

The second best team in **Subtask 1** and best team in **Subtask 2** similarly used a dependency graph parsing approach with an XLM-RoBERTa_Large backbone. Given the rather small size of some of the datasets, this team proposes several data augmentation strategies which prove to be effective. The first is to exploit the Masked Language Modeling pre-training task of XLM-RoBERTa to augment the training data. They do this by randomly masking a small percentage of the words in a text which lie outside of the sentiment expression. They then sample up to 5 new sentences for training, putting a threshold to remove unlikely versions. Secondly, they further pre-train the language model on the training data, but using the Masked Language Model objective. They also use data from similar datasets in Portuguese and the English

	Rank		General Approach						Language Models				Others		
	mono	cross	graph	seq. label	offset pred.	QA	pipeline	dep. graph	large	FT domain	mono	XLM-mono	multi-task	data aug.	ensemble
zhixiaobao	1	-	✓					✓	✓		✓	✓		✓	
MT-speech	2	1	✓	aux.				✓	✓	✓		✓	✓	✓	
Hitachi	3	3	✓	✓			✓		✓		✓				✓
SLPL	4	2	✓	✓			✓								
LyS_ACoruña	8	8	✓					✓	✓		✓		✓	✓	
ECNU_ICA	9	6			✓		✓		?		✓				
ISCAS	10	-	✓			✓	✓		✓		✓	✓			✓
OPI	11	9		✓			✓		✓		✓	✓			
HITSZ-HLT	12	-			✓		✓		EN		✓				
MaChAmp	13	26		✓			✓		✓				✓		
Amex	14	-	✓									✓			
SenPoi	16	15		✓			✓		✓			✓			
ETMS@IITKGP	18	11			✓			✓	non-EN		✓	✓			
SPDB	21	14	✓											✓	✓
UFRGSent	22	18				✓	✓					✓			
SSN_MLRG1	27	-		✓			✓								

Table 4: Characteristics of submissions that submitted a system description. We show the rank for subtask 1 (mono) and 2 (cross), followed by the general approach, the use of language models, and other characteristics. FT domain = Fine-tune to the domain, mono = monolingual LM, XLM-mono = multi-lingual language model used for monolingual task.

SemEval Laptop dataset which has been automatically augmented with polar expressions. This data is converted to the structured sentiment format. Finally, they include several auxiliary tasks to predict the spans in a sentiment graph as sequence labeling tasks. They include final polarity classification auxiliary task, which they perform on the Catalonia Independence Corpus (Zotova et al., 2020).

8.3 Hitachi

The third team compare a graph prediction model (Graph) and a sequence-to-sequence (Seq2Seq) approach. Specifically, Graph includes a BIO sequence labeler for span extraction, followed by relation prediction with biaffine classifiers (Dozat and Manning, 2018). For the Seq2Seq model, they serialize the tuples and use large pretrained language models to predict these serialized tuples. As the task required providing token offsets, as a post-processing step they predict the text anchors using a word-alignment tool (Jalili Sabet et al., 2020).

Generally, they find that graph prediction performs better than Seq2Seq. In an extensive analysis, they find that Seq2Seq's need for an external alignment system is a hindrance if that information is truly necessary. On OpeNER, however, Graph is clearly better. Both approaches generally perform worse on examples with more opinions, although Graph is slightly more robust. Finally, they find that Graph is much faster to train, as there is no decoder. They conclude, however, that there is not enough evidence to conclusively show that Graph is better than Seq2Seq.

8.4 General Trends

We now discuss some general trends within the submissions and their possible effect on the results. We summarize these trends in Table 4.

General Approach: In general, the teams with the best performance all use graph prediction models. The top two teams maintained the dependency graph approach of Barnes et al. (2021). Several strong submissions successfully used a pipeline approach of a sequence labeling model to extract spans, followed by a relation prediction model. Three teams preferred offset prediction to the BIO sequence labelling, while two teams formulated the task as question answering.

Language Models Nearly all teams used some form of pre-trained language models to create contextualized token representations. One important factor seems to be the size of the language model used (Base or Large), as the top teams generally used XLM-RoBERTa_Large (Conneau et al., 2020). MT-Speech find that fine-tuning this model using the masked language model task on the provided training data improves the performance. The benefits of using a monolingual model seem to be language dependent, as for English or Spanish, many submissions found monolingual models best, while for Basque or Catalan, they often found that multi-lingual language models performed better.

Others Finally, two of the top teams used a multi-task learning approach to incorporate further information into their models. Several teams further perform data augmentation to minimize the impact of the smaller datasets. This is a particularly beneficial side-effect explicitly annotating all parts of the sentiment graph, as participants were able to make changes in a controlled way, such that the opinion itself was not changed. Three teams further included ensemble approaches with less success than in other tasks.

9 Further Analysis

To gain insight into the systems' performance, we move on to analyzing their mistakes and correct predictions. We start with a quantitative analysis to understand what types of errors appear most often in the submissions, and how they vary according to the different opinion spans (e.g., h/t), settings (monolingual/cross-lingual) and datasets.

Next, we relate the predictions made in the two sub-tasks to some structural properties of the data. This qualitative analysis aims at shedding light on what makes a tuple O_i easy/difficult to find, and whether the observations that apply to a sub-task generalize well to the other.

9.1 Quantitative Analysis.

For the quantitative analysis, we group the different error types defined by Oberländer and Klinger (2020) in the following way[3]: Too early (,), Too late (,), Other (,), Multiple ( ), False Positive (/), and False Negative ( ).

Figure 2 shows relative frequencies for each error type across all 10 winning teams in the monolingual setting (left) and cross-lingual setting (right)[4]. To obtain the relative frequencies, we flatten the annotations for each span type and treat either a continuous stream of annotated or of unannotated tokens as the base of our predictions. Where a prediction exactly aligns with an annotated span, it counts as a true positive, where there is no prediction on an unannotated part, as a true negative. For all other cases we count this as another error type as grouped above. True positives and negatives are not shown in the plot, but are considered when calculating the relative frequency.

We find that, generally, predicting the correct *holder* of an opinion in the monolingual setting is easier than predicting the *target* or the *polar expression*. This is to be expected and explained by the fact that the holders of an opinion are overall shorter in length, reducing the potential of making mistakes. Interestingly, this finding does not fully hold for the cross-lingual setting where there are more False Negatives ( ) and Multiple ( ) for *holder* than for *polar expression*. A finding which holds for both the monolingual and the cross-lingual setting is that the error type that occurs the most apart from False Positives (/) and False Negatives ( ) is Multiple ( ). Multiple occurs the most when models predict *polar expressions* for the monolingual setting and *target* for the cross-lingual setting. This is likely because the *polar expression* and *target* spans are typically longer, which gives the models more options to find several predictions within the span. The other type of errors are infrequent, notable being Too early (,) and Too late (,), both error types occurring unsurprisingly mostly for the *polar expression* spans.

Now we compare the box plots across the sub-tasks. Looking at the medians, we see that these are well separated with the median for relative frequencies of False Negatives ( ) and Multiple ( ) for all span types being lower for the monolingual than for the cross-lingual setting. The same pattern for False Positives (/) can be seen, with the exception of the span type *polar expressions*. The inter-quartile ranges are not very similar to each other while looking at pairs of the same type of errors across the two sub-tasks. We see this aspect in the lengths of the boxes, which differ quite a lot, especially notable here are *holders*, for which the boxes are two times in length for the cross-lingual setting. We observe that the error frequencies are generally more spread for the cross-lingual subtask, for False Negatives ( ) and Multiple ( ) holders, and False Positives (/) targets. For other span types the trend is not as clear. Also, the overall spreads across the settings are slightly greater for the cross-lingual setup for the same span types. However, looking at the overall spreads is perhaps less informative about dispersion of the frequencies of errors than the comparison of box lengths, because of the outliers we see for the mono-

[3]The top bar shows the gold span, while the bottom corresponds to the predicted span.

[4]Error frequencies across the top 10 systems can be found in Appendix (Table 6 and Table 7).

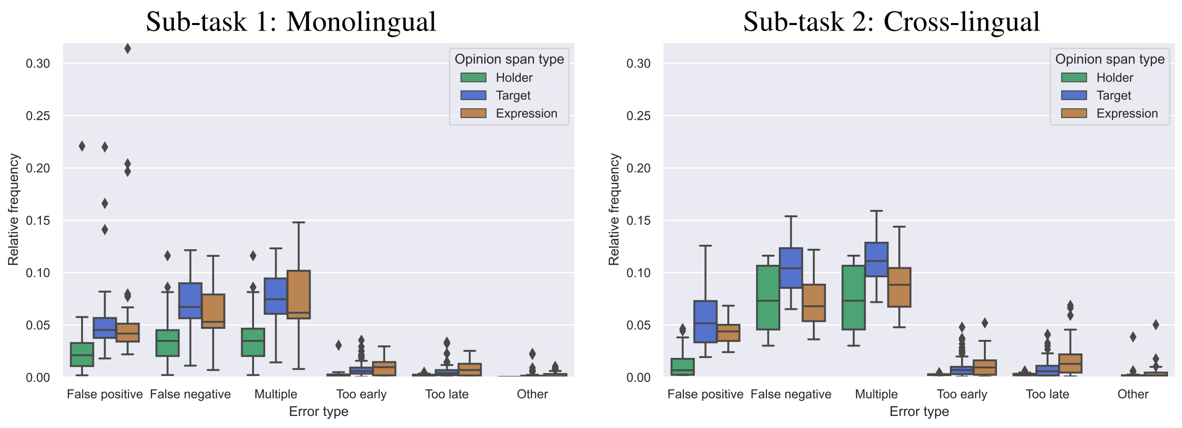


Figure 2: Relative frequencies for each error type for each span type across all teams and datasets.

lingual case. The exceptions are the box plots for the *polar expressions* in the case of the error type Multiple (▬▬). Regarding skewness, we observe generally a right-skewness for the monolingual setup, and no large skew for the cross-lingual setup. For instance, errors on *holders* are skewed to the right in the monolingual case, whereas for the cross-lingual setting, they are slightly skewed to the left. Similarly, *polar expressions* are also skewed to the left for the cross-lingual case and for *targets* the opposite effect can be noticed. Far-away outliers only exist for the monolingual setting, for False Positives (▪▫▪/▬) the far-away outliers are mostly from team ECNU_ICA and team sixsixsix for the **MPQA** and **NoReC$_{\text{Fine}}$** datasets. For the False Negatives (▭) and Multiple (▬▬), the far-away outlier is team ohhhmygosh for both error types on **MultiBooked$_{\text{EU}}$** dataset.

The analysis on the box plots evokes a further question: why are there so many False negatives (▭) for *holders* in the cross-lingual setup in comparison to the monolingual one? By inspecting the datasets, we make two observations that explain this result. First, across all datasets used for the cross-lingual sub-task there are only approximately 10% of instances annotated with *holders* and second, the fraction of non-empty *holders* is higher in the test data than in the train data (the numbers can be found in Table 5 in the Appendix).

9.2 Qualitative Analysis.

We now examine the extent to which the correct (or incorrect) identification of specific sub-parts of a sentiment graph (e.g. (h, t)) correlates with another (e.g., (e, p)), or with additional properties of the ground truth (e.g., sentiment intensity). So far, we considered a sentiment graph to be correctly predicted by one system if the system's output was a true positive (following the definition in Section 5). Now, to aggregate the results of all teams, a correctly predicted (ground-truth) graph is one which corresponds to a majority of true positives across teams. Table 8 in Appendix A reports a subsample of texts in which all O_i are predicted correctly (indicated as $+$) by most teams, and those in which most teams did not score a true positive for any of any tuple (marked as $-$).

Within-graph analysis. We begin by focusing on the predictions of h, t, and e. We observe that a successful detection of an opinion holder and target approximates the exact identification of the opinion expression. This is particularly evident in the cross-lingual setup: 75% of ground-truth opinion tuples that have a match in the holder and target for more than half the teams have an exact match in the expression (81% if also the h, t match is exact). In the monolingual setup, this happens in 70% cases (73% for exact h, t span matches).

Typically, if the spans of holder, target and expression are properly recognized, the polarity of such expression is too. It happens for 95% of correctly predicted h, t, e in the cross-lingual setting, for 93% in the monolingual one. These numbers corroborate the relational nature of the span types involved in a structured sentiment task: determining a holder and a target is crucial to establish the opinion linking the two, and hence, its polarity.

Polarity-intensity link. Going beyond the component of a graph, we investigate the relationship between O_i and the intensity of sentiment – a label that was not evaluated in the competition but which characterizes opinions in the used corpora. Figure 3 shows two example distributions of ground-truth

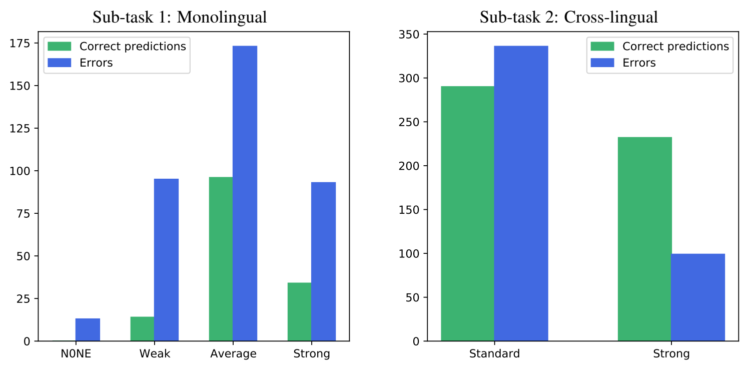


Figure 3: Counts of opinions predicted correctly (with respect to h, t, e, and p) and incorrectly (wrong p): example results from **MPQA** in Sub-task 1, and **OpeNER**$_{ES}$ in Sub-task 2.

opinions for which more than half the teams identified the right (h, t, e, p) (correct predictions), and those where for which (h, t, e) was identified but p was not (errors).

For the setups with only two strength labels, intensity seems to play a role in the prediction of polarity: the submitted systems are consistent in identifying the true polarity of most of the tuples associated to a strong sentiment expression. We see that, for instance in **OpeNER**$_{ES}$, correct predictions are 70% of opinions labelled as "Strong"; the same happens only for 46% of O_i tuples with a standard intensity level. Similar patterns are observed through all corpora in the cross-lingual setup, as well as in the monolingual task based on the same corpora, and **OpeNER**$_{EN}$ (see Appendix Figure 4). This suggests that the recognition of polarity correlates to the intensity with which sentiment is expressed: a sentiment conveyed with higher intensity tends make the prediction "easier".

The link becomes less clear for corpora with non-binary intensity annotation schemas, *i.e.* **MPQA**. Most polarities were not properly recognized across all sentiment strengths. Still, the proportion of wrong-to-correct instances is typically higher with milder intensity degrees (e.g., in **MPQA**, 87% of weak-intensity O_i tuples correspond to errors).

Opinion span sparsity. One feature that differentiates tuples is the sparsity of their spans in the text (e.g., in "*L' habitació un pèl petita per un 5 estrelles*" t and e encompass the whole sentence, while in "*La situació perquè tenim la nostra filla vivint molt a prop . segurament repetirem*", which contains no holder and target, the expression involves only the last two words). Therefore, we observe whether and how the systems' decisions change together with the sparsity of opinion spans, computed as $1 - \frac{\sum \#\,\text{tokens}(h,t,e)}{\#\,\text{tokens}(text)}$ (Figure 5 in Appendix shows the distribution of sparsity values in the two tasks).

On average, sparsity is lower for correct predictions than for errors. It is 0.73 and 0.72 for the missed h, t, and e spans in the cross-lingual and monolingual tasks, 0.67 (cross-lingual) and 0.66 (monolingual) for the predicted h, t, and e spans. This suggests that spans covering larger parts of the text are easier to predict, while errors tend to occur with labels that more scattered in the text.

A comparable observation can be drawn relative to the number of opinions in a text. Opinion tuples coming from texts with a higher number of h, t, e relations appear harder to predict: correct predictions occur for tuples that come from texts containing on average 3.4 O_is (in the cross-lingual setup, and 3 in the monolingual task), while errors arise with spans present in text that have on average 4.16 opinions (in the cross-lingual setup, and 3.27 in the other).

10 Conclusion

We proposed to cast sentiment analysis as a structured prediction problem, explicitly predicting the four main elements, and provide seven preprocessed datasets in five languages. Graph prediction models powered by pre-trained language models generally performed best, although several pipeline sequence labelling models also performed well. An analysis of the errors shows that false negatives and predicting shorter spans are the most common errors, while when models correctly predict the holder and target, they generally predict everything correctly. Finally, both more intense polar expressions and spans that cover much of the text are easier to predict.

Acknowledgements

We would like to thank Stephan Oepen for motivation and guidance during the first stages of the shared task. Rodrigo Agerri is funded by the RYC-2017–23647 fellowship and by the ANTIDOTE - EU CHIST-ERA project (PCI2020-120717-2), from the Agencia Estatal de Investigación through the INT-Acciones de Programación Conjunta Internacional (MINECO) 2020 call.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 431–439, Sydney, Australia. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

Arzoo Katiyar and Claire Cardie. 2016. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 919–929, Berlin, Germany. Association for Computational Linguistics.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

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

Petter Mæhlum, Jeremy Barnes, Lilja Øvrelid, and Erik Velldal. 2019. Annotating evaluative sentences for sentiment analysis: a dataset for Norwegian. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, Turku, Finland.

Laura Ana Maria Oberländer and Roman Klinger. 2020. Token sequence labeling vs. clause classification for English emotion stimulus detection. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 58–70, Barcelona, Spain (Online). Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2019. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis.

Letian Peng, Zuchao Li, and Hai Zhao. 2021. Sparse fuzzy attention for structured sentiment analysis.

Rosalind W. Picard. 1997. *Affective Computing*. MIT Press, Cambridge, MA.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Haris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Peter Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Erik Velldal, Lilja Øvrelid, Eivind Alexander Bergem, Cathrine Stadsnes, Samia Touileb, and Fredrik Jørgensen. 2018. NoReC: The Norwegian Review Corpus. In *Proceedings of the 11th edition of the Language Resources and Evaluation Conference*, Miyazaki, Japan.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 616–626, Austin, Texas. Association for Computational Linguistics.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3316–3322. AAAI Press.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349, Online. Association for Computational Linguistics.

Bishan Yang and Claire Cardie. 2012. Extracting opinion expressions with semi-Markov conditional random fields. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1335–1345, Jeju Island, Korea. Association for Computational Linguistics.

Meishan Zhang, Qiansheng Wang, and Guohong Fu. 2019. End-to-end neural opinion extraction with a transition-based model. *Information Systems*, 80:56 – 63.

Elena Zotova, Rodrigo Agerri, Manuel Nuñez, and German Rigau. 2020. Multilingual stance detection in tweets: The Catalonia independence corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1368–1375, Marseille, France. European Language Resources Association.

A Appendix

A.1 Quantitative Analysis.

A.2 Qualitative Analysis.

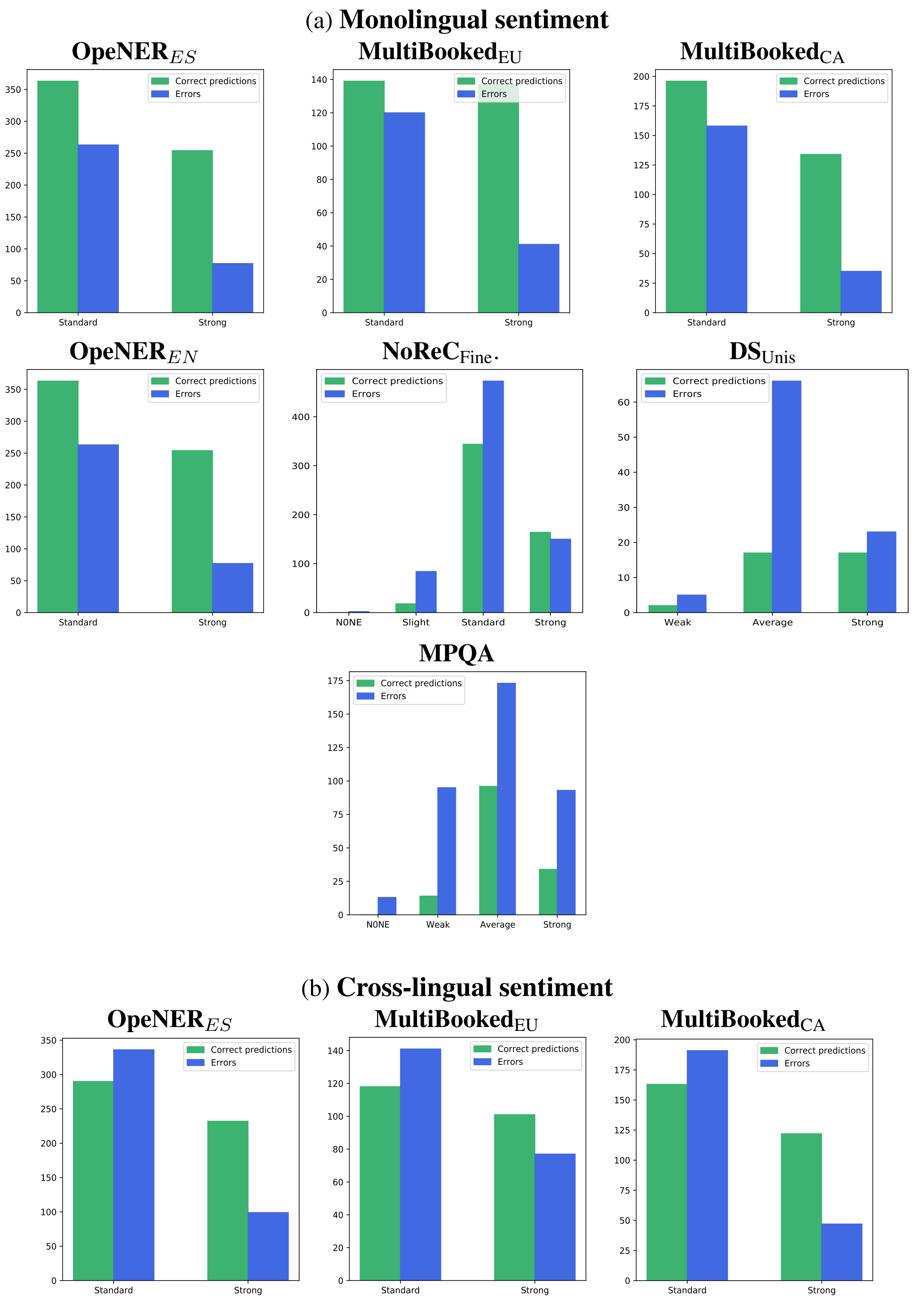


Figure 4: Counts of (h, t, e, p) tuples that are predicted correctly ($>$ half the systems correctly identified all graph components) and incorrectly (wrong p), across intensity levels: (a) Monolingual task, (b) Cross-lingual.

	Holder		**Target**		**Expression**	
Dataset	train	test	train	test	train	test
MultiBooked$_{\text{CA}}$	8.50%	10.13%	85.72%	82.98%	100.00%	100.00%
MultiBooked$_{\text{EU}}$	12.21%	13.27%	76.06%	75.74%	100.00%	100.00%
OpeNER$_{ES}$	5.79%	5.85%	90.34%	88.71%	100.00%	100.00%
OpeNER$_{EN}$	9.22%	11.33%	92.89%	91.68%	100.00%	100.00%
DS$_{\text{Unis}}$	7.82%	9.23%	100.00%	100.00%	100.00%	100.00%
NoReC$_{\text{Fine}}$	10.63%	8.91%	80.23%	80.40%	100.00%	100.00%
MPQA	84.06%	83.78%	86.81%	89.19%	100.00%	100.00%

Table 5: The fraction of non-empty annotations for each span type across datasets.

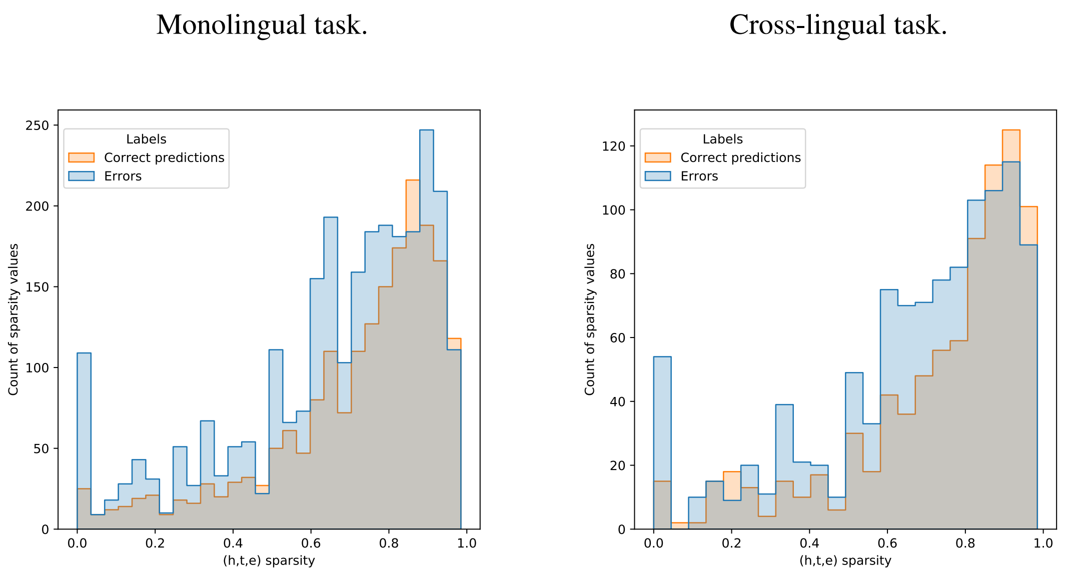


Figure 5: Distribution of sparsity values for h, t, e, spans predicted correctly or incorrectly by most teams in the two setups.

		NoReC			MultiBooked						OpeNER						MPQA			DS		
Team	**Error type**	NO			CA			EU			EN			ES			EN			EN		
		H	T	E	H	T	E	H	T	E	H	T	E	H	T	E	H	T	E	H	T	E
zhixiaobao	False negative	2	9	8	4	7	5	3	5	4	4	5	5	2	6	5	5	5	5	2	9	9
	False positive	2	5	5	1	4	4	3	6	3	1	4	3	2	4	4	3	3	3	1	7	6
	Multiple	2	10	12	4	7	7	3	5	5	4	6	6	2	7	7	5	5	6	2	9	9
	Other	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Too early	0	1	4	0	2	2	0	2	3	0	1	2	0	2	3	0	1	2	0	1	0
	Too late	0	1	3	0	1	2	0	0	2	0	1	2	0	1	3	0	1	1	0	1	1
Cong666 (MT-speech)	False negative	2	9	8	5	6	4	3	4	4	3	5	4	2	6	5	5	5	5	2	9	8
	False positive	2	5	4	1	5	4	4	6	4	2	4	3	4	4	3	3	3	3	1	8	8
	Multiple	2	10	11	5	7	5	3	5	5	3	5	5	2	7	6	5	6	6	2	9	8
	Other	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Too early	0	1	4	0	2	3	0	1	4	0	2	3	0	2	3	0	1	2	0	1	1
	Too late	0	1	3	0	1	2	0	1	2	0	1	2	0	2	3	0	1	1	0	1	1
Hitachi	False negative	1	9	7	4	6	5	4	5	4	3	5	4	2	5	3	4	4	4	2	10	10
	False positive	2	7	5	1	5	4	3	5	3	2	4	3	3	5	4	4	4	4	0	6	5
	Multiple	1	9	11	4	7	6	4	6	5	3	6	5	2	5	5	5	5	5	2	10	10
	Other	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
	Too early	0	1	4	0	2	3	0	1	3	0	1	2	0	3	3	1	1	2	0	0	1
	Too late	0	1	3	0	1	2	0	1	3	0	1	3	0	2	4	0	1	1	0	1	1
colorful	False negative	2	10	9	5	7	5	4	6	4	4	7	5	2	7	4	5	6	5	2	10	9
	False positive	1	5	4	1	6	5	3	4	4	1	4	2	4	4	3	2	2	3	1	7	8
	Multiple	2	11	14	5	8	6	4	7	6	4	7	6	2	8	6	5	7	6	2	10	10
	Other	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Too early	0	2	3	0	1	2	0	1	2	0	1	2	0	2	3	0	1	1	0	0	1
	Too late	0	1	3	0	2	2	0	1	2	0	1	2	0	2	3	0	0	1	0	1	1
sixsixsix	False negative	1	9	8	4	6	5	4	6	5	5	6	5	2	5	5	5	5	5	2	11	11
	False positive	3	7	6	2	5	4	4	5	5	1	4	3	3	4	4	4	4	5	0	7	7
	Multiple	1	9	11	4	7	6	4	7	5	5	6	6	2	6	6	5	5	5	2	11	11
	Other	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	Too early	0	1	4	0	2	3	0	2	3	0	2	3	0	3	4	0	1	2	0	0	3
	Too late	0	1	3	0	2	2	0	1	2	0	0	2	0	2	4	0	1	1	0	1	1
KE_AI	False negative	1	9	8	4	6	5	4	6	5	5	6	5	2	5	5	5	5	5	2	11	11
	False positive	3	7	6	2	5	4	4	5	5	1	4	3	3	4	4	4	3	5	0	7	7
	Multiple	1	9	11	4	7	6	4	7	5	5	6	6	2	6	6	5	5	5	2	11	11
	Other	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	Too early	0	1	4	0	2	3	0	2	3	0	2	3	0	3	4	0	1	2	0	0	3
	Too late	0	1	3	0	2	2	0	1	2	0	0	2	0	2	4	0	1	1	0	1	1
SeqL	False negative	2	11	9	4	7	5	4	7	5	4	6	6	2	7	5	-	-	-	2	11	11
	False positive	2	5	4	1	4	4	2	3	3	2	3	3	4	3	4	-	-	-	1	5	5
	Multiple	2	11	12	4	7	6	4	7	5	4	7	6	2	8	6	-	-	-	2	11	11
	Other	0	1	1	0	1	0	0	0	0	0	2	0	0	2	0	-	-	-	0	0	0
	Too early	0	2	5	0	3	4	0	3	4	0	3	3	0	5	4	-	-	-	0	0	1
	Too late	0	1	3	0	3	3	0	4	3	0	3	2	0	4	4	-	-	-	0	0	1
LyS_ACoruña	False negative	2	11	9	4	8	5	4	7	5	5	8	7	2	8	6	5	6	5	2	10	10
	False positive	2	6	5	1	5	6	3	4	4	1	3	3	3	4	3	3	3	4	1	8	8
	Multiple	2	11	14	4	9	7	4	8	6	5	9	8	2	9	7	6	7	6	2	11	10
	Other	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	Too early	0	2	3	0	1	2	0	1	2	0	1	2	0	2	3	0	1	1	0	1	0
	Too late	0	1	3	0	1	2	0	1	3	0	1	2	0	2	3	0	1	1	0	1	0
ECNU_ICA	False negative	0	4	4	-	-	-	2	8	7	4	8	6	3	10	7	1	1	1	0	5	5
	False positive	6	22	20	-	-	-	4	8	5	4	5	5	3	4	4	22	14	31	1	17	20
	Multiple	0	4	7	-	-	-	2	8	8	4	8	7	3	11	8	1	1	1	0	5	5
	Other	0	0	0	-	-	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Too early	0	1	4	-	-	-	0	1	3	0	2	3	0	2	3	0	1	1	0	0	2
	Too late	0	1	1	-	-	-	0	1	2	0	1	2	0	1	3	0	1	0	0	1	0
ohhhmygosh	False negative	5	12	12	9	10	10	12	9	11	8	9	9	8	10	10	7	6	8	2	11	11
	False positive	0	5	4	0	5	3	0	4	3	0	4	2	0	4	3	4	3	3	0	6	6
	Multiple	5	12	15	9	10	11	12	9	12	8	10	10	8	10	11	7	7	8	2	11	12
	Other	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Too early	0	1	3	0	2	2	0	1	2	0	1	2	0	2	3	3	1	1	0	0	0
	Too late	0	1	2	0	1	2	0	1	2	0	1	2	0	2	3	0	1	0	0	0	1

Table 6: Relative frequencies (%) of error types at the span level across the top 10 systems for the monolingual Sub-task 1. H: holder; T: target; E: polar expression.

Team	Error type	OpeNER			MultiBooked					
		ES			CA			EU		
		H	T	E	H	T	E	H	T	E
Cong666 (MT-speech)	False negative	3	12	6	6	7	5	9	7	6
	False positive	4	2	3	1	8	5	0	7	4
	Multiple	3	12	7	6	8	7	9	8	8
	Other	0	0	0	0	0	0	0	0	0
	Too early	0	2	3	0	3	4	0	2	3
	Too late	0	1	5	0	1	2	0	1	4
colorful	False negative	4	13	7	7	11	8	12	12	11
	False positive	2	2	3	0	5	4	0	2	3
	Multiple	4	14	9	7	12	10	12	13	12
	Other	0	0	0	0	0	0	0	0	0
	Too early	0	1	2	0	3	2	0	1	3
	Too late	0	1	5	0	3	4	0	1	4
Hitachi	False negative	4	11	5	7	8	6	12	10	11
	False positive	3	3	4	0	7	5	0	6	3
	Multiple	4	11	6	7	8	8	12	11	13
	Other	0	0	0	0	0	0	0	0	1
	Too early	0	2	3	0	4	3	0	2	2
	Too late	0	2	6	0	2	3	0	1	4
sixsixsix	False negative	3	9	5	7	7	4	11	8	9
	False positive	3	5	4	1	7	7	1	11	6
	Multiple	3	10	6	8	7	5	11	9	10
	Other	0	0	1	0	0	0	0	0	0
	Too early	0	2	3	0	2	5	0	2	3
	Too late	0	3	6	0	3	5	0	2	6
SeqL	False negative	3	12	5	7	8	4	11	10	8
	False positive	5	2	2	0	6	5	0	7	5
	Multiple	3	12	6	7	9	6	11	11	10
	Other	0	4	5	0	1	0	0	1	2
	Too early	0	5	6	0	8	6	0	4	3
	Too late	1	4	8	0	3	3	0	3	8
ECNU_ICA	False negative	5	13	8	-	-	-	7	7	8
	False positive	1	3	4	-	-	-	1	13	5
	Multiple	5	14	9	-	-	-	7	8	10
	Other	0	0	1	-	-	-	0	0	0
	Too early	0	2	3	-	-	-	0	2	4
	Too late	0	2	4	-	-	-	0	1	3
Mirs	False negative	5	13	7	7	10	7	12	11	12
	False positive	2	2	3	1	7	5	0	6	5
	Multiple	5	14	9	7	12	9	12	12	14
	Other	0	0	0	0	0	0	0	0	0
	Too early	0	1	2	0	3	1	0	1	1
	Too late	0	1	5	0	2	3	0	1	4
LyS_ACoruña	False negative	3	15	9	6	9	6	11	10	9
	False positive	4	2	3	1	8	5	1	8	5
	Multiple	3	16	10	6	10	8	11	11	12
	Other	0	0	1	0	0	0	0	0	1
	Too early	0	1	2	0	3	3	0	1	2
	Too late	0	2	5	0	1	3	0	2	5
OPI	False negative	4	14	5	6	10	5	12	14	11
	False positive	2	3	4	0	5	5	0	3	3
	Multiple	4	14	7	6	10	7	12	15	13
	Other	0	0	0	0	0	0	0	0	0
	Too early	0	1	3	0	4	3	0	1	2
	Too late	0	2	5	0	2	3	0	0	4
KE_AI	False negative	6	13	7	8	10	7	11	11	10
	False positive	2	4	4	0	5	5	0	8	4
	Multiple	6	13	8	8	11	8	11	12	11
	Other	0	0	2	0	0	1	0	0	1
	Too early	0	2	4	0	3	4	0	2	4
	Too late	0	3	6	0	5	5	0	1	7

Table 7: Relative frequencies (%) of error types at the span level across the Top 10 systems for the cross-lingual Sub-task 2. (H: holder; T: target; E: expression)

Cross-lingual sentiment		
MultiBooked$_{\text{CA}}$	$+$	A una habitació , mancaba la teuleta de nit i la persiana estava trencada . hi habia una batidora que no funcionaba .
	$-$	La ubicació , la decoració i la comoditat de els llits
MultiBooked$_{\text{EU}}$	$+$	Langileak oso jatorrak , laguntzeko prest eta profesionalak .
	$-$	Iruzkinak euskaraz egiteko aukera ez izatea .
OpeNER$_{ES}$	$+$	Muy amables y simpaticos , bastante limpio todo .
	$-$	Los de recepción te aconsejan un poco sobre donde ir y que linea de metro o de bus coger para ir a los destinos .
Monolingual sentiment		
MultiBooked$_{\text{CA}}$	$+$	El director de l' hotel molt desagradable .
	$-$	Tot , ben situat a 10 minuts de la ciutat vella .
MultiBooked$_{\text{EU}}$	$+$	Harreran zeuden langileen arreta ez zen onena izan .
	$-$	Bigarren aukera gisan izan bazen ere , zorionekoa izan zen Bergenenen alde hartu genuen aukera .
OpeNER$_{ES}$	$+$	Ideal para pasar un fin de semana de turismo por la capital . .
	$-$	Muy bien ubicado para ver el musical
OpeNER$_{EN}$	$+$	Because of renovation work probably my room was not fully ready .
	$-$	But the receptionist immediately offered me an improved room with riverside view .
DS$_{\text{Unis}}$	$+$	Courses are rigorous and challenging .
	$-$	For the sake of time - I will not comment on quality of education - let's assume it is OK compared to it's competitors .
NoReC$_{\text{Fine}}$	$+$	Dette er dog lett å tilgi når spillbarheten er så overlegen som den er her .
	$-$	Og spesielt fabelaktig er det når Claire og Jamie er i selskap på slottet i Versailles i andre episode .
MPQA	$+$	This announcement was met with unanimous condemnation by the international media .
	$-$	That is a bitter pill to swallow in a thoroughly non-militaristic society such as ours , where the clash of weapons provokes healthy reactions of repulsion .

Table 8: Examples from different setups and corpora. In the texts marked as $+$, the graphs O_i are predicted correctly by more than half the teams. Items marked as $-$ contain graphs for which the majority of teams missed the correct prediction.

AMEX AI Labs at SemEval-2022 Task 10: Contextualized fine-tuning of BERT for Structured Sentiment Analysis

Pratyush Sarangi, Shamika Ganesan, Piyush Arora, Salil Rajeev Joshi
American Express AI Labs, Bangalore, India
{pratyush.k.sarangi|shamika.ganesan|piyush.arora1|salilrajeev.joshi}@aexp.com

Abstract

We describe the work carried out by AMEX AI Labs on the *structured sentiment analysis* task at SemEval-2022. This task focuses on extracting fine grained information w.r.t. to source, target and polar expressions in a given text. We propose a BERT based encoder, which utilizes a novel concatenation mechanism for combining syntactic and pretrained embeddings with BERT embeddings. Our system achieved an average rank of 14/32 systems, based on the average scores across seven datasets for five languages provided for the monolingual task. The proposed BERT based approaches outperformed BiLSTM based approaches used for structured sentiment extraction problem. We provide an in-depth analysis based on our post submission analysis.

1 Introduction

In this paper we present the work done by AMEX AI Labs on the *structured sentiment analysis* monolingual task at SemEval-2022 (Barnes et al., 2022). Structured sentiment analysis (SSA) focuses on performing fine grained analysis and extracting opinion tuples from a given input text (Barnes et al., 2021). An example is presented in Fig. 1.

Input sentence: *Some others give the new UMUC 5 stars - don't believe them*

Expected outputs are two tuples:

- **Tuple1** - ("source": *"Some others"*, "target": *"the new UMUC"*, "polar expression": *"5 stars"*)
- **Tuple2** - ("source": *""*, "target": *"them"*, "polar expression": *"don't believe"*).

Aspect based sentiment analysis is a popular and widely researched topic in the NLP community (Wagner et al., 2014; Pontiki et al., 2015; Schouten and Frasincar, 2015). However, the SSA task goes beyond traditional aspect based sentiment

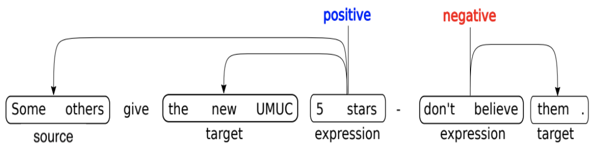


Figure 1: Sample sentence, source: (Barnes et al., 2021)

mining. The main challenges in SSA task are as follows:

- Focus is on correct **span detection** of source, targets, and polar expression elements as shown in Fig. 1.
- Correctly identifying **relationships** between source, target, and polar expression elements, as illustrated in the example above.
- Building a robust, generalized approach that is applicable *across domains* as well as *across languages*.

Most of the state-of-the-art (SOTA) approaches have explored joint learning, span based multi-task learning, BiLSTM encoders or CRF based approaches for SSA related tasks (Barnes et al., 2021; Li et al., 2019; Zhao et al., 2020; Chen et al., 2020). We propose a contextualized BERT based approach called 'CBERT', for capturing context information effectively and address the main challenges for SSA task as discussed above.

The main contributions of our work are:

- We propose a BERT based encoding approach and combine syntactic and pre-trained embeddings using a novel concatenation approach to extract and find opinion tuples effectively.
- We performed a detailed error analysis highlighting the main challenges of SSA task and suggest future directions for building a more efficient and effective system for SSA.

The outline of this paper is as follows: Section 2 describes the prior work in the area of SSA. Section 3 presents an abstract overview of our proposed system. Section 4 provides a detailed description of our CBERT system. Section 5 describes the datasets, tools and resources used for the experimental setup. Section 6 present the main results of the proposed system and describes an in-depth error analysis. Section 7 describes the main conclusions and leanings.

2 Background

Structured Sentiment Analysis (SSA) aims at capturing the natural hierarchical structure of different aspects, their corresponding sentiments in a sentence and the correlation between them. Almars et al. (2017) leveraged a hierarchical tree structure on the aspect terms, analyses the opinions on those aspect terms and connects them to the corresponding evidence for the same. Choi et al. (2006) identify two types of opinion-related entities — expressions of opinions and sources of opinions — along with the linking relation that exists between them. To identify the joint existence of opinion terms, they had utilized integer linear programming approach. Over years, this task was recognized further as Aspect Based Sentiment Analysis which focuses more on the precise identification of aspects and their corresponding sentiments (Chen et al., 2014), (Liu et al., 2012). Other intermediate works also focus on transition based end-to-end opinion extraction approaches (Zhang et al., 2019) which ignore the polarity classification subtask. Target Sentiment Analysis, which is another modified form of SSA focuses on extracting only sentiment targets and classifying their polarity (Jiang et al., 2011), (Mitchell et al., 2013).

Yang and Cardie (2012) is a strong baseline which suggests that the use of Conditional Random Fields could enhance the aspect terms prediction, given a smaller dataset with annotations, for training. During this time, some prominent research works discussed the advantage of semantic dependency parsing (Dozat and Manning, 2018), (Kurtz et al., 2020) which was leveraged for SSA task and for the establishment of the state-of-the-art SSA model which was proposed by (Barnes et al., 2021). This work formulates the SSA task as a dependency parsing problem and predicts all tuple components as a dependency graph. We aim to extend this aforementioned work and propose a modified architecture to improve the results across various language datasets.

3 Model Overview

For the SSA task, prior works have deployed separate models for detecting various elements (source, target and polar expression) (Barnes et al., 2022). However, this results in information regarding prediction of individual elements not being passed on to other element predictions.

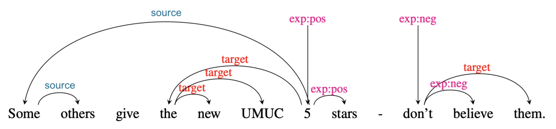


Figure 2: Sample sentence : head-first representation (Barnes et al., 2021)

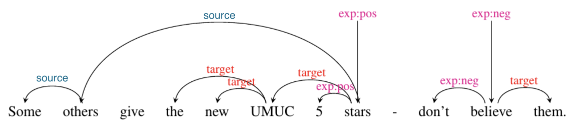


Figure 3: Sample sentence : head-final representation (Barnes et al., 2021)

For leveraging a joint learning technique, (Barnes et al., 2021) suggest two sets of approaches, *viz.*, head-first and head-final representation. *Head-first* considers spans of opinion elements using the first token of each element, *i.e.*, head token first, as shown in Fig. 2. *Head-final* considers the last token, *i.e.*, head token last, as shown in Fig. 3. Our proposed architecture was inspired by this joint learning technique and is elaborated in Section 4.

SSA task involves prediction of opinion elements along with the relationship between them, referred to as arc prediction. In this paper, we provide a comparison between two methods - VBERT, which uses only BERT embeddings and CBERT, which uses BERT embeddings along with concatenation of head and dependent BiLSTM blocks, detailed in Section 4. A comparison of results from these two approaches are provided in Section 6.

4 System Description

In this section we provide a detailed description of our proposed CBERT system. Fig. 4 presents the main architecture, comprising of seven main components, detailed below respectively.

4.1 Encoding Block

The *BERT context* shown in Fig. 4 represents BERT's role as an encoder. Our proposed model uses a BERT base multilingual cased[1] model (Devlin et al., 2018) which has t transformer layers to calculate the fine-tuned BERT vector from the input tokens, $x = (x_1, x_2, x_3, ..., x_n)$ of n padded sequence length. The output BERT fine-tuned vector is shown as, $b = (b_1^t, b_2^t, ..., b_n^t) \in R^{n*dim_b}$ where dim_b denotes it's dimension.

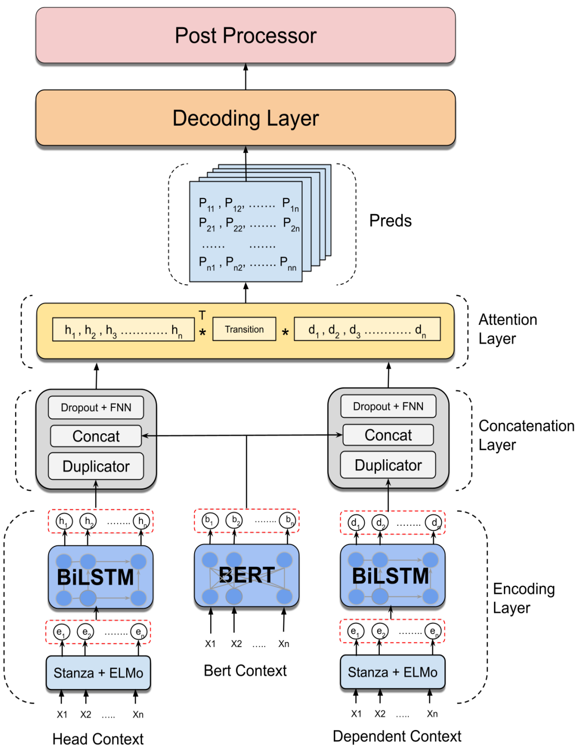


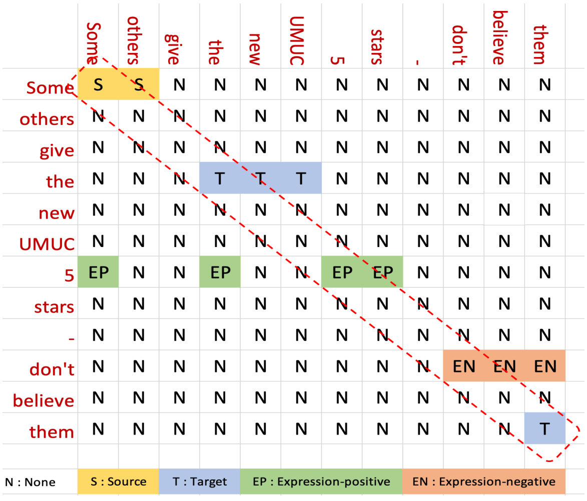
Figure 4: CBERT Architecture

The *Head Context* and *Dependent Context* as shown in Fig. 4, represent independent BiLSTM blocks where we fine-tune the syntactic and pretrained context in our CBERT system. However, in the VBERT system we do not have the aforementioned independent BiLSTM blocks (for more details refer Section A.4).

The input embedding in the BiLSTM block is a concatenation of Stanza[2] pretrained syntactic vectors (Qi et al., 2020) (U-pos, Lemma, Depparse) and ELMo[3] (Peters et al., 1802) embedding represented as, $e = (e_1, e_2, ..., e_n) \in R^{n*dim_e}$ where dim_e is the dimension of the input embedding. After passing these vectors through two independent identical BiLSTM blocks, we get the head and dependent fine-tuned vectors $h = (h_1^r, h_2^r, ..., h_n^r) \in R^{n*dim_c}$ and $d = (d_1^r, d_2^r, ..., d_n^r) \in R^{n*dim_c}$ where dim_c is the dimension of the fine-tuned vector and r is the number of recurrent layers in the BiLSTM block.

4.2 Target Encoding

We follow a head-first encoding (Barnes et al., 2021) which entails adding a label map to each position in the head-dependent 2D mapping, each label showing which category from ***(Expression-positive, Expression-negative, Source, Target, None)*** it belongs to. In the 2D mapping as shown in Fig. 5, heads are shown as rows, dependents as columns and a non ***None*** label shows if head-dependent relation exists or not. For denoting span the starting token of a certain span becomes the head and the other tokens as the dependents, as shown in Fig. 5, where *"the"* is linked to *("the","new","UMUC")*. Similarly, the head token of ***(Expression-positive, Expression-negative)*** labels point to the head tokens of ***(Source, Target)***, if a relation between them exists. For eg., *"5"* being linked to *"the"* as shown in Fig. 5. We also experimented with alternative encoding approaches mentioned in Section A.2



	Some	others	give	the	new	UMUC	5	stars	-	don't	believe	them
Some	S	S	N	N	N	N	N	N	N	N	N	N
others	N	N	N	N	N	N	N	N	N	N	N	N
give	N	N	N	N	N	N	N	N	N	N	N	N
the	N	N	N	T	T	T	N	N	N	N	N	N
new	N	N	N	N	N	N	N	N	N	N	N	N
UMUC	N	N	N	N	N	N	N	N	N	N	N	N
5	EP	N	N	EP	N	N	EP	EP	N	N	N	N
stars	N	N	N	N	N	N	N	N	N	N	N	N
-	N	N	N	N	N	N	N	N	N	N	N	N
don't	N	N	N	N	N	N	N	N	N	EN	EN	EN
believe	N	N	N	N	N	N	N	N	N	N	N	N
them	N	N	N	N	N	N	N	N	N	N	N	T

N : None | S : Source | T : Target | EP : Expression-positive | EN : Expression-negative

Figure 5: Encoding Structure

4.3 Concatenation layer

In this layer, we combine the BERT fine-tuned vector with the head and dependent fine-tuned vectors separately. The head and dependent vectors are based on a space based tokenizer while the BERT

[1]https://huggingface.co/bert-base-multilingual-cased
[2]https://stanfordnlp.github.io/stanza/index.html
[3]https://allenai.org/allennlp/software/elmo

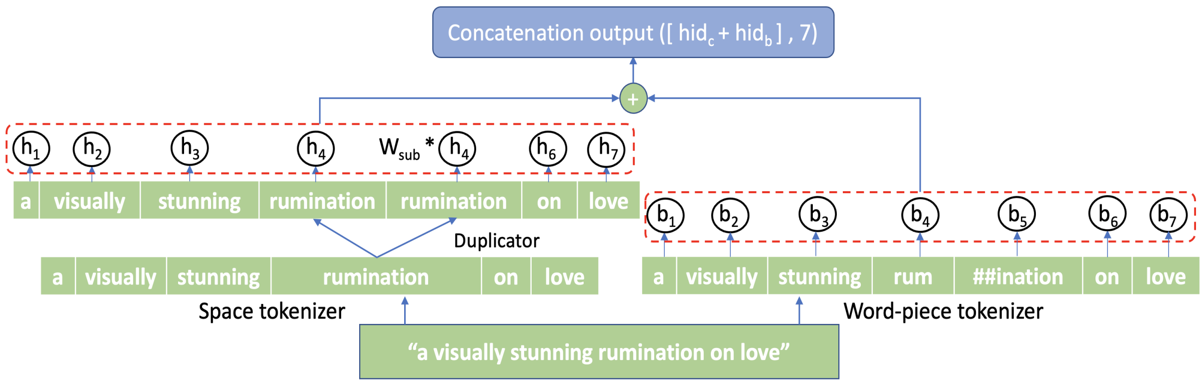


Figure 6: Concatenation of fine-tuned BERT with head and dependent fine-tuned vectors

vector is based on word-piece (Wu et al., 2016) tokenizer. Some of these space delimited tokens are further split by word-piece into sub-tokens. For eg., as shown in Fig. 6, the parent token *"rumination"* is split into sub-tokens *("rum","##ination")*. So we duplicate the parent token's vector,v as weighted vectors in the Duplicator as shown in Fig. 4. Higher weight is assigned to base-token *"rum"* and lower to sub-token *"##ination"*, as we want to put more significance on the base-token of the word for identifying span offsets correctly. Initializing sub-tokens with relatively low weighted vector instead of null vector helps us carry the base-word context till the sub-word. Following are the vector equations,

$$v = (v_1, v_2, ..., v_n)$$
$$v_{sub} = (W_{sub} * v_1, W_{sub} * v_2, ..., W_{sub} * v_n)$$
$$v_{base} = (W_{base} * v_1, W_{base} * v_2, ..., W_{base} * v_n)$$

Here v represents the parent token vector, v_{sub} the sub-token vector and v_{base} represents the base-token vector. The lower constant weight for sub-tokens is denoted by $W_{sub} \in [0, 1]$ compared to $W_{base} = 1$ assigned to base-token. The weight is decided after doing multiple runs with variations in W_{sub} as shown in Fig. 8. We choose $W_{sub} = 0.4$ empirically, as we get higher average scores in our experiments on the dev set.

4.4 Attention layer

The Concatenation layer outputs are then passed into the Attention layer which uses a transition matrix of $Shape(U) = dim_h * labels$. It transforms the head and dependent inputs to a 3D tensor matrix predicting the head-dependent relation arcs and labels jointly as represented in Fig. 4 as the Attention layer. The output is $Preds \in R^{n*n*labels}$ where $labels$ is the number of predicted labels and n is the padded sequence length. (for more details refer Section A.3)

4.5 Loss Function

We use a weighted cross-entropy function where the weights are assigned for each of the labels, l with respect to the target label frequency. We found out that the frequency of ***None*** label vs other labels was unbalanced. Hence, we conducted different runs with the following weight distribution $w = (w_{None}, 1, 1, 1, 1)$ by varying w_{None}. We choose the weights $w = (0.8, 1, 1, 1, 1)$ empirically, as we get higher average scores in our experiments on the dev set, as shown in Fig. 9.

$$loss_n = -w_{y_n} * log(\frac{exp(x_{n,y_n})}{exp(\sum_{l=1}^{L} x_{n,l})})$$
$$loss = -\sum_{n=1}^{N} \frac{loss_n}{\sum_{n=1}^{N} w_{yn}}$$

where x_n is the input probabilities, y_n is the corresponding target label, N is the batch length of input, w are the weights and L is the number of labels.

4.6 Decoding

In this subsection, we describe the methodology behind decoding the Attention layer output into the opinion pairs or tuples. As shown in Algorithm 1, in lines 2 to 12, we move along the main diagonal to find the predicted spans with labels of each category. Each of these spans have their starting offset denoted by the head pointing to itself as shown in Fig. 5. The end offset of a span is the last consecutive occurrence of the same tag as the head. For the next step shown in lines 13 to 22, we look at expression head's dependents to find if any of them are heads to other spans. This signifies a relation arc between spans.

Algorithm 1 Algorithm for Decoding the scores to tuples

Input: $Preds$, the 3D tensor output which contains the predicted label with maximum class probability for every head-dependent token pair.
Output: $Arcs$, which contain all the possible Source-Target-Expression tuples.

```
Require: $Preds \neq null$
 1: Initialize $Tar$, $Src$, $Exp$ and $Arcs$ sets to empty
 2: while Left index $l \leq length(Preds)$ and Right index $l < r \leq length(Preds)$ do
 3:     if $Preds(l,l) = T$ and all $Preds(l,r) = Preds(l,r-1)$ then
 4:         Add tokens $(t_l, ...t_r)$ to $Tar$
 5:     end if
 6:     if $Preds(l,l) = S$ and all $Preds(l,r) = Preds(l,r-1)$ then
 7:         Add tokens $(t_l, ...t_r)$ to $Src$
 8:     end if
 9:     if $Preds(l,l) \in (EN, EP)$ and all $Preds(l,r) = Preds(l,r-1)$ then
10:         Add tokens $(t_l, ...t_r)$ to $Exp$
11:     end if
12: end while
13: while Left index $l \leq length(Preds)$ and Right index $r \leq length(Preds)$ do
14:     if $Preds(l,l) \in (EN, EP)$ and $Preds(l,l) = Preds(l,r)$ and $l \neq r$ then
15:         if $t_r \in Tar$ then
16:             $Arcs \leftarrow (Tar(t_r), Exp(t_l))$
17:         end if
18:         if $t_r \in Src$ then
19:             $Arcs \leftarrow (Src(t_r), Exp(t_l))$
20:         end if
21:     end if
22: end while
```

4.7 Post processing

For post processing, in the generated tuples from the decoding layer, we remove overlapping Source or Target spans from edges of Expression spans. As shown in Example 1, "experience" is removed from the polar expression.

Example 1: *Thank you for such a wonderful experience.*
Raw tuple - ("source": "", "target": "*experience*", "polar expression": "*wonderful experience*")
Post processed tuple - ("source":"", "target": "*experience*", "polar expression": "*wonderful*").

5 Experimental Setup

5.1 Datasets & Evaluation Metric

The shared task has seven datasets across five languages that are used for final evaluation and ranking of submitted systems MPQA (Wiebe et al., 2005), NoRec (Øvrelid et al., 2020), Multi-book (Barnes et al., 2018), Opener (Agerri et al., 2013) and Darmstadt (Toprak et al., 2010). We submitted our system for monolingual task, where we fine-tuned our model for each individual dataset mentioned above.

The experiments conducted as a part of this work use Sentiment-F1 (SF1) score (Barnes et al., 2021) as the evaluation metric, as used in SemEval-2022 Shared Task 10. This metric defines true positive as an exact match for each element of the opinion terms, averaging the overlap in predicted and annotated spans for each element across source, target and polar expressions.

5.2 Experimental Settings

We use the Pytorch[4] implementation of BERT base multilingual cased model pretrained on the 104 languages. The model has $t = 12$ transformer layers, the hidden size dim_h is 768 and 12 self-attention heads. The padding length used for encoding is 128. We use ELMo embeddings which are domain-general with 256-dimensions, pre-trained with $800M$ tokens and $28M$ parameters. The Pytorch BiLSTM implementation has hidden dimension of 64, number of layers as 2 and dropout probability as 0.3. The Concatenation layer has a output dimension as 32 and dropout probability of 0.2. The batch size is set to 16 for Train and 8 for devset. We adopt an Adam optimizer (Kingma and Ba, 2014) with learning rate of $2e-5$ and 500 warmup steps. After conducting train-dev runs for maximum 100 epochs we select best epoch based on Sentiment-F1 score on dev-set. A detailed view of these experiments have been included in Fig. 7 in Appendix. We conduct these experiments on a DGX server consisting of 8 Nvidia Tesla V100-SXM2 with 16GB V-RAM. We use multiple GPU's using the DataParallel[5] Pytorch class.

6 Experimental Results and Analysis

The detailed results of final submission are as shown in Table 1. The proposed CBERT and VBERT models performed relatively better than BiLSTM based head-first and head-final approaches. CBERT performed 9% relatively better than VBERT, supporting the hypothesis that the concatenation of contextualized information with syntactic information is more effective.

Our system achieved an average rank of 14/32 systems, based on the average scores across seven datasets. An in-depth analysis reveals that although

[4] `https://huggingface.co/transformers/v1.2.0/`
[5] `https://pytorch.org/docs/stable/generated/torch.nn.DataParallel.html`

Dataset	Head-first	Head-final	VBERT	CBERT	*Median*	*Best System*
Opener_en	0.524	0.542	0.576	**0.634**	0.623	0.760
MPQA	0.218	0.218	0.218	**0.283**	0.367	0.447
Darmstadt_unis	0.181	0.209	0.271	**0.320**	0.342	0.494
Opener_es	0.480	0.552	0.537	**0.595**	0.539	0.722
NoRec	0.254	0.272	0.315	**0.343**	0.329	0.529
Multibook_ca	0.529	0.543	0.595	**0.634**	0.525	0.728
Multibook_eu	0.501	0.536	0.564	**0.559**	0.478	0.739
Average Scores	0.384	0.410	0.439	**0.481**	0.458	0.631

Table 1: Results on the test set across seven datasets for five languages, scores of the final system is in bold face

our model performed quite good for five datasets over the median system, for two datasets *MPQA* and *Darmstadt_unis* our proposed system did not perform relatively well, thus hampering the average scores. The best system scores are also relatively quite high, than our proposed CBERT system. One of the potential reasons for relatively lower performance for two datasets could be the existence of excessive number of neutral labels in these two datasets, which was not separately handled in the experimental setup, as discussed in Section 3.2.

Following is a qualitative analysis to identify main challenges in the CBERT system.

- **Boundary Detection:** It was observed that there were differences in the source, target and polar expression boundaries based on punctuation marks, stopwords or adjectives. For eg., *It is really very basic.* contains polar expression as *very basic* in its gold annotations whereas CBERT predicted as *really very basic.*

- **Tuple Formation:** In sentences that contain a single target with multiple polar expressions, CBERT predicts them as a single tuple. For eg., *The rooms are clean and functional.* In this sentence, there is one target, *the rooms* and its two polar expressions are *clean* and *functional.* Gold annotations provided for the sentence have two separate tuples for polar expressions whereas CBERT predicts them as a single tuple.

- **Differences in Gold Annotation:** In the datasets, some data annotations exist only for texts which have an explicit indication towards the theme of the dataset. For instance, the *NoRec* dataset is a collection of restaurant reviews. Hence, the data which do not have an explicit indication towards restaurant related comments are not annotated. However, CBERT does not distinguish between themes and produces predictions for such texts as well. For eg., *As a gold member I enjoyed an upgrade to a very large room with lots of floor space* . does not have any annotations. However, CBERT predicts Source as *I*, Target as *room* and Polar Expressions as *enjoyed, very large, lots of floor space*.

A detailed set of examples are shown in Table 2.

7 Conclusion and Future Scope

This paper discusses the proposed system and experimental results on SSA task at SemEval-2022. The proposed model called CBERT, assumes a joint learning technique using BERT-base multilingual model embeddings along with contextualized embeddings created using pretrained Stanza vectors and pretrained ELMo embeddings. The arc prediction between opinion term elements are achieved using an Attention layer. The proposed CBERT approach performed relatively better than BiLSTM approaches and VBERT approach. We find that using BERT based encoding, along with concatenation of contextualized information with syntactic information is more effective for SSA task.

For improving CBERT, leveraging 2D CRFs (Zhu et al., 2005) can be explored for better span detection over longer texts. Different joint learning architecture on top of CBERT's encoding like SDRN (Chen et al., 2020) can be leveraged. Approaches to fine-tune models for specific tasks (Zhao et al., 2020) is also a potential area for exploration.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Es-*

pañola para el Procesamiento del Lenguaje Natural, volume 51, pages 215–218.

Abdulqader Almars, Xue Li, Xin Zhao, Ibrahim A. Ibrahim, Weiwei Yuan, and Bohan Li. 2017. Structured sentiment analysis. pages 695–707.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402.

Jeremy Barnes, Andrey Kutuzov, Laura Ana Maria Oberländer, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6515–6524, Online. Association for Computational Linguistics.

Zhiyuan Chen, Arjun Mukherjee, and B. Liu. 2014. Aspect extraction with automated prior knowledge learning. In *ACL*.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 431–439, Sydney, Australia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 151–160, Portland, Oregon, USA. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Robin Kurtz, Stephan Oepen, and Marco Kuhlmann. 2020. End-to-end negation resolution as graph parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 14–24, Online. Association for Computational Linguistics.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019. Exploiting BERT for end-to-end aspect-based sentiment analysis. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 34–41, Hong Kong, China. Association for Computational Linguistics.

Kang Liu, Liheng Xu, and Jun Zhao. 2012. Opinion target extraction using word-based translation model. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, page 1346–1356, USA. Association for Computational Linguistics.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654, Seattle, Washington, USA. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

ME Peters, M Neumann, M Iyyer, M Gardner, C Clark, K Lee, and L Zettlemoyer. 1802. Deep contextualized word representations. arxiv 2018. *arXiv preprint arXiv:1802.05365*, 12.

Maria Pontiki, Dimitrios Galanis, Harris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 486–495.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.

Kim Schouten and Flavius Frasincar. 2015. Survey on aspect-level sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(3):813–830.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Joachim Wagner, Piyush Arora, Santiago Cortés Vaíllo, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. 2014. Dcu: Aspect-based polarity classification for semeval task 4. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 223–229.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

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

Bishan Yang and Claire Cardie. 2012. Extracting opinion expressions with semi-Markov conditional random fields. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1335–1345, Jeju Island, Korea. Association for Computational Linguistics.

Meishan Zhang, Qiansheng Wang, and Guohong Fu. 2019. End-to-end neural opinion extraction with a transition-based model. *Inf. Syst.*, 80:56–63.

He Zhao, Longtao Huang, Rong Zhang, Quan Lu, and Hui Xue. 2020. SpanMlt: A span-based multi-task learning framework for pair-wise aspect and opinion terms extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3239–3248, Online. Association for Computational Linguistics.

Jun Zhu, Zaiqing Nie, Ji-Rong Wen, Bo Zhang, and Wei-Ying Ma. 2005. 2d conditional random fields for web information extraction. In *Proceedings of the 22nd international conference on Machine learning*, pages 1044–1051.

A Appendix

A.1 Additional Experimental Settings

- In Fig. 7, each of the values correspond to SF1 scores obtained over devset, post to training on Training set (train/dev). It was observed that scores are consistent at 100 epochs.

- In Fig. 8, we ran train/dev experiments for 60 epochs to record SF1 scores by varying the sub-token weight w_{sub} to find optimal w_{sub} weights to be used in the Concatenation layer.

- In Fig. 9, we conducted train/dev experiments for 60 epochs to record SF1 scores by varying weight for ***None*** label w_{None} to find optimal weight to be used in weighted cross-entropy function.

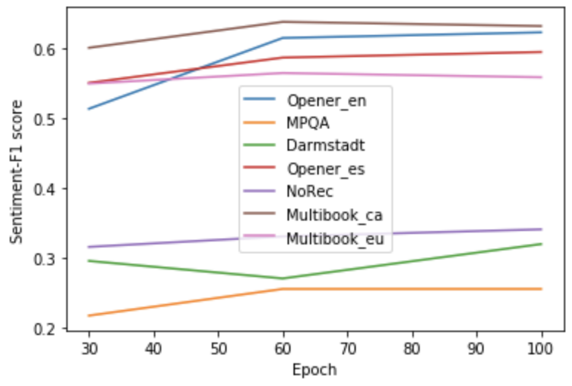


Figure 7: Graph showing Sentiment-F1 scores for our proposed model against the number of epochs.

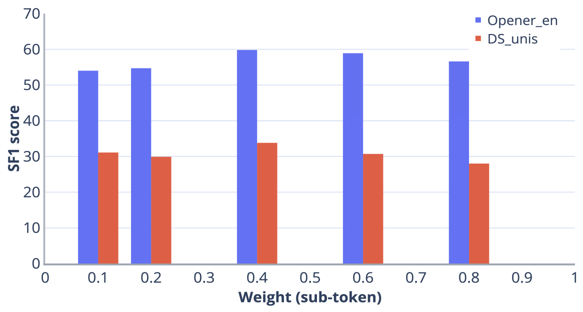


Figure 8: Sub-token weight vs devset SF1

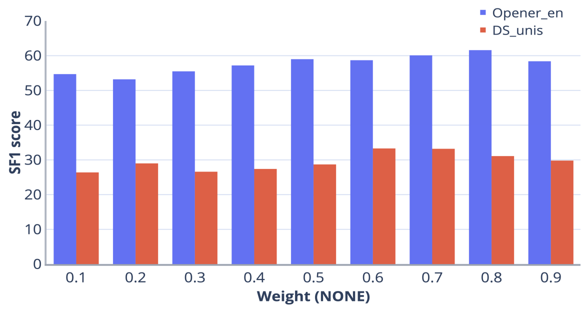


Figure 9: Weight(None) vs devset SF1

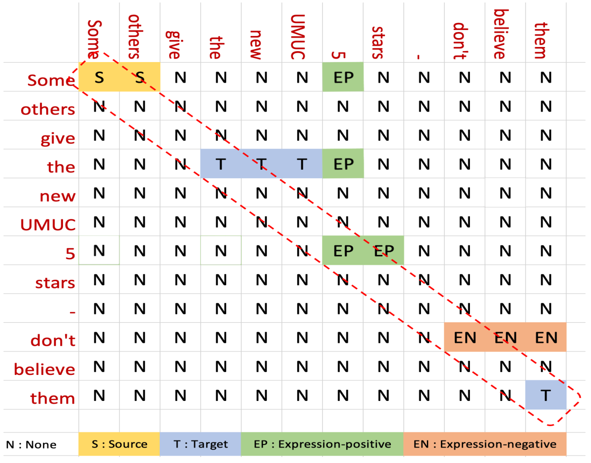
A.2 Alternative encoding schemes

We introduce another scheme for encoding where we mirror the labels of the lower triangular matrix to the upper triangular matrix along the main diagonal using results from our original encoding scheme as shown in Fig. 10. This helps us reduce the size of Attention layer's output $Preds$ by half,

Error Category	Remarks	Example
Boundary Detection	Based on Adjectives	For, *The first floor 24 hr bar was well run and no matter what time of day there was always a waiter on hand to serve .* CBERT predicts Polar Expression as *there was* whereas the annotaions contain *there was always*
	Based on Punctuations	For, *Robbed in elevator of hotel !* CBERT predicts Polar Expression as *Robbed in elevator of hotel* whereas annotations contain punctuation
	Based on Stopwords	For, *The best about this hotel is its location .* CBERT predicts Polar Expression as *The best about* whereas annotations contain *The best*
	Long Sentences	For, *There are good conference rooms with all necessary infrastructure ,good location allowing you to have nice evening , pool , restaurants , internet , 5 minutes to airport everything you need to do a business and relax after that .* CBERT predicts Source and Target efficiently but fails to detect Polar Expressions
Tuple Formation		For *Relax and enjoy* CBERT predicts two Polar Expressions separately as *Relax*, *enjoy* whereas annotations contain *Relax and enjoy* as single expression
Gold Annotations		For, *No words for this country and its people .* no annotations were provided

Table 2: Examples for Error Analysis categories

as we can always mask the lower half while applying the loss function.



	Some	others	give	the	new	UMUC	5	stars	-	don't	believe	them
Some	S	S	N	N	N	N	EP	N	N	N	N	N
others	N	N	N	N	N	N	N	N	N	N	N	N
give	N	N	N	N	N	N	N	N	N	N	N	N
the	N	N	N	T	T	T	EP	N	N	N	N	N
new	N	N	N	N	N	N	N	N	N	N	N	N
UMUC	N	N	N	N	N	N	N	N	N	N	N	N
5	N	N	N	N	N	N	EP	EP	N	N	N	N
stars	N	N	N	N	N	N	N	N	N	N	N	N
-	N	N	N	N	N	N	N	N	N	N	N	N
don't	N	N	N	N	N	N	N	N	N	EN	EN	EN
believe	N	N	N	N	N	N	N	N	N	N	N	N
them	N	N	N	N	N	N	N	N	N	N	N	T

N : None | S : Source | T : Target | EP : Expression-positive | EN : Expression-negative

Figure 10: Upper Triangular encoding

A.3 Attention layer

The output of Attention layer is represented as, $y = (y_{none}, y_{exp-pos}, y_{exp-neg}, y_{tar}, y_{src}) \in R^{n*n*labels}$ which is a 3D tensor where y_{label} is a 2D tensor corresponding to a certain label, $labels$ is the number of predicted labels and n is the padded input sequence length. The matrix transformations done in the attention module (Bi-linear label Attention) (Barnes et al., 2021) are shown below,

$$score(h_i, d_j) = h_i^T * U * d_j \text{ where}$$

$$h_i = Concat_{head}(c_i),$$
$$d_i = Concat_{dep}(c_i)$$
$$Shape(U) = dim_h * labels$$

A.4 VBERT overview

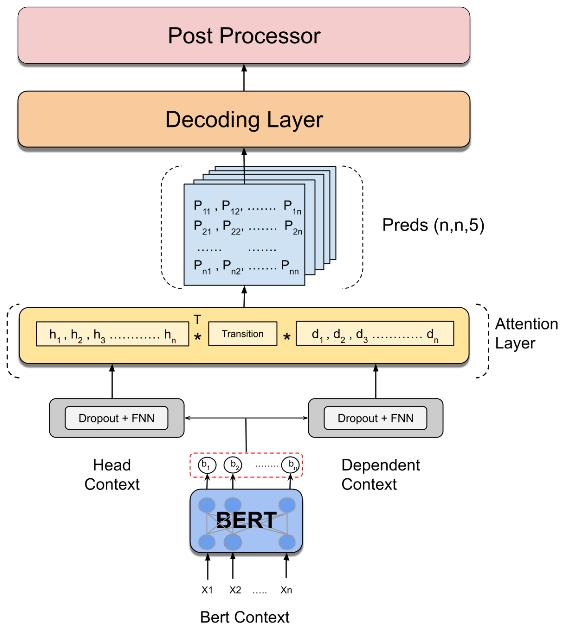


Figure 11: VBERT Architecture

This architecture shown in Figure 11 uses BERT as an encoder and splits the BERT output into head and dependent context.

ISCAS at SemEval-2022 Task 10: An Extraction-Validation Pipeline for Structured Sentiment Analysis

Xinyu Lu[*], **Mengjie Ren**[*], **Yaojie Lu**, **Hongyu Lin**
Chinese Information Processing Laboratory
Institute of Software, Chinese Academy of Sciences, Beijing, China
181303216@yzu.edu.cn, mjren@bupt.edu.cn,
{yaojie2017,hongyu}@iscas.ac.cn

Abstract

ISCAS participated in both sub-tasks in SemEval-2022 Task 10: Structured Sentiment competition. We design an extraction-validation pipeline architecture to tackle both monolingual and cross-lingual sub-tasks. Experimental results show the multilingual effectiveness and cross-lingual robustness of our system. Our system is openly released on: https://github.com/luxinyu1/SemEval2022-Task10/.

1 Introduction

Aspect-based Sentiment Analysis (ABSA) aims to detect the fine-grained sentiment tendency lying underneath texts. After decades of development, this area has formed a large family of tasks. Nevertheless, many of them are too simple or overlap with each other. Meanwhile, the popular evaluation resources are limited both in number and linguistic diversity. SemEval-2022 Task 10 (Barnes et al., 2022) is proposed to unify different sub-tasks in ABSA and introduces new metrics, new datasets on different languages to better evaluate methods in this area. Task 10 challenges its participants to extract opinion quadruple $(holder, target, expression, polarity)$ from texts across English, Spanish, Basque, Catalan and Norwegian in monolingual (Sub-task 1) or cross-lingual (Sub-task 2) manners. Figure 1 provides two aspect-level annotations in a same sentence.

We applied an extraction-validation pipeline system and participated in both sub-task. Our system ranked at 10^{th} in 32 teams on the monolingual task without using extra data, and achieved competitive performance on the cross-lingual task. Besides, the proposed pipeline can be employed universally in monolingual and cross-lingual scenarios.

[*]This work was finished during their internship at ISCAS-CIP Lab.

2 Background

2.1 Task Definition

Task 10 is formalized as detecting all opinion tuples $O = O_i, ..., On$ in given text s. Concretely, each opinion O_i is a quadruple (h, t, e, p), denoting a ***holder*** who expresses a ***polarity*** $\in$ {Positive, Neutral, Negative} towards a ***target*** through a sentiment ***expression***. It's worthy to note that, *h*, *t*, *e* can be empty in this task. Following Cai et al. (2021), tuples with empty values are regard as implicit opinions in this system description paper. For the example in Figure 1, the quadruple "(–, them, don't believe negative)" is an implicit opinion.

2.2 Related Work

Aspect-based Sentiment Analysis Recently, there has been a large body of work focusing on different sub-tasks of ABSA. Generally we divide these sub-tasks into two categories: atomic and compound. Atomic ones take single element (e.g., t, e, or p) as the output and most of them can be treated as a sequence tagging problem (Li and Lam, 2017; Xu et al., 2018; Li et al., 2018; Wu et al., 2020b; Pouran Ben Veyseh et al., 2020). The compound ones need to find pairs (e.g., (t, p)), triplets (e.g., (t, e, p)) or even quadruples (e.g., (h, t, e, p)) from the inputs. Some works (Peng et al., 2020; Xu et al., 2021) use pipeline architecture to extract the elements separately and then make combinations; meanwhile some works use Seq2Seq models (Yan et al., 2021; Zhang et al., 2021) or unified tagging schemes (Mitchell et al., 2013; Zhang et al., 2015) to solve these sub-tasks in an end-to-end manner.

Pre-trained Language Models Pre-trained Language Models (PLMs) are deep neural networks pre-trained on large-scale corpora. Unlike traditional static word embedding methods, PLMs aim to learn dynamic contextual embedding of words in sentences from the unlabeled text. Recent re-

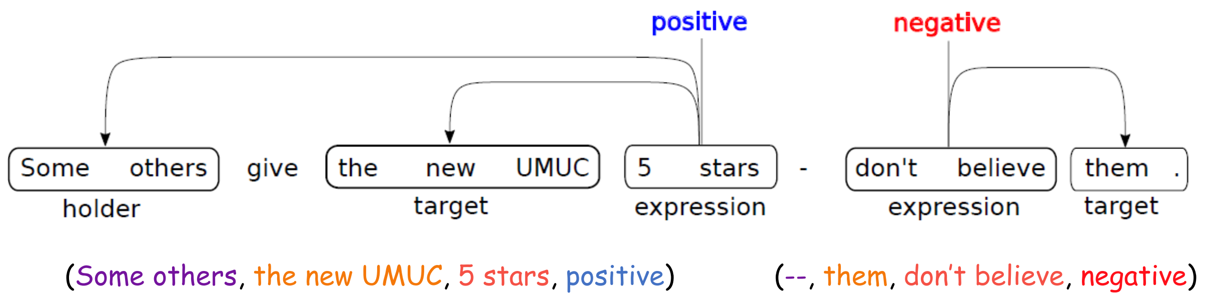


Figure 1: An example of Semeval 2022 Task 10. This figure is modified based on Figure 1 in (Barnes et al., 2021), the original sentiment graph representation is linearized to quadruple representation in this task. "–" indicates that this element in quadruple is empty.

search shows PLMs perform well in various syntactic tasks, such as POS tagging.

BERT (Devlin et al., 2019) is a typical language representation model based on the Transformer encoder architecture. It is pre-trained on two unsupervised tasks: Mask Language Modeling (MLM) and Next Sentence Prediction (NSP). mBERT[1] is a multilingual version of BERT pre-trained on the wiki dumps of 104 languages.

RoBERTa (Liu et al., 2019) removes the NSP task, which has no prominent effect in BERT pre-training and further improves BERT with dynamic masking, deeper network, longer input sequence, and larger training corpora. By virtue of these robust optimizations, RoBERTa significantly outperforms BERT on many tasks. XLM-RoBERTa (Conneau et al., 2020) extends RoBERTa architecture to the multilingual scenario by scalable pre-training on filtered CommonCrawl data containing 100 languages.

SKEP (Tian et al., 2020) incorporates sentiment knowledge into PLMs through sentiment masking and three sentiment pre-training objectives. It provides a unified contextual representation for downstream sentiment tasks.

NB-BERT (Kummervold et al., 2021) is a Norwegian instance of BERT in low-resource language. To alleviate the shortage of pre-training Norwegian corpora, OCR is conditionally used to mine good texts from digital copies.

3 System Overview

To tackle this task, we design a pipeline system that decouples this complex problem into a two-step pipeline with an extraction stage and validation stage. In the extraction stage, we first extract *target-expression-polarity* using an extended grid tagging schema, and then extract *holder* with a question answering system. In the validation stage, we employ a neural validator to determine the extracted results whether are valid in texts. Figure 2 illustrates the overall architecture of our system.

3.1 *Target-expression-polarity* co-extraction

Target-expression-polarity co-extraction aims to extract (t, e, p) triplets from text s (Peng et al., 2020). However, existing works (Peng et al., 2020; Wu et al., 2020a) usually assume that all opinions are expressed explicitly and pay little attention to implicit opinion extraction. In our system, we extend Grid Tagging Scheme (GTS) (Wu et al., 2020a) to adapt both implicit and explicit opinion extraction.

Original tagging space in GTS is an upper triangular grid, whose length and width is the tokenized sequence length l. Specifically, for $i, j \in [0, l]$, cell $(i,\ j)$ contains the tag for token-pair $(t_i,\ t_j)$ in the grid tagging. We integrate two new labels $\{\mathrm{IA}, \mathrm{IO}\}$ into the original tagging scheme and end up with a label set containing eight labels: $\mathcal{Y} = \{\mathrm{A}, \mathrm{O}, \mathrm{IA}, \mathrm{IO}, \mathrm{Pos}, \mathrm{Neu}, \mathrm{Neg}, \mathrm{N}\}$. The grid representation of implicit opinions can thus be implemented by filling IA or IO label in the cell of token-pair $(t_0,\ t_0)$ while not interfering with the representation of explict opinions. We believe this strategy is also reasonable under the perspective of sentence embedding in pretrained encoders, owing to that hidden-state of `[CLS]` (or `<s>` in RoBERTa) token which later fed into the token-level classifier, is often used as the semantic representation of the whole sentence.

We list the meanings of labels in our extended GTS separately in Table 1 and provide a tagging example for the extended GTS in Figure 3.

The decoding algorithm and inference steps we exploit are identical to the original paper.

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

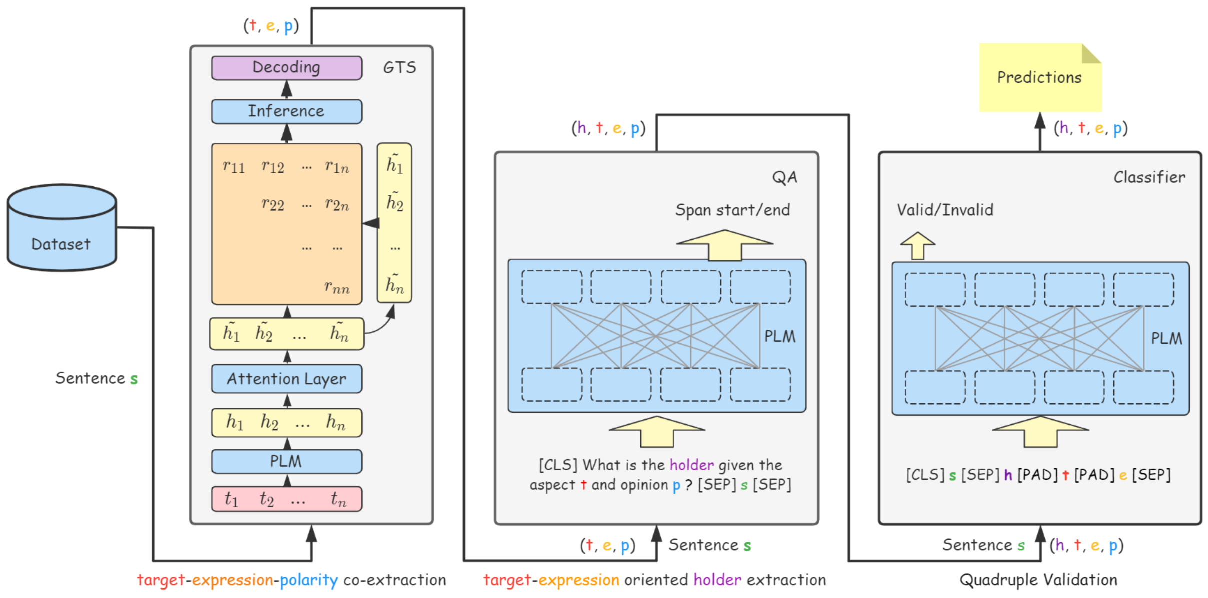
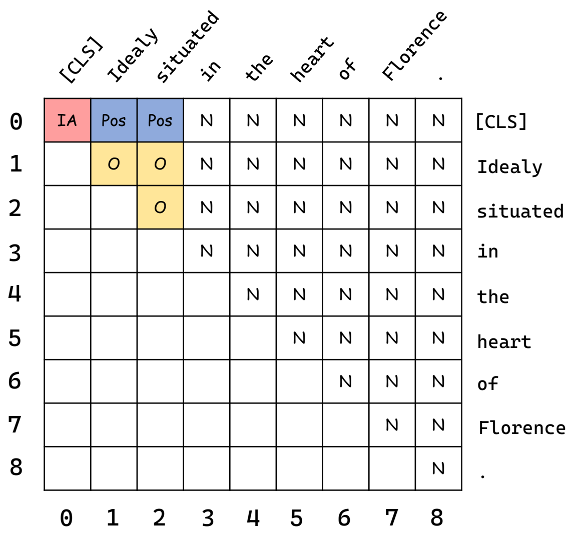

(t, e, p)
Decoding
GTS
Inference
r_{11} r_{12} ... r_{1n}
r_{22} ... r_{2n}
... ...
r_{nn}
$\tilde{h}_1$ $\tilde{h}_2$ ... $\tilde{h}_n$
$\tilde{h}_1$ $\tilde{h}_2$ ... $\tilde{h}_n$
Attention Layer
h_1 h_2 ... h_n
PLM
t_1 t_2 ... t_n
Dataset
Sentence s
(h, t, e, p)
QA
Span start/end
PLM
[CLS] What is the holder given the aspect t and opinion p ? [SEP] s [SEP]
(t, e, p)
Sentence s
Predictions
(h, t, e, p)
Classifier
Valid/Invalid
PLM
[CLS] s [SEP] h [PAD] t [PAD] e [SEP]
Sentence s
(h, t, e, p)
target-expression-polarity co-extraction
target-expression oriented holder extraction
Quadruple Validation

Figure 2: The overview of our system in SemEval-2022 Task 10. Best viewed in color.

Tags	Meanings of tags in cell (i, j)
A O	t_i and t_j belong to the same *target* term. t_i and t_j belong to the same sentiment *expression* term.
IA IO	$i = j = 0$, indicating an implicit *target* term. $i = j = 0$, indicating an implicit *expression* term.
Pos Neu Neg	t_i and t_j respectively belong to an *target* term and an *expression* term, and they form Positive/Neutral/Negative opinion pair relation.
N	No relation between t_i and t_j

Table 1: The meanings of tags in our extended GTS. Cell (i, j) contains the tag for token-pair (t_i, t_j).



	[CLS]	Idealy	situated	in	the	heart	of	Florence	.	
	0	1	2	3	4	5	6	7	8	
0	IA	Pos	Pos	N	N	N	N	N	N	[CLS]
1		O	O	N	N	N	N	N	N	Idealy
2			O	N	N	N	N	N	N	situated
3				N	N	N	N	N	N	in
4					N	N	N	N	N	the
5						N	N	N	N	heart
6							N	N	N	of
7								N	N	Florence
8									N	.

Figure 3: The extended Grid Tagging Scheme on opinion triplet $(-, \text{Idealy situated}, \text{Positive})$ in sentence "Ideally situated in the heart of Florence.". "–" indicates this element in triplet is empty.

Model Ensemble We ensemble the different GTS models using a variety of backbones as the final predictor. Specifically, we perform an unweighted average of predicted distributions $p_{ij} \in \mathbb{R}^d$ from each model on token-pair (t_i, t_j) and get $\bar{p_{ij}}$. The final predicted label index is $\text{argmax}(\bar{p_{ij}})$.

3.2 *Target-expression* oriented *holder* extraction

After obtaining (t, e, p) triplets from the previous step, we further predict *holder* for each given triplet extracted from text s, i.e., *target-expression* oriented holder extraction. We cast this problem as a Question Answering (QA) task, where the context is text s and the answer is the holder span.

Query Construction For holder extraction, we construct the query q for the QA system with the (t, e, p) triplet. Under the multilingual setting of this task, we design different question templates in different languages. The details of the question templates are shown in Table 2.

Encoding and Inference We adopted the same setting as Devlin et al. (2019) to handle the QA task. The input query message q and text s are presented as a single packed sequence:

$$x = \begin{cases} [[\texttt{CLS}]; q; [\texttt{SEP}]; s; [\texttt{SEP}]] & \text{if BERT} \\ [\texttt{<s>}; q; \texttt{</s></s>}; s; \texttt{</s>}] & \text{if RoBERTa} \end{cases} \tag{1}$$

Language	Question Template
English	What is the holder given the aspect t and the opinion e ?
Spanish	¿Cuál es el titular de la opinión dado el aspecto t y la opinión e ?
Basque	Zein da helburu t eta e iritzia emanda iritzia duenak ?
Catalan	Quin és el titular de l'opinió donat l'aspecte t i l'opinió e ?
Norwegian	Hva er meningshaveren gitt aspektet t og meningen e ?

Table 2: The question templates we make to get query message in different languages. t denotes the target term and e denotes the expression term. When t or e is empty, the string "empty" are given to the templates as the term.

Then the context-aware representations of $\boldsymbol{x}$ are fed to a feed-forward linear layer to detect the span-start and span-end position. Note that we treat the special symbol `[CLS]` (or `<s>`) as the impossible answers for implicit opinions that without corresponding *holders*.

In detail, we feed the tokenized input sequence $\boldsymbol{x}$ into the encoder of PLMs. The last hidden-states $\mathbf{H}^x \in \mathbb{R}^{l \times d}$ can be represented by:

$$\mathbf{H}^x = \begin{cases} [\boldsymbol{h}_{[\text{CLS}]}; \boldsymbol{h}_q; \boldsymbol{h}_{[\text{SEP}]}; \boldsymbol{h}_s; \boldsymbol{h}_{[\text{SEP}]}] & \text{if BERT} \\ [\boldsymbol{h}_{\texttt{<s>}}; \boldsymbol{h}_q; \boldsymbol{h}_{\texttt{</s></s>}}; \boldsymbol{h}_s; \boldsymbol{h}_{\texttt{</s>}}] & \text{if RoBERTa} \end{cases} \tag{2}$$

where l is the length of the tokenized sentence, and d is the dimension of PLMs. The final linear span prediction network takes $\mathbf{H}^x$ as the input and outputs two probabilities $p_s, p_e \in \mathbb{R}^l$ for span-start and span-end prediction:

$$p^s, p^e \propto \text{softmax}(\text{Linear}(\mathbf{H}^x)) \tag{3}$$

For model learning, the whole parameters in the QA model are optimized by maximizing the likelihood of span-start and span-end positions:

$$\mathcal{L}_{QA} = -\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(p^s_{y^s_i}\right) + \log\left(p^e_{y^e_i}\right)\right] \tag{4}$$

where N is the number of spans in a single batch, y^s_i and y^e_i are ground-truth span-start and span-end positions respectively.

3.3 Quadruple Validation

To reduce the errors accumulated in previous steps, we design a binary classifier that determines if a combination of *holder*, *target*, and *expression* is valid in text s. The valid triplets predicted by this sub-system are kept along with their corresponding *polarity*, while the others are discarded.

Encoding and Inference We utilize the pre-trained transformers to obtain the representation of text and triplets. Since BERT-like models are more sensitive to sentence-pair input, we concatenate h, t, e with a special symbol `[PAD]` and treat them together as sentence `B`. Concretely, we build the sequence pack in the form of:

$$\boldsymbol{x} = [[\texttt{CLS}]; s; [\texttt{SEP}]; h\,\texttt{[PAD]}\,t\,\texttt{[PAD]}\,e; [\texttt{SEP}]] \tag{5}$$

Under the circumstances of implicit opinion, the empty h, t, e terms are replaced with a special token `[EMP]`.

The validator network takes the representation of $\boldsymbol{x}_{\texttt{[CLS]}}$ as the input and returns the binary validation result. We implement the validator with a linear feed-forward layer.

Span Manipulation Considering that the combinations from sub-spans of the golden *holder*, *target*, and *opinion* terms are also treated as weighted correct predictions, we perform span manipulation to build a more robust classifier. For each ground-truth *holder*, *target*, *expression* term in triplet, we enumerate all the sub-spans and the original term in their corresponding set $\mathbf{H}, \mathbf{T}, \mathbf{E}$, the final triplet candidate pool is the Cartesian product of the three set: $\mathbf{H} \times \mathbf{T} \times \mathbf{E}$.

Finally, for each golden triplet, we randomly select at most k triplets (must include the original one) from the candidate pool as positive samples.

Negative Sampling We further design several rules to mine the negative (i.e., invalid) samples from raw datasets and manipulated golden triplets, including:

1.1. If a golden triplet has a holder, remove the holder and keep other elements.

1.2. If a golden triplet doesn't have a holder, use a holder dictionary to mine pseudo holders from text, packaging the mined holders (if there exist any) with the golden triplet.

2. If a text has multiple golden triplets, exchange the holder / target / expression terms in one with the other.

3. Randomly sample triplets.

Rules $1 \to 3$ are sequentially executed until q samples have been harvested, where q is in positive correlation with the number of positive samples. Meanwhile, we remove all the weighted true and true samples from the mined pseudo negative samples.

4 Experimental setup

4.1 Data Splits

Monolingual Sub-task This sub-task contains 7 different datasets (Agerri et al., 2013; Wiebe et al., 2005; Toprak et al., 2010; Barnes et al., 2018; Øvrelid et al., 2020) across 5 languages. We leveraged the origin splits provided by the organizer and did not include any extra data. The details of data splits are shown in Table 3.

Dataset	Splits		
	Train	Dev	Test
OpeNER$_{en}$	1,744	249	499
OpeNER$_{es}$	1,438	206	410
NoReC$_{Fine}$	8,634	1,531	1,272
MPQA	5,873	2,063	2,112
DS$_{unis}$	2,253	232	318
MultiB$_{ca}$	1,174	167	335
MultiB$_{eu}$	1,063	152	305

Table 3: Data splits.

Cross-lingual Sub-task This sub-task uses a zero-shot setting in which models are trained on the resource that does not contain annotations in the target language. For each target language, we combine all the training sets of OpeNER$_*$, MPQA, and MultiB$_*$ in other languages.

4.2 Implementation and Hyperparameters

This section generally describes the system implementation details and the selection of parameters. The detailed settings can be found in Appendix A.

Monolingual Sub-task For extended GTS and QA part in our pipeline, we tune and select models based on SF$_1$ (Sentiment Graph F1 (Barnes et al., 2021)) scores on the development splits. For the validator part, the models are tuned based on the classification accuracy on the manipulated and sampled development datasets.

In order to maximize the advantages of our system, we test a number of high-performing PLMs and finally RoBERTa$_{large}$, XLM-RoBERTa$_{large}$, NB-BERT$_{large}$, SKEP-ERNIE$_{large}$ and ensemble model [BERT$_{large}$+SKEP-ERNIE$_{large}$] are adopted to the training on different datasets in extended GTS. The max sequence length is set to the max of training and development sets, and meanwhile, the number of hops is chosen in 2 and 3 for GPU memory limitation. The large extended GTS models are trained on a single A100 80G GPU.

For QA sub-system, we use several task-pre-trained PLMs as backbones, such as XLM-RoBERTa$_{large}$-SQuAD[2], distilBERT$_{base}$-SQuAD[3] and RoBERTa$_{large}$-SQuAD[4].

For the validation step, we add LaBSE[5], which is a PLM focusing on language-agnostic sentence embedding and mBERT to the model pools in GTS training.

We fine-tuned all models on the training data using linear learning rate scheduler and the warming up strategy with the learning rate of 3e-5/3e-6 and the batch size of 8~64. We set all random seeds to 1 for reproducibility.

Cross-lingual Sub-task We set all holder positions in tuples to empty instead of leveraging the QA sub-system to extract holders. This is because the QA sub-system requires extra enhancements to fitting the cross-lingual setting (Cui et al., 2019).

The cross-lingual backbone in extended GTS is XLM-RoBERTa$_{large}$, and LaBSE for the validator.

5 Results

In this section, we report the scores on the development and test datasets of two sub-tasks separately. We use *SF*$_1$ (Sentiment Graph F1), *SP* (Sentiment Graph Precision) and *SR* (Sentiment Graph Recall) to evaluate the performance of our system.

5.1 Monolingual Sub-task

Table 4 reports the results of the monolingual sub-task, which ranks 10th in 32 teams. Table 5 shows the ablation analysis of different components on the development set of monolingual tasks. We can see that: 1) Grid-tagging-scheme based target-expression-polarity co-extraction achieves good performance in different languages. 2) The proposed validator can effectively filter out invalid

[2]https://huggingface.co/deepset/xlm-roberta-large-squad2/

[3]https://huggingface.co/distilbert-base-uncased-distilled-squad/

[4]https://huggingface.co/deepset/roberta-large-squad2/

[5]https://tfhub.dev/google/LaBSE/1/

	SF_1	SP	SR
OpeNER$_{\text{en}}$	0.710	0.788	0.646
OpeNER$_{\text{es}}$	0.669	0.735	0.614
NoReC$_{\text{Fine}}$	0.487	0.539	0.444
MPQA	0.269	0.369	0.211
DS$_{\text{unis}}$	0.416	0.480	0.366
MultiB$_{\text{ca}}$	0.658	0.720	0.605
MultiB$_{\text{eu}}$	0.651	0.705	0.605

Table 4: Sub-task 1 Results.

	System	SF_1	SP	SR
OpeNER$_{\text{en}}$	Co-Extraction	0.686	0.710	0.664
	+ Holder Extraction	0.705	0.732	**0.681**
	+ Quadruple Validation	**0.717**	**0.786**	0.660
OpeNER$_{\text{es}}$	Co-Extraction	0.707	0.716	**0.698**
	+ Holder Extraction	0.707	0.716	**0.698**
	+ Quadruple Validation	**0.728**	**0.768**	0.692
NoReC$_{\text{Fine}}$	Co-Extraction	0.501	0.510	**0.492**
	+ Holder Extraction	0.501	0.510	**0.492**
	+ Quadruple Validation	**0.510**	**0.565**	0.465
MPQA	Co-Extraction	0.139	0.148	0.131
	+ Holder Extraction	0.345	0.362	**0.330**
	+ Quadruple Validation	**0.358**	**0.424**	0.309
DS$_{\text{unis}}$	Co-Extraction	0.370	0.453	0.313
	+ Holder Extraction	0.393	0.480	**0.333**
	+ Quadruple Validation	**0.398**	**0.493**	**0.333**
MultiB$_{\text{ca}}$	Co-Extraction	0.674	0.707	0.643
	+ Holder Extraction	0.677	0.711	**0.646**
	+ Quadruple Validation	**0.706**	**0.800**	0.631
MultiB$_{\text{eu}}$	Co-Extraction	0.567	0.553	0.581
	+ Holder Extraction	0.601	0.577	**0.627**
	+ Quadruple Validation	**0.625**	**0.665**	0.589

Table 5: Ablation analysis of our pipeline system on the dev sets in Sub-task 1.

triples and significantly improve the precision of the model.

5.2 Cross-lingual Sub-task

Table 6 shows the results on the cross-lingual sub-task. Compared to the monolingual sub-task, the experimental results shows that the proposed cross-lingual system still performs competitively without training on the target language.

6 Conclusion

In this paper, we propose a pipeline system for $(holder, target, expression, polarity)$ quadruple extraction in ABSA, and adopt a verity of pre-trained language models in distinct parts of system. The evaluation results demonstrate the effectiveness and robustness of our system.

	SF_1	SP	SR
OpeNER$_{\text{es}}$	0.620	0.716	0.548
MultiB$_{\text{ca}}$	0.605	0.596	0.615
MultiB$_{\text{eu}}$	0.569	0.573	0.566

Table 6: Sub-task 2 Results.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Oberländer Laura Ana Maria Kutuzov, Andrey and, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Hongjie Cai, Rui Xia, and Jianfei Yu. 2021. Aspect-category-opinion-sentiment quadruple extraction with implicit aspects and opinions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 340–350, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2019. Cross-lingual machine reading comprehension. In *Proceedings of* *the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1586–1595, Hong Kong, China. Association for Computational Linguistics.

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

Per E Kummervold, Javier De la Rosa, Freddy Wetjen, and Svein Arne Brygfjeld. 2021. Operationalizing a national digital library: The case for a Norwegian transformer model. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 20–29, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4194–4200. International Joint Conferences on Artificial Intelligence Organization.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2886–2892, Copenhagen, Denmark. Association for Computational Linguistics.

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

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654, Seattle, Washington, USA. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Amir Pouran Ben Veyseh, Nasim Nouri, Franck Dernoncourt, Dejing Dou, and Thien Huu Nguyen. 2020. Introducing syntactic structures into target opinion word extraction with deep learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8947–8956, Online. Association for Computational Linguistics.

Hao Tian, Can Gao, Xinyan Xiao, Hao Liu, Bolei He, Hua Wu, Haifeng Wang, and Feng Wu. 2020. SKEP: Sentiment knowledge enhanced pre-training for sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4067–4076, Online. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020a. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2576–2585, Online. Association for Computational Linguistics.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020b. Latent opinions transfer network for target-oriented opinion words extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9298–9305.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and CNN-based sequence labeling for aspect extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 592–598, Melbourne, Australia. Association for Computational Linguistics.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4755–4766, Online. Association for Computational Linguistics.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*

(Volume 1: Long Papers), pages 2416–2429, Online. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, and Duy-Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 612–621, Lisbon, Portugal. Association for Computational Linguistics.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021. Towards generative aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 504–510, Online. Association for Computational Linguistics.

A Experiment Details

Table 7 shows the detailed configurations of each sub-system in the two sub-tasks.

Dataset	Subsystem	Backbone	Hyper-parameters
		Monolingual	
OpenNER$_{en}$	Co-Extraction	[BERT$_{large}$+SKEP-ERNIE$_{large}$]	n-hop=3,lr=3e-5, bs=8, msl=132, epochs=100
	Holder Extraction	RoBERTa$_{large}$-SQuAD	lr=3e-5, bs=16, msl=384, epochs=15
	Quadruple Validation	BERT$_{large}$	bs=16, lr=3e-6, msl=512, epochs=10
OpenNER$_{es}$	Co-Extraction	XLM-RoBERTa$_{large}$	n-hop=3,lr=3e-5, bs=8, msl=193, epochs=100
	Holder Extraction	XLM-RoBERTa$_{large}$-SQuAD	lr=3e-5, bs=32, msl=384, wus=100, epochs=15
	Quadruple Validation	LaBSE	bs=32, lr=3e-5, msl=512, epochs=10
NoReC$_{Fine}$	Co-Extraction	NB-BERT$_{large}$	n-hop=3,lr=3e-5, bs=16, msl=125, epochs=100
	Holder Extraction	XLM-RoBERTa$_{large}$-SQuAD	lr=3e-5, bs=32, msl=384, epochs=15
	Quadruple Validation	NB-BERT$_{base}$	bs=32, lr=3e-6, msl=512, epochs=10
MPQA	Co-Extraction	RoBERTa$_{large}$	n-hop=2,lr=3e-6, bs=16, msl=230, wus=2000, epochs=100
	Holder Extraction	BERT$_{base}$-distilled-SQuAD	lr=3e-5, bs=64, msl=384, epochs=15
	Quadruple Validation	SKEP-ERNIE$_{large}$	bs=64, lr=3e-6, msl=512, epochs=5
DS$_{unis}$	Co-Extraction	SKEP-ERNIE$_{large}$	n-hop=3,lr=3e-5, bs=8, msl=229, wus=500, epochs=100
	Holder Extraction	RoBERTa$_{large}$-SQuAD	lr=3e-5, bs=16, msl=384, wus=1000, epochs=20
	Quadruple Validation	SKEP-ERNIE$_{large}$	bs=64, lr=3e-5, msl=512, epochs=10
MultiB$_{ca}$	Co-Extraction	XLM-RoBERTa$_{large}$	n-hop=3,lr=3e-5, bs=8, msl=265, epochs=100
	Holder Extraction	XLM-RoBERTa$_{large}$-SQuAD	lr=3e-5, bs=32, msl=384, epochs=15
	Quadruple Validation	mBERT$_{base}$	bs=32, lr=3e-5, msl=512, epochs=10
MultiB$_{eu}$	Co-Extraction	XLM-RoBERTa$_{large}$	n-hop=3,lr=3e-5, bs=8, msl=132, epochs=100
	Holder Extraction	XLM-RoBERTa$_{large}$-SQuAD	lr=3e-5, bs=32, msl=384, epochs=15
	Quadruple Validation	LaBSE	bs=32, lr=3e-5, msl=512, epochs=10
		Cross-lingual	
OpenNER$_{es}$	Co-Extraction	XLM-RoBERTa$_{large}$	n-hop=3,lr=3e-6, bs=8, msl=265, epochs=100
	Quadruple Validation	LaBSE	bs=128, lr=3e-6, msl=512, wus=1000, epochs=10
MultiB$_{ca}$	Co-Extraction	XLM-RoBERTa$_{large}$	n-hop=3,lr=3e-6, bs=8, msl=193, epochs=100
	Quadruple Validation	LaBSE	bs=128, lr=3e-6, msl=512, wus=1000, epochs=10
MultiB$_{eu}$	Co-Extraction	XLM-RoBERTa$_{large}$	n-hop=3,lr=3e-6, bs=8, msl=152, epochs=100
	Quadruple Validation	LaBSE	bs=128, lr=3e-6, msl=512, wus=1000, epochs=10

Table 7: Detailed configurations of the subsystems. The abbreviation "bs" stands for batch size, "msl" for max sequence length, "wus" for number of warm-up steps. "[A+B]" represents an ensemble model using backbones A and B.

SenPoi at SemEval-2022 Task 10: Point me to your Opinion, SenPoi

Jan Pfister
University of Würzburg

Sebastian Wankerl
University of Würzburg
DHBW Mosbach

Andreas Hotho
University of Würzburg

{pfister,wankerl,hotho}@informatik.uni-wuerzburg.de

Abstract

Structured Sentiment Analysis is the task of extracting sentiment tuples in a graph structure commonly from review texts. We adapt the Aspect-Based Sentiment Analysis pointer network BARTABSA to model this tuple extraction as a sequence prediction task and extend their output grammar to account for the increased complexity of Structured Sentiment Analysis. To predict structured sentiment tuples in languages other than English we swap BART for a multilingual mT5 and introduce a novel Output Length Regularization to mitigate overfitting to common target sequence lengths, thereby improving the performance of the model by up to 70%. We evaluate our approach on seven datasets in five languages including a zero shot crosslingual setting.

1 Introduction

The goal of sentiment analysis is to understand a writer's opinions expressed in a text. In recent years, this topic became of particular research interest since with the advent of social media platforms, users were encouraged to share their opinions about a wide range of subjects with the world. For example, websites like Yelp collect and share opinions about restaurant visits and various online retailers allow customers to publish their reviews about items in their assortment. Consequently, various text corpora emerged that made it feasible to apply machine learning based approaches to this task (Yadav and Vishwakarma, 2020).

Sentiment analysis can be approached with various degrees of granularity. It is possible to only classify the overall polarity of a sentence as positive or negative, or a more precise intermediate value (Nguyen et al., 2020; Devlin et al., 2019).

Furthermore, the sentiment can be predicted in a more fine-grained manner, like with regard to the various targets addressed in the sentence. This can be of particular interest if the writer is ambiguous in their review and no overall sentiment label can be assigned to a sentence. For example, a customer might describe the service or the interior of a restaurant as unpleasant but the food itself as good. This can be extended to more complex sentence structures, taking into account not only the aspects or the sentiment, but also its holder (i.e. who is expressing the sentiment) or the opinion term itself, e.g. (Yan et al., 2021; Mukherjee et al., 2021).

In this paper, we work on the challenge described by Barnes et al. (2022). Here, the goal is to extract an arbitrary number of opinion tuples from a text. Each tuple can consist of a *holder*, a *target*, a *sentiment expression*, and a *polarity*. However, it is also possible that some of these mentioned *entities* are not present in a certain tuple. Moreover, a sentence may include no sentiment and hence no tuples should be extracted.

The challenge spans data from five different languages, namely English, Spanish, Catalan, Basque, and Norwegian and two different subtasks. First, there is the monolingual setting in which you are allowed to train on all sentiment data, including those in the language that is subsequently used to test the model's capability of extracting sentiment. Moreover, in the crosslingual setting the models are tested on Spanish, Catalan and Basque sentiment data while being trained on any of the other languages. It was also allowed to train on Spanish, Catalan and Basque data as long as it does not contain any annotated sentiment information. This makes the crosslingual structured sentiment analysis task a zero shot setting.

Our approach is based on BARTABSA (Yan et al., 2021), who leverage a *pointer network* to predict sentiment tuples (aspect, opinion, sentiment) by representing them using a custom output grammar. They extend a BART model with a pointer generator network which is thus able to predict token *indices* from the input sequence as output.

Our contributions are as follows: (i) we introduce a new, flexible grammar to model structured sentiment graphs as a pointer sequence, (ii) we explore mT5 as base model allowing us to make cross-lingual predictions, (iii) we introduce a new length regularizer to prevent overfitting to common sequence lengths.

2 Preliminaries

2.1 BART & mT5

BART (Lewis et al., 2020) is a sequence-to-sequence model built using transformer (Vaswani et al., 2017) neural networks. In its default configuration BART consists of 6 encoder and 6 decoder transformer layers. BART is trained as a denoising autoencoder. Hence, the input to its encoder are sentences which are noised using five different methods like masking or permutation of tokens. Consequently, the decoder is trained to restore the original sentence like defined in Equation 1.

mT5 (Xue et al., 2021; Raffel et al., 2020) is another sequence-to-sequence model based on the transformer architecture. It is trained on 101 languages at the same time using the span corruption objective and at time of publication achieved state of the art on many multi-lingual benchmarks. Its training corpus includes all languages used in this challenge (section 1).

2.2 BARTABSA

We model the Structured-Sentiment Analysis task as a seq2seq-task by adopting the framework introduced in BARTABSA. It encodes sentiment tuples (section 1) as a sequence of token indices (pointers) and special tokens.

2.2.1 Output Grammar

In BARTABSA, the target sequence consists of tuples with a fixed size of five:

$$\ldots, a_i^s, a_i^e, o_i^s, o_i^e, s_i^p, \ldots$$

which allows them to express the i^{th} aspect term (a) and its opinion term (o) in an input sentence by their respective start^s and end^e integer *token-index* in the input token sequence, as well as the sentiment polarity class s^p associated with this combination of aspect and opinion terms. Sequences structured like this can be unambiguously converted to aspect-based sentiment tuples as they occur in fixed sequence lengths of five: four token indices and one sentiment class.

2.2.2 Model

Sequences of this *output grammar* can be predicted by the model by *pointing* to the tokens in the input sequence in this fixed order and finally completing the triplet of aspect, opinion and sentiment by predicting the associated sentiment class.

Thus, the task is modelled as an auto-regressive decoding task where the probability of the next output token $P(Y|X)$ depends on both the input X as well as the previously decoded tokens $Y_{<t}$. For a sequence of length m the decoding process is hence given as

$$P(Y|X) = \prod_{t=1}^{m} P(y_t|X, Y_{<t}) \quad (1)$$

Usually, the decoding is stopped as soon as the End-Of-Sequence token `EOS` is predicted.

An architecture to generate such output was introduced as BARTABSA (Yan et al., 2021). BARTABSA is based on BART (Lewis et al., 2020) and uses a pretrained BART encoder to encode the input sentence. Hence, given an input sequence of n tokens $s = [x_1, \ldots, x_n]$, BARTABSA first applies the encoder to obtain the input representation

$$H_e = \text{BARTEncoder}(s) \quad (2)$$

with $H_e \in \mathbf{R}^{n \times d}$ (d denotes the embedding size).

Since the goal of BARTABSA is to predict indices to tokens in the input sequence, its decoder is augmented with a pointing mechanism (Vinyals et al., 2015) to generate these indices (Yan et al., 2021) instead of tokens from a vocabulary (see Equation 5 – 8). To be able to predict l additional sentiment class labels C, they are concatenated to the input and treated as tokens of the input sentence. However, to train and inference the neural network in an auto-regressive manner, i.e. feeding the previous output back into the decoder, the yet generated pointers must be mapped back to their indexed tokens first using the following mapping:

$$\hat{y}_t = \begin{cases} X_{y_t} & \text{if } y_t \in X \\ C_{y_t - n} & \text{if } y_t \text{ is an index of a class label} \end{cases} \quad (3)$$

The BART decoder is then applied to the mapped pointers:

$$h_t^d = \text{BARTDecoder}(H^e; \hat{Y}_{<t}) \quad (4)$$

Using h_t^d, the distribution for pointing to the input sequence is then calculated as:

$$E^e = \text{BARTTokenEmbed}(X) \quad (5)$$

$$C^d = \text{BARTTokenEmbed}(\text{C}) \tag{6}$$

BARTTokenEmbed is a shared embedding layer which is used to embed both the tokens from the input sequence X as well as the class labels C.

$$\bar{H}^e = \alpha\ \text{MLP}(H^e) + (1-\alpha)E^e \tag{7}$$

Finally, the embedded input sequence is concatenated with the embedded class labels. The pointing distribution is then given as the softmax over the concantenated sequence multiplied with the hidden representation h_t^d obtained from the BART decoder:

$$\begin{aligned} P_t &= \text{softmax}([\bar{H}^e; C^d]h_t^d) \\ &= P(y_t|X, Y_{<t}) \end{aligned} \tag{8}$$

where $E^e, H^e, \bar{H}^e \in \mathbb{R}^{n\times d}$, $C^d \in \mathbb{R}^{l\times d}$, $P_t \in \mathbb{R}^{n+l}$. Teacher forcing (Williams and Zipser, 1989) is used during training together with negative log-likelihood as optimization criterion.

3 Methodology

Our approach to predicting sentiment graphs on the given datasets is grounded on BARTABSA (2.2). We adopt and extend their framework to not only be able to predict aspect-based sentiment but also structured sentiment. For this we introduce a new output grammar which is able to model and represent the sentiment tuples as required for this task.

We adapt their sequential pointer representation for *Triplet Extraction* (aspect, opinion, sentiment) to the task at hand.

3.1 Extensions to the Output Grammar

We extend the expressive power of their output grammar in several ways to account for the increased target complexity of the structured sentiment task: Each entity (can be a or o for Triplet Extraction) for structured sentiment analysis can...

- be optional
- consist of arbitrarily many discontiguous parts
- be “source”/“holder” of the sentiment

Defining such an enhanced output grammar which can model these properties enables us to unambiguously represent the sentiment tuples required for the structured sentiment task. In the following we adapt the notion of BARTABSA from “aspect term” to the sentiment *target* and “opinion term” to the polar *expression*.

3.1.1 Entity Absence

First we account for *optional absence* of entities by introducing a special token for each type of entity indicating the begin of its respective entity. This prevents ambiguity if an entity is absent and transforms the previous example from subsubsection 2.2.1 to the following sequence:

$$\ldots, \text{TGT}_{BEG}, t_i^s, t_i^e, \text{EXP}_{BEG}, e_i^s, e_i^e, s_i^p, \ldots$$

where XYZ_{BEG} is the unique class token for each entity type indicating the begin of entity XYZ, t_i and e_i are *integers* referring to token positions in the input sequence and s_i^p is the sentiment class token (see Figure 1 for example). This way if, e.g., there is no target term to be predicted for the i^{th} sentiment tuple the sequence becomes

$$\ldots, \text{EXP}_{BEG}, e_i^s, e_i^e, s_i^p, \ldots$$

and thereby stays unambiguous as it is still well defined that the tokens between e_i^s and e_i^e resemble the expression term and not the missing target term. This can be clearly interpreted although e_i^s and e_i^e are the first two predicted indices of the i^{th} sentiment tuple which were allocated to target entity indices before.

3.1.2 Entity Splitting

Second we allow for *split entities* as described in subsection 4.3 by further extending the output grammar. After every begin-of-entity special token (XYZ_{BEG}) we do not only allow for a single start^s and end^e index tuple but arbitrarily many such tuples. Therefore a two-part target term in the i^{th} triplet is represented like this:

$$\ldots, \text{TGT}_{BEG}, t_i^{s1}, t_i^{e1}, t_i^{s2}, t_i^{e2}, \text{EXP}_{BEG}, e_i^s, \ldots$$

In this syntax the first part of the i^{th} target term can be found between t_i^{s1} and t_i^{e1} and the second part between t_i^{s2} and t_i^{e2} (see Figure 1 for example). Every entity can be modeled by arbitrarily many such pointer tuples and due to the special tokens (XYZ_{BEG}) indicating transitions between the predicted entities the association between pointer indices and entity type stays unambiguous.

3.1.3 Sentiment Source

Last we enable the prediction of the *sentiment holder* or source. Given the previous modifications to the output grammar this can be easily achieved by introducing a new special token HOL_{BEG}.

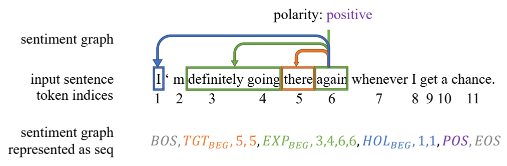

polarity: positive
sentiment graph
input sentence
token indices
I ' m definitely going there again whenever I get a chance.
1 2 3 4 5 6 7 8 9 10 11
sentiment graph
represented as seq
BOS, TGT_BEG, 5, 5, EXP_BEG, 3,4,6,6, HOL_BEG, 1,1, POS, EOS

Figure 1: Exemplary input sentence, its associated sentiment graph (target, expression, holder, sentiment) as well as the sentiment graph modeled using our grammar.

3.1.4 Constructing the Output Grammar

Combining all our extensions to the output grammar introduced above we are able to unambiguously represent structured sentiment tuples in a sentence as a sequence consisting of pointers and special tokens.

First we sort all sentiment tuples per sentence ascending by their token index in the input sentence. For this we only look at the index of the start token of each entity. We first sort the tuples by the start index of the target term, if they are equal by their expression term and lastly on equality we fall back to their holder term. This is done the same way in BARTABSA and PASTE as it is shown that predicting the sentiment tuples of a sentence in a strict order boosts the performance of the final model (Yan et al., 2021; Mukherjee et al., 2021).

A sentiment graph for an input sentence is modelled as follows using our output grammar (example in Figure 1): After the `BOS`-token we encode the sentiment tuples in the order described above. First if the i^{th} sentiment tuple contains an target term we insert an TGT_{BEG}-token followed by a sequence of tuples of start (t_i^s) and end (t_i^e) indices, one for every discontiguous part of the target term. We repeat this for the expression and holder terms of the sentiment tuple. The representation of the i^{th} sentiment tuple is completed by its sentiment polarity token (pos, neu, neg). This process is repeated for every sentiment tuple of the input sentence. After the last sentiment polarity token the output sequence is closed with an `EOS`-token.

If a sentence has no sentiment tuples it is represented by the empty sequence $[BOS, EOS]$.

3.2 Output Length Regularizer

Predicting the structured sentiment graph for an input sentence using our above (3.1.4) defined output grammar is naturally very sensitive to the placement of the `EOS`-token. If our model predicts the `EOS`-token e.g. only a single token too early the last sentiment tuple — consisting of target, expression, holder and sentiment class — becomes incomplete as it lacks at least the sentiment class for the entire tuple. Therefore it can no longer be correctly interpreted or converted to the sentiment graph structure. A missplaced `EOS`-token can lead us to miss an entire sentiment tuple or even more if the `EOS`-token is off by more than the length of a tuple.

During our experiments we noticed some models being prone to overfitting to common positions of the `EOS`-token (also see Newman et al. (2020)) in the train set (5.1). When analysing the predictions of such a model during training we found that often the `EOS`-token was predicted not only at the correct location in the sequence, but also once more, earlier at the most-common `EOS`-token position in the train set. Usually this is right after the `BOS`-token. As we convert the output sequence to the sentiment graph up to the first `EOS`-token (3.1.4) this results in an empty graph. Such an incorrect sequence consisting of an otherwise correctly predicted sentiment unit with a target and an expression exemplarily looks like this:

$$[BOS, EOS], t_i^s, t_i^o, \mathrm{EXP}_{BEG}, e_i^s, e_i^e, s_i^p, EOS$$

In this case the TGT_{BEG}-token got erroneously replaced by an overfitted `EOS`-token, thereby stopping the output sequence early — as indicated by the gray font. As can be clearly deducted from Figure 1 such a token replacement would result in an empty sequence and thereby no extracted sentiment graph. We cannot just ignore/fix such a misplaced `EOS`-token automatically because we cannot decide whether such an error occurred at all or which begin-of-entity (XYZ_{BEG}) got replaced by the first `EOS`.

Therefore we have to prevent these errors from occurring in the first place. To encourage the model to predict the `EOS`-token at the correct location only, we extended the loss function to make it more sensitive to the placement of the `EOS`-token.

We realize this by introducing an additional loss component $\mathcal{L}_{\mathcal{RE}}$. The predictions $\hat{y}$ of our model during training have the shape $\mathbb{R}^{s \times \hat{t}}$, where $\hat{t}$ and s is the length of the prediction $\mathcal{P}$ and source sequence $\mathcal{S}$ respectively, as we predict t *target* indices pointing to tokens in the source sequence of length s (including special tokens).

Now, along every column $j \in 1 \ldots \hat{t}$ of $\hat{y}$ we calculate the softmax to decide the token probability distribution at prediction step j. From this matrix we now extract the row $o = \hat{y}_i \in \mathbb{R}^{\hat{t}}$, $i \in 1 \ldots s$ corresponding to the EOS-token. Then, $\mathrm{softmax}(o)$ at position j represents the probability that the j-th token in the target sequence is an EOS-token over all positions in the target sequence.

We calculate the cross entropy between this resulting vector o and the correct position for the EOS-token in the target sequence.This loss addition penalizes high-probabilities for EOS-tokens in the wrong locations:

$$\mathcal{L}_{\mathcal{RE}}(y, \hat{y}) = \mathrm{CE}(t, \mathrm{softmax}(\hat{y}_i)) \qquad (9)$$

where t is the length of the target sequence y — thereby the correct position of the EOS-token — and i is the vocabulary index of EOS.

We add our Output Length Regularization to the Cross Entropy loss:

$$\mathcal{L}(y, \hat{y}) = \mathrm{CE}(y, \hat{y}) + \mathcal{L}_{\mathcal{RE}}(y, \hat{y}) \qquad (10)$$

4 Experiments

We base our model on two different but related architectures depending on the subtask. For both tasks, we build a model similar to BARTABSA (Yan et al., 2021)[1] consisting of a transformer encoder-decoder model augmented with a pointer layer for predicting indices in the input sequence.

We always train for 75 epochs and select the best model based on the best performance on the validation set. This strategy is in line with BARTABSA but we increased the maximum number of epochs from 50 to 75 as we found in preliminary experiments that mT5 sometimes was able to improve slightly on the validation set beyond 50 epochs. If we train on multiple datasets at the same time we also validate on a concatenation of their respective validation sets. We finetune the models once with and once without the Output Length Regularization (3.2). We select model dependent learning rates, learning rate schedules and optimizers as suggested in their respective original papers or BARTABSA. Therefore we finetune BART models using Adam with a peak learningrate of $5e^{-5}$ in a triangular learning rate schedule and mT5 using AdaFactor with a constant learningrate schedule and a learningrate of 0.001.

The results are evaluated using the Sentiment Graph F_1 introduced by Barnes et al. (2021). For this metric they calculate the true positives by averaging the overlap for exact token-level matches between predicted and gold spans over all three entities. To obtain precision and recall they now divide the number of correctly predicted tokens by the total number of predicted tokens and gold tokens respectively.

4.1 Subtask 1: Monolingual

For the monolingual task we divide between English and non-English datasets. For the English datasets we select a pretrained monolingual English BART model as our transformer architecture as suggested in BARTABSA. As there are no BART models publicly available for all non-English languages from our datasets, we choose a pretrained mT5 model for those instead which was pretrained on 101 languages including all languages present in our datasets. We do not train a separate mT5 for every single dataset or language since finetuning mT5 on small datasets lead to significant instabilities during training (5.1) which we counteract by concatenating all datasets available to us.

We compare our approach against the baselines provided by the task organizers[2]. They provide a sequence labelling approach which consists of three separate BiLSTM models, one each for extracting the holders, targets, and expressions followed by another BiLSTM-based relation prediction model. On top of the concatenation of these three outputs a classification task is trained to predict whether two predicted elements are related or not. The second baseline is a graph parsing model described by Barnes et al. (2021). It works by modelling the sentiment tuples as dependency graphs and then predicting those using the neural dependency parser introduced by Dozat and Manning (2018).

4.2 Subtask 2: Crosslingual

In the crosslingual setting it is important for our model to be able to generalize and transfer well between languages. Therefore we once again select mT5 as our basemodel and train it on English and Norwegian at the same time as these are the only languages available for training in this setting. We train on both available languages at the same time since we expect this to improve generalization between languages.

[1]For our experiments we reuse their codebase making use of torch, transformers and fastNLP.

[2]https://github.com/jerbarnes/semeval22_structured_sentiment

4.3 Datasets

The datasets provided for this challenge can be split into five languages: English (darmstadt (Toprak et al., 2010), mpqa (Wiebe et al., 2005), opener (Agerri et al., 2013)), Spanish (opener), Catalan (multibooked (Barnes et al., 2018)), Basque (multibooked) and Norwegian (norec (Øvrelid et al., 2020)). All datasets are structured identically consisting of a source sentence and its annotated sentiment graphs, which are represented by their sentiment tuples (target, expression, holder, polarity) as introduced in section 1. Additional complexity is added by the fact that every sentiment tuple entity can be either completely missing or even consist of arbitrarily many discontiguous parts as described in subsubsection 3.1.4: e.g. in "[. . .] have degraded the image of the University severely" the opinion term consists of two parts: "degraded" and "severly".

4.4 Dataset Sampling

During preliminary experiments we observed stability problems while finetuning mT5 (5.1) which we were only able to mitigate by training on the concatenation of different datasets and sampling the training dataset. We were able to address this instability by undersampling the training set in such a way that *at most* one fifth of the training set consists of empty samples. We implemented this sampling by first ensuring this ratio for all datasets separately and concatenating them afterwards. We kept this sampling strategy for all experiments using mT5 except when training only on the concatenation of the English datasets where we found that a maximum of one fourth empty samples achieved better results on the validation set.

We ran the same experiments with BART and found it to be not nearly as susceptible to a sampling strategy compared to mT5. Nevertheless we found that BART is able to gain small improvements on the validation set only when training on the concatenation of all English datasets without Output Length Regulatization while sampling the training datasets such that at most half of the samples are empty. So we sampled the training dataset only in this specific case when finetuning BART.

5 Results & Evaluation

5.1 Training Instability of mT5

While finetuning mT5 on our datasets during preliminary runs we noticed that the model is not only generally very unstable during training but also that the resulting performance is very sensitive to different dataset splits. When training on individual training datasets including all samples the model was not able to achieve sentiment-F_1-scores above 5% on the validation set. For this it did not matter whether we included the Output Length Regularization (3.2) or not. We only managed to achieve competitive results using mT5 after we significantly undersampled the empty sentiment tuples in the dataset as described in subsection 4.4. We believe this to be already first signs of EOS-token location overfitting (3.2) as all empty samples are represented by the sequence $[BOS, EOS]$.

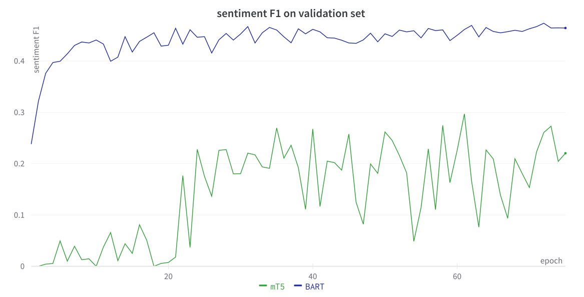


Figure 2: Comparison of two representative English validation set sentiment-F_1 curves for BART and mT5 during training. The models got evaluated in each of the 75 training epochs.

Training instabilities of mT5 become evident when looking at the F_1-scores on the validation set during training as representatively plotted in Figure 2 (green). For comparison we rerun the same setup using BART (blue). While the training loss of the mT5-models descends during training without large jumps or spikes the F_1-score on the validation set oscillates heavily. Although all mT5-models we trained contained such frequent and huge jumps in the F_1-score, there became no common pattern apparent between different runs. Even the slightest change e.g. different dataset sampling can result in a completely different F_1-score curve on the validation set. At the same time the loss during training on the same dataset remained smooth. This is not surprising given the fact that a wrongly placed EOS-token can invalidate the entire prediction as described in subsection 3.2, while in comparison according to the cross entropy loss a higher probability for the EOS-token at the e.g. second position only slightly decreases the measured performance. In other words the final evaluation metric is way more sensitive to the placement of the EOS-token

Table 1: Submitted results for the monolingual subtask as described in subsection 4.1.

		ours		baselines	
dataset	model	sent-F_1	place	seq. label	graph parser
opener_en	BART	0.651	12	0.33	0.52
mpqa	BART	0.338	11	0.02	0.12
ds_uni	BART	0.417	6	0.06	0.20
average		0.469	—	0.14	0.28
opener_es	mT5	0.504	17	0.24	0.50
norec	mT5	0.280	18	0.20	0.36
multib_ca	mT5	0.517	16	0.34	0.52
multib_eu	mT5	0.439	18	0.37	0.55
average		0.435	—	0.29	0.64
overall avg		0.449	16	0.22	0.40

than the training objective. This is the core issue we are addressing by introducing the additional loss component (3.2). We did not observe a similar phenomenon during training of any BART model.

5.2 Subtask 1: Monolingual

As we approached the monolingual setting using two different base-models we also analyse them separately. Overall for this monolingual subtask we placed 16^{th} out of 31 teams on the leaderboard at the time the challenge ended.

For mT5 we report the results for finetuning on all datasets on all languages at the same time using our Output Length Regularization, as training mT5 became slightly more stable on a larger dataset.

For the English results we finetune BART with our Output Length Regularization only on the respective dataset the model was tested on except when testing on the darmstadt dataset. For darmstadt we trained BART on a combination of all English datasets and without Output Length Regularization. If we train only on the darmstadt dataset itself and with Output Length Regularization like for the other datasets, we achieve an F_1-score of only 0.389. We chose this strategy for all BART runs in general and the darmstadt dataset in specific as it resulted in the highest scores on the respective validation datasets. The results for the other datasets when training BART on all english datasets at the same time is included in Table 2.

In Table 1 we report our performance on the monolingual task and compare it against the baselines provided by the task organizers. On English datasets using BART we achieved in general a considerably higher placement (6,11,12) than our over all placement (16) and are able to comfortably beat the employed baselines (subsection 4.1). This indicates that our method works comparatively better using BART as base model than mT5 (16, 17, 18, 18) where our approach consistently beats the sequence labelling approach but matches the performance of the graph parser only on some datasets.

Table 2: Comparison of a single BART and a single mT5 trained on all English datasets at the same time.

	sent-F_1	
dataset	BART	mT5
opener_en	0.493	0.471
mpqa	0.326	0.159
darmstadt_uni	0.365	0.218
average	0.395	0.283

5.2.1 BART vs. mT5 Performance

To further evaluate this presumed performance discrepancy between BART and mT5 we compare our approach using both base models on the English datasets as this is the only language both models have in common. We train both models using the Output Length Regularization and only differing by their respective optimal sampling strategy as laid out in subsection 4.4. For a fair comparison we train both models on all datasets at the same time as a larger dataset reduces training instabilities of mT5 (5.1). The results are visible in Table 2.

We find the performance to be drastically dropping overall when we switch from a BART to an mT5 model. The overall validation loss for both models in every epoch is reported in Figure 2. We assume this drop is hugely driven by the training instabilities (5.1) we observed, although we also suspect the differences in pretraining (2.1) to lead to this discrepancy. This explains our comparatively better placement in Table 1 when we are able to use BART instead of mT5 to solve the task.

5.2.2 Ablation: Output Length Regularization

In order to evaluate how well our novel Output Length Regularization (3.2) is able to improve model performance we finetune a separate BART for every English dataset once with and once with-

Table 3: Comparison of performance for models trained with and without length regularization. BART models were trained on the dataset they are tested on while mT5 models were trained on all languages at the same time.

		sent-F_1	
model	dataset	w/ lenreg	w/o lenreg
BART	opener_en	0.651	0.635
	mpqa	0.338	0.320
	darmstadt_uni	0.389	0.320
	average	0.459	0.425
mT5	opener_es	0.504	0.410
	norec	0.280	0.251
	multibooked_ca	0.517	0.374
	multibooked_eu	0.439	0.353
	average	0.435	0.347
	overall average	0.449	0.367

Table 4: Average number of predicted sentiment tuples per sentence compared to actual average number of sentiment tuples per sentence in the testset.

		lenreg		
	dataset	w/	w/o	testset
BART	opener_en	1.77	1.88	1.73
	mpqa	0.21	0.29	0.24
	darmstadt_uni	0.31	0.47	0.41
mT5	opener_es	2.02	1.49	2.33
	norec	1.28	0.77	0.97
	multibooked_ca	1.39	1.00	1.56
	multibooked_eu	1.19	0.76	1.43

out length regularization. We repeat this setup for mT5 but train a single common mT5 on all datasets together as we found mT5 to be more stable during training with increasing dataset sizes.

We compare the results in Table 3. It is apparent that for all runs for both base models on all datasets and languages the sentiment F_1 score improves when adding our length regularization. Therefore we conclude that our Output Length Regularization (3.2) does indeed help the model learn where to place the EOS-token and thereby decide how many sentiment tuples are present in the sentence. This results in better predictions for this task especially for the mT5 model.

Originally we introduced the Output Length Regularization to fix an overfitted EOS-token at the second position for the mT5 model as exemplarily indicated by the colors in subsection 3.2. This led the mT5 model to predict too few (commonly zero) sentiment tuples. When analysing the differences in length of the output sequences (see Table 4) we found that for the mT5 model it consistently increased the number of sentiment tuples predicted by the model and thereby almost always moves the average number of predicted tuples per sentence closer to the average number of sentiment tuples present in the dataset splits. Only for norec the average number of predicted tuples on the testset overshoots the average number of sentiment tuples on the testset.

Table 5: Performance of BART per dataset (columns) when finetuning only on a single dataset (rows). All models trained with Output Length Regularization.

	op_en	mpqa	ds_uni
opener_en	0.651	0.007	0.195
mpqa	0.008	0.338	0.050
darmstadt_uni	0.300	0.005	0.389
all English	0.493	0.326	0.365

5.2.3 Crossdomain Performance

The crosslingual subtask of this challenge primarily evaluates how well a model is able to generalize between different languages and different domains at the same time. We also analyze the performance of our BART model when we change only the target domain by crossdomain zero shot evaluating on a different english dataset. Therefore we finetune a separate BART model with Output Length Regularization on each of the three English datasets separately and then use each model to predict all other English datasets.

In Table 5 we find BART to be able to generalize between darmstadt_uni and opener_en albeit this comes at the cost of a significant performance loss in both directions. Meanwhile mpqa does not seem to be similar enough to either of the other two English datasets for the model to output meaningful crossdomain predictions. This can be explained as both darmstadt_uni and opener_en datasets consist of reviews of universities and hotels respectively, while mpqa is a dataset focused around political opinion expression. It is likely that the model benefits from the more similar phrasing used in both review datasets compared to political opinions.

Table 6: Submitted results for the crosslingual subtask as described in subsection 4.2. We finetuned mT5 with Output Length Regularization on all datasets at once.

dataset	sent-F_1	place
opener_es	0.315	14
multibooked_ca	0.259	15
multibooked_eu	0.243	14
average	0.272	15

Table 7: Comparison of an mT5 trained with and without Output Length Regularization (3.2).

	sent-F_1	
dataset	w/ lenreg	w/o lenreg
opener_es	0.315	0.245
multibooked_ca	0.259	0.105
multibooked_eu	0.243	0.131
average	0.272	0.160

5.3 Subtask 2: Crosslingual

For the crosslingual task (Table 6) we predict structured sentiment tuples in Spanish, Catalan and Basque without prior training on any sentiment annotations in any of these languages. We again finetune an mT5, but for this subtask on all languages at once which are not in the set of target languages: English & Norwegian. Training on a larger dataset consisting of multiple languages not only stabilizes training but also possibly helps the model to generalize between languages to predict structured sentiment in unseen languages. Compared to the monolingual subtask where mT5 was trained using sentiment annotations in the target languages (Table 1), here mT5 loses on average over 40% of performance in this zero shot crosslingual setting. Therefore we are able to show that despite a significant performance loss pointer prediction models are able to zero shot generalize between languages and domains at the same time.

5.3.1 Ablation: Output Length Regularization

Again we evaluate our Output Length Regularization by finetuning two mT5 on all English and Norwegian datasets at the same time, once with and once without Output Length Regularization. We find the same results as already described in 5.2.2: Output Length Regularization is able to improve the zero shot crosslingual generalization significantly as can be found in Table 7. Finetuning mT5 for pointer prediction using this length regularization increases the performance by 70% averaged over all target datasets.

6 Related Works

The PASTE framework (Mukherjee et al., 2021) also uses pointer networks to solve the task of aspect-based sentiment analysis. Instead of Transformers, they use an LSTM (Hochreiter and Schmidhuber, 1997) as decoder and two additional Bi-LSTMs as pointer networks — one for pointing to aspect and opinion term each.

Peng et al. (2020) propose an approach to extract opinion triplets from text in a generative manner without using pointers. However, their output grammar is also less flexible meaning that all of the entities (target, aspect, and sentiment) have to be present in each triple. They also introduced one of the data sets used to train BARTABSA and PASTE.

Apart from structured sentiment analysis, pointer networks were also successfully applied to various further NLP tasks where it is beneficial to directly transfer parts of the input sequence to the output. This includes automatic summarization (See et al., 2017; Enarvi et al., 2020) and entity extraction (Nayak and Ng, 2020).

7 Conclusion

In this work, we adapted BARTABSA, a pointer network based on BART, for the task of structured sentiment analysis. In particular, we introduced a new output grammar which is able to model the increased complexity of this task by taking into account new entity types, split entities and missing entities in sentiment tuples.

We also experimented replacing BART with an mT5 network to allow for input sequences in languages other than English. We found that using the approach of BARTABSA it is possible to swap out BART for another base model but in the case of mT5 this comes with a significant performance hit which we suspect is mainly driven by training instabilities we encountered. Moreover, we introduced a output length regularizer to reduce overfitting to common sequence output lengths from the trainset. We found this to be very beneficial consistently on all data sets and to increase relative performance by up to 70%.

Acknowledgements

This research project is funded by the Bavarian Research Institute for Digital Transformation (bidt), an institute of the Bavarian Academy of Sciences and Humanities. The author is responsible for the content of this publication.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Andrey Kutuzov, Laura Ana Maria Oberländer, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Seppo Enarvi, Marilisa Amoia, Miguel Del-Agua Teba, Brian Delaney, Frank Diehl, Stefan Hahn, Kristina Harris, Liam McGrath, Yue Pan, Joel Pinto, Luca Rubini, Miguel Ruiz, Gagandeep Singh, Fabian Stemmer, Weiyi Sun, Paul Vozila, Thomas Lin, and Ranjani Ramamurthy. 2020. Generating medical reports from patient-doctor conversations using sequence-to-sequence models. In *Proceedings of the First Workshop on Natural Language Processing for Medical Conversations*, pages 22–30, Online. Association for Computational Linguistics.

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

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

Rajdeep Mukherjee, Tapas Nayak, Yash Butala, Sourangshu Bhattacharya, and Pawan Goyal. 2021. PASTE: A tagging-free decoding framework using pointer networks for aspect sentiment triplet extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9279–9291, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8528–8535.

Benjamin Newman, John Hewitt, Percy Liang, and Christopher D. Manning. 2020. The EOS decision and length extrapolation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 276–291, Online. Association for Computational Linguistics.

Xuan-Phi Nguyen, Shafiq Joty, Steven Hoi, and Richard Socher. 2020. Tree-structured attention with hierarchical accumulation. In *International Conference on Learning Representations*.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8600–8607.

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

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

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

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

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.*, 1(2):270–280.

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

Ashima Yadav and Dinesh Kumar Vishwakarma. 2020. Sentiment analysis using deep learning architectures: a review. *Artificial Intelligence Review*, 53(6):4335–4385.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2416–2429, Online. Association for Computational Linguistics.

SSN_MLRG1 at SemEval-2022 Task 10: Structured Sentiment Analysis using 2-layer BiLSTM

Karun Anantharaman, Divyasri K, Jayannthan PT
Ms. S.Angel Deborah , Ms. S. Rajalakshmi,
Dr. R.S. Milton, Dr. T. T. Mirnalinee
Department of Computer Science and Engineering
SSN College of Engineering
Chennai 603 110, Tamil Nadu, India
karun19049@cse.ssn.edu.in,
{divyasri2011037,jayannthan2010606}@ssn.edu.in,
{rajalakshmis, angeldeborahs}@ssn.edu.in,
{miltonrs, mirnalineett}@ssn.edu.in

Abstract

Task 10 in SemEval 2022 is a composite task which entails analysis of opinion tuples, and recognition and demarcation of their nature. In this paper, we will elaborate on how such a methodology is implemented, how it is undertaken for a Structured Sentiment Analysis, and the results obtained thereof. To achieve this objective, we have adopted a bi-layered BiLSTM approach. In our research, a variation on the norm has been effected towards enhancement of accuracy, by basing the categorization meted out to an individual member as a by-product of its adjacent members, using specialized algorithms to ensure the veracity of the output, which has been modelled to be the holistically most accurate label for the entire sequence.Such a strategy is superior in terms of its parsing accuracy and requires less time. This manner of action has yielded an SF1 of 0.33 in the highest-performing configuration.

1 Introduction

Sentiment Analysis is a specialized area under Natural Language Processing which deals with the extraction of opinions and emotions from text which may include reviews, social media posts, forums or news. Sentiment Analysis has become a powerful tool for detecting the public opinion on any topic.
This research article evinces how the sentiments and opinions expressed by people in various environments could be understood by means of a structured sentiment analysis (SSA) approach. There are conventional means to this end, in transition-based and graph-based techniques, but a potent alternative is the novel way to envision dependency parsing tasks as algorithmic pattern-recognition by way of sequence labelling. Sequence labeling is essentially an agglomeration of multiple discrete categorizations, extended on to each element of the entire sequence. The model is trained on 7 seven data sets with 5 different languages English, Spanish, Basque, Norwegian and Catalan.

In Subtask-A we try to separately identify the holder and target in the text, along with the sentiment expression used using a Sequence Labelling Approach.

In Subtask-B we predict whether the extracted tokens have a relation or the lack of one.

Subtask-C involves the concatenation of the results of Subtask-A and Subtask-B to provide the final predictions.

2 Background

2.1 Definitions

The following contains descriptions of the models made use of, and related terminology. Namely we will define LSTM, BiLSTM, Sigmoid, Linear, Max Pooling Layers and Word Embeddings and the BIO Sequence Labelling approach.

Long Short-Term Memory(LSTM)

Speech-recognition language models, among numerous other activities, have seen encouraging levels of success by availing Recurrent Neural Networks (RNN) (Mikolov et al., 2010, 2011) (Graves and Schmidhuber, 2005). The model is made to predict the current output by analysing the long-distance features, by virtue of how history information is incorporated into an RNN. Long-range dependencies are better detected and processed by LSTMs than RNNs (Graves and Schmidhuber, 2005) by virtue of their difference in terms of how the latter has purpose-built memory cells in lieu of the hidden layer updates in the former (Hochreiter and Schmidhuber, 1997). Aside from this striking feature, LSTMs and RNNs are essentially the same.

BiDirectional Long Short-Term Memory(BiLSTM) - It a sequence processing model that comprises of one LSTM receiving input in the forward direction, and another which drives input backwards. BiLSTMs are used to enhance the subject and context to be used by the network, by proliferating the quantity of data that could be accessed by the algorithm (e.g. cognizance of the words in front of, and following a certain word that is to be analyzed).

Sigmoid Layer The sigmoid layer is responsible for making sure that the output remains confined within the interval (0,1), through subjecting a sigmoid function on the input.

$$S(x) = \frac{1}{1 + e^{-x}}$$

Linear Layer Mathematically, this module is designed to calculate the linear equation $Ax = b$ where x is input, b is output, A is weight.

Max Pooling Layer Max pooling task results in a map output in which the significant features from the preceding feature map is reflected, considering how it extracts the most crucial constituents of the feature map under the purview of the filter.

Word Embeddings It refers to a specialized means of representation for input text, dependent on its connotation, thus attributing the same representation to words who depict a similar meaning. In this paper we use GloVe and FastText embeddings.

BIO Sequence Labelling - "B-I-O" is a tagging scheme, where either of "B" (Beginning), "I" (Inside), or "O" (Outside) labels denote the relative position of the part of speech. If none of these three assignments could be meted out to the text in question, a special label denoting that case could be assigned.

2.2 Related Work

Sentiment analysis constitutes five operations in sequence as i) sentiment expression extraction, ii) sentiment target extraction, iii) sentiment-holder extraction, iv) definition of relationships between elements, v) assignment of polarity. (Yadav and Vishwakarma, 2020) compares various SOTA DL techniques which have been applied to this problem, including CNNs, Recursive Neural Nets, RNNs, LSTM, GRU and Deep Belief Networks, concluding that LSTMs give better results compared to other models. (Xu et al., 2019) explored the possibility of using BiLSTMs for sentiment analysis on comments, and found improved accuracy. (Phan et al., 2020) proposed an ensemble model of various feature vectors to form embeddings which were fed to a CNN, this method greatly improved accuracy on sentences with fuzzy sentiment. A Bidirectional RNN-CNN (Basiri et al., 2021) was also found to achieve SOTA results. Thus, emphasizing the fact that bidirectional models capture context better in textual data.

Sequence Labeling Typically, it would be advantageous to formulate an NLP operation akin to a general sequence-labeling task. Each element from a defined input sequence is considered, and a collection of labels is scrutinized to pick out one relevant to the text, and an assignment is made with the aforementioned. In this paper we propose a Part of Speech tagging approach. (Akhundov et al., 2018) have shown how BiLSTM-CRF can be used for this task. (Prasad and Kan, 2017) shows the extraction of keyphrases and relation prediction using CRFs for sequence labeling.

2.3 Data

The seven datasets in question encompass five different languages of which the expressions are composed of, to be subjected to structured sentiment analysis. They are made such as to typify the way in which the particular approach to this study performs, as necessitated by the task. The datasets are similar in terms of possessing holders, expressions and targets, while dissimilarities are by virtue of their frequency and distribution.

The maximum number of holders in an individual dataset would be the 2,054, featured in MPQA (Wiebe et al., 2005), while the lion's share of targets (8,293) and expressions (11,115) both are allocated to NoReCFine (Øvrelid et al., 2020). The set DSUnis (Toprak et al., 2010) possesses the least amount of all among holders, targets, and expressions in 94, 1601, and 1082 respectively.

The dataset provided by opeNER project (Agerri et al., 2013) are opener_en and opener_es . The number of targets and expressions in opener_en, opener_es are 1286, 1760 , 1062 and 1625 respectively.

All the groups carry markers for polarity of the text, which shows the positive or negative connotation carried by each member. MPQA and DSUnis are distinctive in how they also include instances of "neutrality", beside the extremities. In DSUnis, this feature is utilized while dealing with clauses that showcase varying degrees of both polarities, with contextual variance. The neutralities in MPQA are much less complicated, as they only vouch for words that are subjective, and may not necessarily have a polarizing effect.

MPQA is entirely in English, and carries text from news agencies. The two datasets involving critiques of hotels are MultiBEU and MultiBCA (Barnes et al., 2018), which are Basque and Catalan, respectively. They include markers that further qualify each polarity, as "strong positive" or "strong negative". DSUnis is essentially an agglomeration of user reviews in English from the internet towards e-commerce and educational institutions, with only the latter being considered as part of this research. This is a consequence of the e-commerce reviews mostly comprising only the relevant polar targets that account for polarity, without holding the expressions themselves. NoReCFine is a Norwegian dataset made up of professional reviews belonging to a multitude of domains, and is also the most voluminous set of the lot. It additionally shows the intensity of the polarity for each expression, as in slight, normal or strong, which is deemed beyond the scope of the study.

3 System Overview

Here we provide a model that first learns to extract the sub-elements (holders, targets, expressions) using sequence labelers, and then tries to classify whether or not they have a relationship.

Specifically, we first train three separate BiLSTM models to extract holders, targets, and expressions, respectively. We then train a relation prediction model, which uses a BiLSTM + max pooling to create contextualized representations of 1) the full text, 2) the first element (either a holder or target) and 3) the sentiment expression.

These three depictions are concatenated and sent to a linear layer, followed by a sigmoid function. The training consists of predicting whether two elements have a relationship or not, converting the problem in binary classification.

4 Experimental Setup

The sequence labelling model employed in our research essentially attempts to divaricate and demarcate various elements of the input text into tuples. To begin with, the starting file Get_baseline.sh is executed in order to call the subsequent files in convert-to-bio.py, and

Convert-to-rels.py. The former is responsible for converting the given statements into the workable format, as in the stratification of data into holder, target and expression parts, with "B-I-O" labels. The labels are also carriers of the polarity (positive, negative, or neutral) of each text. Next up, the Convert-to-rels.py file is called, upon which, it extracts the target/expression pairs and holder/expression pairs, and creates 2 new fields e1, e2. Furthermore, the BiLSTMs are trained to extract holder, target and expression from the data. Pretrained GloVe or FastText embeddings are also provided to the model for the operation of labeling. After passing the input through the aforementioned processing stages, the vocabulary obtained in the end is stored in a vocabulary dictionary. 3 BiLSTMs are trained separately for each labeling task, following which, is the relation prediction model trained. The full sentence, in addition to e1 and e2, are all scanned for relations. If relations are present, then predictions are made accordingly. Finally to get a consolidated prediction.json terminal output file, the inference.py file is called. The holders, targets, and expressions have been extracted using the trained BiLSTMs already and the polarity is available from the expression labelling. The data is formatted accordingly and lastly packed into a neat json format. The following model incorporates a learning rate of 0.01. We have run it for upto 10 epochs, to arrive at a considerable amount of accuracy. The number of hidden layers in this model is kept as 1 by default.

5 Results

The efficacy of the sentiment analysis model towards encapsulating the full sentiment graph can be depicted in terms of the criteria as enumerated by two benchmarks, in Sentiment Graph F1 (SF1) and Non-polar Sentiment Graph F1 (NSF1). The sentiment graphs are considered in terms of holder, target, expression, and polarity, for evaluation by SF1, whereas NSF1 takes into account all elements of the tuple, except polarity for scrutinization. In this case, a perfect match on the graph, with respect to the mean of all three spans, and including weights pertaining to the gold and expected spans for each member is considered as a true positive. The precision value is a ratio where the numerator is the sum of predicted tokens that are found to be right, with the denominator being the total sum of all predicted tokens (amount of gold tokens is the denominator in case of recall). Empty targets and tokens are also taken into consideration.

Dataset	SF1-Score
norec	0.191
multibooked_ca	0.323
multibooked_eu	0.331
opener_en	0.306
opener_es	0.257
mpqa	0.015
darmstadt_unis	0.104
average	0.218

Table 1: Results

6 Conclusion

In summation, our research work presents the three-tier model which we constructed, and is found to have the best accuracy of 0.33. The model has been proven to consume significantly less time as opposed to graph parsing. An inconsistency however has been documented, with regard to the processing of complex sentences carrying multiple expressions, in how it considers only the polarity attributed to the terminal element. Going forward, the component BiLSTM could be supplanted with BiLSTM CRF and run, owing to the nature of the latter to be robust and independent of word embedding, and capability to provide superior accuracy levels on Parts of Speech tagging, Name Entity Recognition of data sets and chunking.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Adnan Akhundov, Dietrich Trautmann, and Georg Groh. 2018. Sequence labeling: A practical approach. *CoRR*, abs/1808.03926.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Mohammad Ehsan Basiri, Shahla Nemati, Moloud Abdar, Erik Cambria, and U. Rajendra Acharya. 2021. Abcdm: An attention-based bidirectional cnn-rnn deep model for sentiment analysis. *Future Generation Computer Systems*, 115:279–294.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610. IJCNN 2005.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. volume 2, pages 1045–1048.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, J.H. Cernocky, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. pages 5528 – 5531.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Huyen Trang Phan, Van Cuong Tran, Ngoc Thanh Nguyen, and Dosam Hwang. 2020. Improving the performance of sentiment analysis of tweets containing fuzzy sentiment using the feature ensemble model. *IEEE Access*, 8:14630–14641.

Animesh Prasad and Min-Yen Kan. 2017. WING-NUS at SemEval-2017 task 10: Keyphrase extraction and classification as joint sequence labeling. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 973–977, Vancouver, Canada. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210.

Guixian Xu, Yueting Meng, Xiaoyu Qiu, Ziheng Yu, and Xu Wu. 2019. Sentiment analysis of comment texts based on bilstm. *IEEE Access*, 7:51522–51532.

Ashima Yadav and Dinesh Kumar Vishwakarma. 2020. Sentiment analysis using deep learning architectures: a review. *Artificial Intelligence Review*, 53(6):4335–4385.

MT-Speech at SemEval-2022 Task 10: Incorporating Data Augmentation and Auxiliary Task with Cross-Lingual Pretrained Language Model for Structured Sentiment Analysis

Cong Chen, Jiansong Chen, Cao Liu, Fan Yang, Guanglu Wan, Jinxiong Xia,
Meituan, Beijing, China
{chencong29, chenjiansong, liucao}@meituan.com

Abstract

Structured Sentiment Analysis (SSA) is an important component of sentiment analysis, which is a critical task in NLP. Traditional SSA methods cannot capture the cross-lingual interactions between different language and there is insufficient annotated data, especially in the cross-lingual settings. In this paper, we use the pre-trained language models with two auxiliary tasks and adopt data augmentation to address the above problems. Specifically, we employ XLM-RoBERTa to capture the cross-lingual knowledge interactions and enhance the generalization in multilingual/cross-lingual settings. Furthermore, we leverage two data augmentation techniques and propose two auxiliary tasks to improve the performance on the few-shot and zero-shot settings. Experiments demonstrate that our model ranks first on the cross-lingual sub-task and second on the monolingual sub-task of SemEval-2022 task 10.

1 Introduction

Structured Sentiment Analysis (SSA) is an important task in sentiment analysis (Barnes et al., 2021; Liu, 2012; Mitchell et al., 2013). The goal of SSA is to extract all opinion tuples from given texts. The opinion tuple (h, t, e, p) consists of a holder (h) which expresses a polarity (p) towards a target (t) by a textual sentiment expression (e). Benefiting a variety of business applications, such as human-machine dialogue and recommendation systems, SSA has attracted much more attention from both academia and industry (Pang et al., 2008; Mitchell et al., 2013; Xu et al., 2020; Ovrelid et al., 2020; Li et al., 2019).

The mainstream method for SSA is to adopt a pipeline approach by separately performing the subtasks including holder extraction and target extraction. However, such methods can not capture dependencies of multiple sub-tasks. To address this problem, Barnes et al. (2021) leverages graph-based dependency parsing to capture the dependencies among opinion tuples, where sentiment holders, targets and expressions are the nodes, and the relations of them as the arcs. This model has obtained state-of-the-art performance on SSA.

However, the aforementioned methods still suffer from some important issues. Firstly, the knowledge of the pre-trained language models (PLMs) has not been fully exploited. In fact, the cross-lingual PLMs contain rich knowledge of the interactions among different languages. Secondly, the above data-driven models rely on a large amount of annotation data, but there is insufficient or even no annotated data in the real scene. For example, in SemEval-2022 shared task 10 (Barnes et al., 2022), the **MultiB**$_{EU}$ (Barnes et al., 2018) dataset has only 1215 sentences and the **MultiB**$_{CA}$ (Barnes et al., 2018) dataset have only 1341 sentences, and there is no training data for the target language in the cross-lingual setting, which heavily hinders the performance on SSA.

To address the above problems, we propose a unified and end-to-end model for SSA, which performs data augmentation and adopts auxiliary tasks with cross-lingual PLMs. Specifically, we employ XLM-RoBERTa (Conneau and Lample, 2019; Conneau et al., 2019) as the backbone encoder to make use of its multilingual/cross-lingual knowledge. To alleviate the problem of insufficiency or lack of annotated data, we adopt two data augmentation methods: the one is to add in-domain annotated data of the same task under the training stage, and the other is to employ Masked Language Model (MLM) (Devlin et al., 2018) for generating similar texts. Furthermore, in addition to predicting each tuple in the dependency parsing graph simultaneously, we add two auxiliary tasks: 1) sequence labeling to predict the span of the holder / target / expression, and 2) sentiment polarity classification. Note that both of them do not need additional

Methods	**NoReC**$_{Fine}$	**MultiB**$_{CA}$	**MultiB**$_{EU}$	**OpeNer**$_{EN}$	**OpeNer**$_{ES}$	**MPQA**	**DS**$_{Unis}$	Average
Top1	0.529(2)	0.728(1)	0.739(1)	0.760(2)	0.722(4)	0.447(1)	0.494(1)	0.631(1)
Top2(Ours)	0.524(3)	0.728(1)	0.739(1)	0.763(1)	0.742(1)	0.416(2)	0.485(2)	0.628(2)
Top3	0.533(1)	0.709(3)	0.715(3)	0.756(3)	0.732(3)	0.402(3)	0.463(3)	0.616(3)
Top4	0.504(4)	0.681(6)	0.723(2)	0.747(4)	0.735(2)	0.375(5)	0.410(9)	0.596(4)
Top5	0.483(8)	0.711(2)	0.681(6)	0.727(5)	0.686(7)	0.379(4)	0.373(13)	0.577(5)

Table 1: Comparisons on monolingual evaluation leader board.

Methods	**OpeNer**$_{ES}$	**MultiB**$_{CA}$	**MultiB**$_{EU}$	Average
Top1(Ours)	0.644(1)	0.643(1)	0.632(1)	0.640(1)
Top2	0.618(3)	0.562(7)	0.584(2)	0.588(2)
Top3	0.628(2)	0.607(3)	0.527(4)	0.587(3)
Top4	0.604(5)	0.596(4)	0.512(7)	0.571(4)
Top5	0.589(6)	0.593(5)	0.516(6)	0.566(5)

Table 2: Comparisons on cross-lingual evaluation leader board.

annotations.

We conduct experiments on subtask 1 and subtask 2 of SemEval-2022 shared task on SSA. Experimental results demonstrate that our method outperforms strong baselines. We rank first on the cross-lingual sub-task and rank second on the monolingual subtask in SemEval-2022 task 10[1].

To summarize, our contributions are as follows.

- We leverage cross-lingual pre-trained language models to capture the interactive information knowledge among different languages.
- We combine existing in-domain training data and produce new training data by MLM to alleviate the problem of insufficiency or lack of annotated data.
- We propose two auxiliary tasks that do not require additional annotations to further improve the performance.
- Experimental results demonstrate the effectiveness of our proposed model, and we rank first on the subtask 2 and rank second on the subtask 1 on SemEval-2022 task 10.

2 Method

We incorporate the dependency graph parsing approach (Barnes et al., 2021) into our model. The general architecture is a pre-trained language model (e.g BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), etc..) followed by a three-layers BiLSTMs (Schuster and Paliwal, 1997; Cross and Huang, 2016) and the bilinear (Kiperwasser and Goldberg, 2016; Dozat and Manning, 2016) attention as the decoding component. Hence, we can take advantage of the knowledge of large-scale PLMs (Vaswani et al., 2017; Radford et al., 2019; Raffel et al., 2020; Yang et al., 2019; Wolf et al., 2019) and deep semantic dependency parsing (Dozat and Manning, 2016; Oepen et al., 2020; Kurtz et al., 2020).

2.1 Encoder

We consider several state-of-the-art models as the candidates of our model's backbone, such as Multilingual BERT (mBERT) (Devlin et al., 2018), XLM-RoBERTa (Conneau et al., 2019), and info-XLM(Chi et al., 2021). Particularly, we choose the XLM-RoBERTa backbone as the baseline. Because subtask 1 is a multilingual problem and subtask 2 is a cross-lingual zero-shot problem. They both benefit from the Translation Language Modeling (TLM) objective in XLM-RoBERTa. The TLM and Masked Language Modeling (MLM) objectives in the XLM-family models perform better than mBERT, which is simply trained on multilingual corpus with the MLM objective. Additionally, XLM-RoBERTa is trained on more data, which makes it more robust. Another reason we choose XLM-RoBERTa is that it is a large open-source model for downstream applications. We did not employ info-XLM as it is trained with the sentence-level classification objective, which is not suited for this task.

[1]https://competitions.codalab.org/competitions/33556

Methods	MPQA	DS$_{Unis}$	OpeNer$_{EN}$	OpeNer$_{ES}$	MultiB$_{CA}$	MultiB$_{EU}$	NoReC$_{Fine}$	Average
w2v + BiLSTM	0.103	0.166	0.525	0.526	0.524	0.539	0.320	0.386
mBERT	0.231	0.280	0.571	0.611	0.526	0.517	0.373	0.446
mBERT +BiLSTM	0.266	0.285	0.621	0.614	0.619	0.589	0.386	0.483
XLM-R +BiLSTM	0.332	0.357	0.705	0.654	0.669	0.650	0.481	0.550

Table 3: Monolingual task performances with different encoders. All models use the same bilinear attention decoder. All results are evaluated on the official released development set.

2.2 Data Augmentation

We provide two data augmentation methods to further boost the performance of our model. First is the in-domain data enhancement (DA1) to better make use of the data in different languages. The second is the MLM data augmentation (DA2).

2.2.1 In-domain Data Enhancement

We combined different dataset that belong to the same domain in the training phase, to help improve generalization. Noticed that the four datasets **MultiB**$_{EU}$, **MultiB**$_{CA}$, **Opener**$_{ES}$, and **Opener**$_{EN}$ (Agerri et al., 2013) are all from the hotel review corpus. We observe these datasets share some common features even though they are of different languages. These languages share the same or similar words for the same objects or concepts. For example, the "hotel" word in Catalan and Spanish are also "hotel", and in Basque it is a similar word "hotela". Besides, the people who use these languages share the same sentiment polarity tendency on the hotel review domain. Combining the four language datasets together as a whole training set will improve the overall performance. We additionally add the Portuguese hotel review dataset (BOTE-rehol)(Barros and Bona, 2021) and the English laptops review dataset (RES14) (Pontiki et al., 2014; Xu et al., 2020) for extra training, which needs to be converted to the same format as this task.

2.2.2 Data Augmentation by Masked Word Generation

The Masked Language Model corrupts the input texts by randomly replacing the tokens with the [MASK] tokens, and predicts the original token at the [MASK] positions. For each sample with valid opinion tuples, by randomly masking a small portion of the tokens in the text, we obtain a new sample whose meaning is similar as the original with the same labels. Note that in this task we do not mask the sentiment expression words as the PLMs may generate words of different polarities which are inconsistent with the original labels.

2.3 Auxiliary Tasks

SSA consists of structure prediction and sentiment polarity classification, and to handle these two tasks in an end-to-end manner is non-trivial. We propose two auxiliary tasks to provide more training signals to the model to better handle structure prediction and polarity classification. For structure prediction, we add a sequence labeling task to explicitly predict the type (target, holder, or expression) of each token. For polarity classification, we add more sentiment polarity classification data as extra tasks. Specifically, we use the average pooling of the model's BiLSTM hidden-states as sentence-level representations. The representation is fed to a multilayer perceptron(MLP) for sentence-level sentiment polarity classification. The total loss is the weighted sum of the main loss and the auxiliary losses:

$$\mathcal{L} = \mathcal{L}^p + (\mathcal{L}^s + \mathcal{L}^c)/2 \quad (1)$$

where $\mathcal{L}^p$ is the primary loss of the SSA task. $\mathcal{L}^s$ and $\mathcal{L}^c$ are the losses for sequence-labeling and classification task, respectively.

3 Experiments

3.1 Experimental data

We use the officially released development set as the test set, and randomly split the original training set into the training and development sets. We keep the size of the split development sets the same as the official released development set.

We transform the BOTE-rehol and RES14 datasets into the graph format and leave the opinion tuples' holders empty since the two datasets do not contain holder labels.

For each dataset, we convert the sentiment graph labels to sequence labeling labels, which will be added as an auxiliary task during training. Additionally, for the **MultiB**$_{CA}$ and **Opener**$_{ES}$ datasets, we make use of the Catalonia Independence Corpus (CIC) (Zotova et al., 2020) as the extra training

Methods	**MultiB**$_{CA}$	**MultiB**$_{EU}$	**OpeNer**$_{ES}$	**OpeNer**$_{EN}$
baseline	0.685	0.650	0.654	0.712
w / DA1	0.727	0.670	0.711	0.729

Table 4: Monolingual performances of data augment (DA) on official released development set.

Methods	**MultiB**$_{CA}$	**MultiB**$_{EU}$	**OpeNer**$_{ES}$
OpeNer$_{EN}$	0.574	0.438	0.630
w / DA1	0.600	0.550	0.620
w / DA1-2	0.623	0.567	0.657

Table 5: Cross-lingual performances of data augment (DA) on official released development set.

data of the polarity classification task.

3.2 Implementation details

We adopt the head-first setting (Barnes et al., 2021) which sets the first token within each span as the head of the span with all other tokens within that span as dependents. The root node is represented by the first token within the sentiment expression. For any word that is tokenized into a head-token followed by several sub-tokens, we set the head-token as the head and its following sub-tokens as the dependents. We use *holder, targ, exp-Positive, exp-Negative, exp-Neutral, None* as labels to denote different node types. The relation between each node is expressed by the attention value between the head-tokens of the nodes.

We try different combinations to get the best results for different subtasks. With XLM-RoBERTa-large as the backbone, details combinations are listed in Table 6 for the four hotel review datasets (**MultiB**$_{EU}$, **MultiB**$_{CA}$, **Opener**$_{ES}$ and **Opener**$_{EN}$). For **DS**$_{Unis}$ dataset (Toprak et al., 2010), we chose English RES14 dataset which also has few holders elements as its in-domain dataset. As most of the sentences in the two dataset are expressed without holders elements.

When generating new samples via MLM, for each sentence with at least one valid sentiment tuple, we mask one position i at a time and feed the masked sentence to the XLM-RoBERTa-large model. The PLM generates a word based on the highest probability p_i. We pick the top 5 most confident samples ranked by the PLM's output probability P_i for $i \in n$, where n denotes the possible masked positions. And set a threshold p as 0.85 to filter out any samples with a probability lower than the threshold. Repeated samples are not considered valid. The generated samples are treated as supplementary data to the original dataset.

For domain adaptation, we further pre-trained XLM-RoBERTa-large with all the data from the released datasets via Mask Language Modeling (MLM)(Devlin et al., 2018). We pick the best checkpoint according to the lowest perplexity on the development set.

3.3 Overall Comparisons

Comparison Settings. Firstly, we compare our model with other participant teams on the leader board of the structured sentiment competition. Table 1 and Table 2 record the comparison results of the monolingual and cross-lingual evaluation, respectively.

Comparison Results. (1) Our methods rank second and first on the monolingual and cross-lingual evaluation, respectively, which demonstrates the effectiveness of our proposed model. (2) Our model remarkably outperforms the top2 team in the cross-lingual subtask, which indicates our model has better generalization on the zero-shot cross-lingual settings.

3.4 Effectiveness of Cross-lingual Pre-trained Language Model

Comparison Settings. To prove the effectiveness of XLM-RoBERTa[2], a cross-lingual pre-trained language model, we compare it with the following baselines: 1) w2v + BiLSTM, BiLSTMs with word2vec (Mikolov et al., 2013) word embeddings; 2) mBERT, the Multilingual BERT (Devlin et al., 2018); 3) mBERT + BiLSTM; 4) XLM-RoBERTa + BiLSTM.

Comparison Results. (1) Table 3 demonstrates that XLM-RoBERTa + BiLSTM obtains the best performance among all of the benchmarks, and the average score outperforms the strongest baseline (mBERT + BiLSTM) by 6.7%. It proves that our model has great generalization ability. (2) BiLSTM can improve the performance by 3.7%, which indicates the BiLSTM layer can capture sequence information, which is beneficial to sequence encoding (Cross and Huang, 2016).

[2]We leverage the large version of XLM-RoBERTa to improve performances.

Methods	**MultiB**$_{CA}$	**MultiB**$_{EU}$	**OpeNer**$_{ES}$
Data combination	**OpeNer**$_{ES}$ **MultiB**$_{EU}$	**OpeNer**$_{EN}$ **MultiB**$_{CA}$ **OpeNer**$_{ES}$ bote-rehol	**OpeNer**$_{EN}$ **MultiB**$_{CA}$

Table 6: In domain data combination for cross-lingual evaluation. No target language data is participated in training and development.

Methods	**MPQA**	**DS**$_{Unis}$	**OpeNer**$_{EN}$	**OpeNer**$_{ES}$	**MultiB**$_{CA}$	**MultiB**$_{EU}$	**NoReC**$_{Fine}$
baseline	0.296	0.337	0.648	0.641	0.662	0.647	0.400
w / Auxiliary-task	0.305	0.346	0.674	0.660	0.687	0.657	0.411

Table 7: Performances on the official released development set with auxiliary tasks. We use RoBERTa-base (Liu et al., 2019) for **MPQA** (Wiebe et al., 2005), **DS**$_{Unis}$ and **OpeNer**$_{EN}$, bert-base-spanish-wwm-cased (Cañete et al., 2020) for **OpeNer**$_{ES}$, RoBERTa-base-ca (Armengol-Estapé et al., 2021) for **MultiB**$_{CA}$, berteus-base-cased (Agerri et al., 2020) for **MultiB**$_{EU}$, and norwegian-RoBERTa-base `https://huggingface.co/patrickvonplaten/norwegian-roberta-base` for **NoReC**$_{Fine}$ (Øvrelid et al., 2020).

3.5 Effectiveness of Data Augmentation

Comparison Settings. In order to demonstrate the effectiveness of data augmentation, we utilize existing training data for data augmentation (DA1) including **MultiB**$_{EU}$ **MultiB**$_{CA}$, **OpeNer**$_{ES}$ and **OpeNer**$_{EN}$. Furthermore, we leverage MLM to generate new training data for data augmentation (DA2). We record the performance in Table 4 and Table 5, where "w/" means "with", and "DA1-2" means "DA1 combined with DA2".

Comparison Results. We can conclude the following from Table 4 and Table 5: both DA1 and DA2 contribute to performance improvement, with performance increases on almost every benchmark. Specifically, the performance has remarkably improved in the cross-lingual settings, and data augmentation is more helpful on the few-shot and zero-shot settings.

3.6 The Effectiveness of Auxiliary Tasks

As shown in Table 7, we leverage the auxiliary tasks including sequence labeling and sentiment polarity classification to improve the performances. We can observe that the auxiliary tasks improve performances on all of the datasets, which demonstrate the effectiveness of the two auxiliary tasks.

4 Conclusion

This paper studies the task of structured sentiment analysis. In order to deal with the problems of poor interactions of different languages and lack of annotation data, we adopt the cross-lingual pre-trained language model and adopt data augmentation and auxiliary tasks. Specifically, we employ XLM to capture the interactive information in the pre-training stage. Furthermore, we leverage two data augmentation strategies and two auxiliary tasks to improve the performance for lack of training data. Experiments demonstrate the effectiveness of our models. Our models rank first on the cross-lingual sub-task and rank second on the monolingual sub-task of SemEval-2022 task 10.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Rodrigo Agerri, Iñaki San Vicente, Jon Ander Campos, Ander Barrena, Xabier Saralegi, Aitor Soroa, and Eneko Agirre. 2020. Give your text representation models some love: the case for basque. In *Proceedings of the 12th International Conference on Language Resources and Evaluation*.

Jordi Armengol-Estapé, Casimiro Pio Carrino, Carlos Rodriguez-Penagos, Ona de Gibert Bonet, Carme Armentano-Oller, Aitor Gonzalez-Agirre, Maite Melero, and Marta Villegas. 2021. Are multilingual models the best choice for moderately under-resourced languages? A comprehensive assessment for Catalan. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4933–4946, Online. Association for Computational Linguistics.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*

(LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Oberländer Laura Ana Maria Kutuzov, Andrey and, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

José Meléndez Barros and Glauber De Bona. 2021. A deep learning approach for aspect sentiment triplet extraction in portuguese. In *Brazilian Conference on Intelligent Systems*, pages 343–358. Springer.

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

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and M. Zhou. 2021. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. In *NAACL*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

James Cross and Liang Huang. 2016. Incremental parsing with minimal features using bi-directional lstm. *ArXiv*, abs/1606.06406.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

E. Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Robin Kurtz, Stephan Oepen, and Marco Kuhlmann. 2020. End-to-end negation resolution as graph parsing. In *IWPT*.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. *ArXiv*, abs/1811.05082.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

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

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *EMNLP*.

Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Timothy J. O'Gorman, Nianwen Xue, and Daniel Zeman. 2020. Mrp 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *CONLL*.

Lilja Ovrelid, Petter Maehlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for norwegian. In *LREC*.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in information retrieval*, 2(1–2):1–135.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Haris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *COLING 2014*.

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

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

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

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Elena Zotova, Rodrigo Agerri, Manuel Nunez, and German Rigau. 2020. Multilingual stance detection: The catalonia independence corpus. *arXiv preprint arXiv:2004.00050*.

ECNU_ICA at SemEval-2022 Task 10: A Simple and Unified Model for Monolingual and Crosslingual Structured Sentiment Analysis

Qi Zhang[1], Jie Zhou[2*], Qin Chen[1], Qingchun Bai[3], Jun Xiao[3], and Liang He[1]

[1]School of Computer Science and Technology East China Normal University, China
[2]School of Computer Science, Fudan Univerisity, China
[3]Shanghai Open University, China
qzhang@stu.ecnu.edu.cn, jzhou@ica.stc.sh.cn
{qchen,lhe}@cs.ecnu.edu.cn, qc_bai@foxmail.com, xiaoj@sou.edu.cn

Abstract

In this paper, we focus on the structured sentiment analysis task that is released on SemEval-2022 Task 10. The task aims to extract the structured sentiment information (e.g., holder, target, expression and sentiment polarity) in a text. We propose a simple and unified model for both the monolingual and crosslingual structured sentiment analysis tasks. We translate this task into an event extraction task by regarding the expression as the trigger word and the other elements as the arguments of the event. Particularly, we first extract the expression by judging its start and end indices. Then, to consider the expression, we design a conditional layer normalization algorithm to extract the holder and target based on the extracted expression. Finally, we infer the sentiment polarity based on the extracted structured information. We conduct the experiments on seven datasets in five languages. It attracted 233 submissions in monolingual subtask and crosslingual subtask from 32 teams. Finally, we obtain the top 5 place on crosslingual tasks.

1 Introduction

The identification of sentiment in the text is an important field of study. Users' opinions on products, events, topics, and so on are valuable for both the company and government to improve products or policies. Recently, more and more researchers focus on fine-grained sentiment analysis tasks, such as structured sentiment analysis, which can be formulated into tuple extraction from the context (Wiebe et al., 2005).

We focus on structured sentiment analysis, which is released by SemEval 2022 task 10. Formally, the task aims to extract all of the opinion tuples $O = \{O_i, ..., O_n\}$ in a text. A tuple (h, t, e, p) can represent the structure of sentiment in context. As shown in Figure 1, "Some other " (h) is a holder who expresses a polarity "positive" (p)

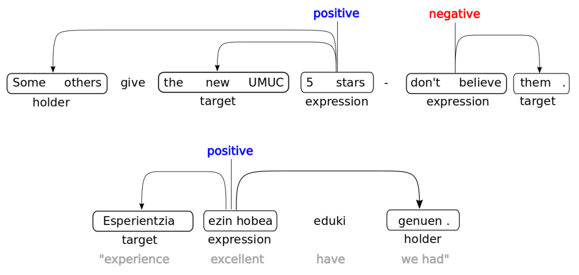


Figure 1: A structured sentiment graph is composed of a holder, target, sentiment expression, their relationships and a polarity attribute. Holders and targets can be null.

towards a target "the new UMUC" (t) through a sentiment expression "5 stars" (e), implicitly defining the relationships between the elements of a sentiment graph. Moreover, the SemEval task 10 is divided into two settings of increasing complexity: **Setting A** trains and predicts the sentiment polarity on monolingual. **Setting B** trains on the source language and tests on the target language. Particularly, we can train on any of the other datasets, as well as any other resource that does not contain sentiment annotations in the target language. The datasets used for this task are provided by the competition organizers, who collected and annotated the corpus (Barnes et al., 2021).

Structured sentiment analysis can be resolved down into four sub-tasks: a) expression extraction, b) target extraction, c) holder extraction, and d) sentiment polarity classification. Previous work on information extraction has used pipeline methods that extract the holders, targets, and expressions (tasks a-c). Most of the works conduct experiments on the MPQA dataset (Wiebe et al., 2005). Choi et al. (2006); Yang and Cardie (2012) used CRFs with the extracted features (e.g., named-entity tagger, sentiment lexicons, and dependency parsers), which results in a very strong baseline. For a complicated task with a small size of the training data, these feature-based mechanism learning techniques often still perform better than neural-based models,

[*] Jie Zhou is the corresponding author of this paper.

such as Bi-LSTMs (Katiyar and Cardie, 2016). The end-to-end approaches have shown some potential by learning the relationships among the multiple subtasks (Zhang et al., 2019; Zhou et al., 2020a). However, all of these studies ignore the sentiment polarity classification subtask. Barnes et al. (2021) regarded the structured sentiment analysis task as a dependency parsing task to model the relationships among the elements.

In this paper, we transform this task into an event extraction framework. Because we found the expression word is the same as the trigger word in the event extraction, both of them can uniquely represent the whole structured tuple. Besides, we regard the other elements (e.g., holder and targets) as the arguments of the sentiment structured information. Above all, we propose a pipeline model for this task. The key problem for this task is how to model the relationships among the targets, holders and expressions. To take the expressions into account, we introduce a conditional layer normalization method to extract the holders and targets w.r.t. the expression. Our simple and unified model is appropriate for both monolingual and crosslingual tasks. Experiments on datasets of monolingual and crosslingual tasks show our approach is capable to improve performance significantly on structured sentiment analysis as event extraction.

2 Related Work

2.1 Structured Sentiment Analysis

The goal of structured sentiment analysis is to extract the holders, expressions, and polarities w.r.t. the targets (Zhou et al., 2019). Early researches formulated the subtasks of structured sentiment analysis into independent span extraction or relation extraction. Choi et al. (2006) investigated a joint approach to extract entities and relations at the same time for opinion recognition and analysis. Yang and Cardie (2012) proposed semi-Markov conditional random fields (semi-CRFs) to extract opinion expressions at segment level. Katiyar and Cardie (2016) used deep bidirectional LSTMs for joint extraction of opinion entities and relations (e.g., the IS-FROM, IS-ABOUT) that connect the entities to extract expressions with their associated holders and targets. Zhang et al. (2019) presented a transition-based end-to-end method to extract the elements (e.g., holders, targets, and expressions) with their relationships. Then, to take the sentiment polarity into account, systems like IMN (He et al., 2019), SK-GCN (Zhou et al., 2020b) and RACL (Chen and Qian, 2020) have been developed.

Moreover, Barnes et al. (2021) introduced a parsing-based algorithm that implements a real unified structured sentiment analysis. They regarded structured sentiment analysis as a dependent parsing problem to model the relatedness among the holder, target, expression, and polarity. Different from these works, we translate structured sentiment analysis to an event-extraction framework.

2.2 Multilingual Sentiment Classification

Traditional methods for multilingual sentiment analysis are based on machine translation. Then, neural-based models product the source and target languages into a common space via parallel data or dictionaries. Lample and Conneau (2019) proposed two methods to learn cross-lingual language models: one unsupervised that uses cross language modeling to learn cross language representation. And one supervised that leverages parallel data with a new cross-lingual language model objective. Tellez et al. (2017) trained an SVM classifier by leveraging language-dependent and independent features. However, these machine learning approaches also require a feature extraction phase. We eliminate by incorporating deep learning approaches since they can learn features automatically. Furthermore, (Wan, 2008) designed a new approach to improve Chinese sentiment analysis using reliable English datasets. Recently, multi-lingual pre-trained language models (e.g., mBert (Devlin et al., 2019)) obtains state-of-the-art performance for crosslingual tasks.

3 Our Approach

In this paper, we propose a simple and unified model to extract the structured sentiment analysis. This section describes how we redefine the sentiment classification task to an event extraction task, detail the pipeline method. Fig. 3 shows the framework of our model, which consists of three components, expression extraction, arguments extraction (including holder extraction and target extraction), sentiment classification. First, we extract the expression based on (multi-lingual) pretrained language models (PLMs) (See Subsection 3.2). Second, we design a conditional layer normalization strategy to extract the holders and targets by incorporating the expression. Finally, we perform sentiment classification tasks based on the sentence

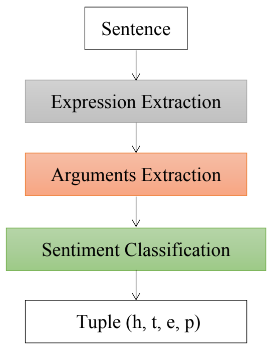


Figure 2: The process of our solution, It consists of three parts: Expression Extraction, Argument Extraction, Sentiment classification.

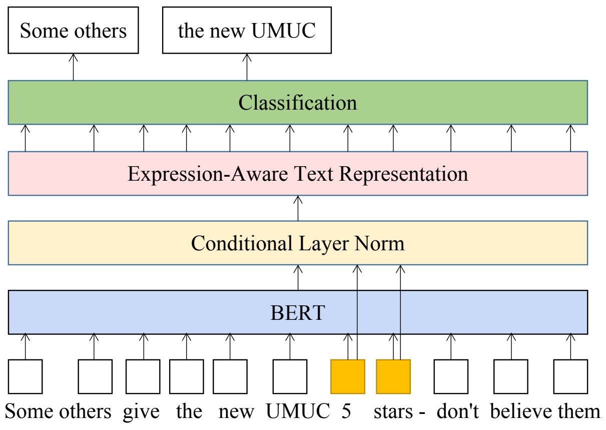

Some others
the new UMUC
Classification
Expression-Aware Text Representation
Conditional Layer Norm
BERT
Some others give the new UMUC 5 stars - don't believe them

Figure 3: The framework of our model

representation learned by the previous part (See Subsection 3.3 and 3.4).

3.1 Task Definition

Given a sentence $s = \{w_1, ..., w_{|s|}\}$, where w_i is the i-th word in the sentence s, which contains $|s|$ words. The goal of this task is to extract all the sentiment tuples $O = \{\mathrm{O}_1, ..., \mathrm{O}_{|O|}\}$ in the text, where the tuple (e, a, p) consists of expression e, arguments a (target t, holder h) and sentiment polarities c tuple. The event sentiment polarity class $c \in \{P, N, O\}$, which represents positive, negative and neutral.

3.2 Expression Extraction

Through the data analysis, we found that the expression uniquely identifies the tuple O since the targets and holders may not exist in the text. Thus, we extract the expression first by regarding it as the trigger. We first obtain the contextual word embedding based on the PLMs. For the monolingual setting, we use language-specific PLMs. For crossligual setting, we use the multi-lingual PLMs. Particularly, we input the sentence s into PLMs to obtain the word embedding $X = \{x_1, x_2, ..., x_{|s|}\}$,

$$X = \mathrm{PLM}(s) \tag{1}$$

Then, we extract the expression by two token-level binary classifiers to predict the start and end indices of the expression. A linear layer with an activation function is adopted as the classifier,

$$\begin{aligned} p_e^s &= \mathrm{Sigmoid}(W_e^s X + b_e^s) \\ p_e^e &= \mathrm{Sigmoid}(W_e^e X + b_e^e) \end{aligned} \tag{2}$$

where p_e^s and p_e^e are the predicted probability distribution of expression e's start and end indices, $W_e^s, W_e^e, b_e^s, b_e^e$ are the learnable weights.

3.3 Argument Extraction

The key challenge for extracting targets and holders with respect to the extracted expression is how to integrate the expression into the extractor model. Inspired by (De Vries et al., 2017), we design a conditional layer normalization method to take the expression into account 3. Specifically, we use the representation of the expression to control the β and γ in the layer normalization. In this way, we can obtain the expression-aware representation for target and holder extraction. The word representation h_i after the conditional layer normalization (CLN) is computed as,

$$\begin{aligned} h_i &= \mathrm{CLN}(x_i) = \gamma \frac{e_i - \mu}{\sigma} + \beta, \\ \gamma &= W_\gamma [x_{e^s}; x_{e^e}], \\ \beta &= W_\beta [x_{e^s}; x_{e^e}], \end{aligned} \tag{3}$$

where μ, σ are the mean and standard deviation of the elements in x_i, e^s and e^e are the start and end indices of the expression e, W_γ and W_β are the trainable parameters. We use the concatenation of the expression's start and end word representation as the expression representation.

Based on the expression-aware representation $H = \{h_1, h_2, ..., h_{|s|}\}$, we train two classifiers to predict the probability distribution of the start and end indices for targets and holders respectively.

$$\begin{aligned} p_t^s &= \mathrm{Sigmoid}(W_t^s H + b_t^s) \\ p_t^e &= \mathrm{Sigmoid}(W_t^e H + b_t^e) \\ p_h^s &= \mathrm{Sigmoid}(W_h^s H + b_h^s) \\ p_h^e &= \mathrm{Sigmoid}(W_h^e H + b_h^e) \end{aligned} \tag{4}$$

where p_t^s/p_h^s and p_t^e/p_h^e are the predicted probability distribution of target t/holder h's start and end in-

Table 1: The statistics information of datasets.

Dataset	Language	# sents	# holders	# targets	# expr.
NoReC$_{\text{Fine}}$	Norwegian	11437	1128	8923	11115
MultiB$_{\text{EU}}$	Basque	1521	296	1775	2328
MultiB$_{\text{CA}}$	Catalan	1678	235	2336	2756
OpeNer$_{\text{es}}$	Spanish	2057	255	3980	4388
OpeNer$_{\text{en}}$	English	2494	413	3850	4150
MPQA	English	10048	2279	2452	2814
DS$_{\text{Unis}}$	English	2803	86	1119	1119

dices, W_t^s, W_t^e, b_t^s, b_t^e/W_h^s, W_h^e, b_h^s, b_h^e are the learnable weights.

3.4 Sentiment Classification

Finally, we infer the sentiment polarity of the structured tuple (h, t, e). Since the text representation contains the expression information that can represent the tuple, we use a max pooling operator based on H to obtain the tuple representation r,

$$r = \text{MaxPooling}(H) \tag{5}$$

We input the tuple representation r into a sentiment classifier to predict the sentiment polarity distribution towards the target,

$$p_c = \text{Softmax}(W_c r + b_c) \tag{6}$$

where W_c and b_c are the parameters.

4 Experiments

In this section, we first give the experimental setup, including datasets, implementation details and evaluation metrics (Section 4.1). Then, we present the experimental results and analysis on both the monolingual and multilingual settings (See Section 4.2).

4.1 Experimental Setup

Datasets To evaluate the effectiveness of our model, we conduct our experiments on datasets of multiple languages, including English, Basque, Catalan, Norwegian. MultiB$_{\text{EU}}$, MultiB$_{\text{CA}}$ (Barnes et al., 2018) and NoReC$_{\text{Fine}}$ (Øvrelid et al., 2020) are the reviews data in Basque, Catalan, and Norwegian. MPQA (Wiebe et al., 2005) is a English dataset that contains expressions, holders, targets and their relationships. DS$_{\text{Unis}}$ (Toprak et al., 2010) contains labeled opinions for user reviews about universities and services. The OpenNER dataset consists of labeled reviews of hotels from the guests. And it's divided into two languages: OpeNer$_{\text{en}}$ in English, OpeNer$_{\text{es}}$ in Spanish. The statistics of the datasets are shown in Table 1. We can find that the size of the labeled data is limited, especially the number of labeled holders. For example, in DS$_{\text{Unis}}$ dataset, there are only 86 holders, which limits the performance of the neural models largely.

Implementation Details For the monolingual setting, we utilize the language-specific pre-trained language models (PLMs) as the word embedding, which are downloaded from Huggling Face [1]. We finetune the parameters on the training data for this setting. For the crosslingual setting, we use the multi-lingual PLMs (e.g., mBEER, XLM) and fix the parameters on the training phase. We use Adam optimizer with the learning rates of 1e-5. The dimensions of word embedding are 128. The max sequence length is 512. The dropout is 0.1. We train all models for 100 epochs and keep the model that performs best regarding F1 on the dev set. We use default hyperparameters from Kurtz et al. (2020) and run all of our models five times with different random seeds. The reported test results are based on the parameters that obtain the best performance on the development.

Evaluation Metrics Following the previous works (Barnes et al., 2021), as we are interested not only in extraction or classification but rather in the full structured sentiment task, we use Sentiment Graph F1 as the final metric.

4.2 Experimental Results and Analysis

For experimental results, we report the majority baseline for each language. Our unified model can perform on both the monolingual and crosslingual settings and obtain good performance on these tasks, our main experimental results are presented in Table 2 and 3. We follow metrics in (Barnes

[1]https://huggingface.co/models

Table 2: Top 15 Results for the monolingual setting.

	Monolingual						
User	NoReC$_{\text{Fine}}$	MultiB$_{\text{CA}}$	MultiB$_{\text{EU}}$	OpeNer$_{\text{en}}$	OpeNer$_{\text{es}}$	MPQA	DS$_{\text{Unis}}$
zhixiaobao	0.529 (2)	0.728 (1)	0.739 (1)	0.760 (2)	0.722 (4)	0.447 (1)	0.494 (1)
Cong666	0.524 (3)	0.728 (1)	0.739 (1)	0.763 (1)	0.742 (1)	0.416 (2)	0.485 (2)
gmorio	0.533 (1)	0.709 (3)	0.715 (3)	0.756 (3)	0.732 (3)	0.402 (3)	0.463 (3)
colorful	0.504 (4)	0.681 (6)	0.723 (2)	0.747 (4)	0.735 (2)	0.375 (5)	0.410 (9)
whu_stone	0.483 (8)	0.711 (2)	0.681 (6)	0.727 (5)	0.686 (7)	0.379 (4)	0.373 (13)
KE_AI	0.483 (8)	0.711 (2)	0.681 (6)	0.727 (5)	0.686 (7)	0.364 (7)	0.373 (13)
Fadi	0.484 (7)	0.704 (4)	0.703 (4)	0.725 (6)	0.698 (5)	0.254 (20)	0.420 (5)
lys_acoruna	0.462 (9)	0.653 (9)	0.680 (7)	0.698 (9)	0.692 (6)	0.349 (10)	0.414 (8)
QiZhang	**0.496** (5)	**0.684** (5)	**0.686** (5)	**0.676** (10)	**0.623** (11)	**0.351** (8)	**0.409** (10)
luxinyu	0.487 (6)	0.658 (8)	0.651 (9)	0.710 (7)	0.669 (8)	0.269 (19)	0.416 (7)
rafalposwiata	0.459 (10)	0.650 (10)	0.653 (8)	0.670 (11)	0.663 (9)	0.326 (13)	0.395 (12)
evanyfyang	0.213 (21)	0.635 (12)	0.639 (10)	0.703 (8)	0.642 (10)	0.350 (9)	0.449 (4)
robvanderg	0.366 (12)	0.648 (11)	0.605 (11)	0.632 (14)	0.614 (13)	0.296 (16)	0.344 (14)
psarangi	0.343 (15)	0.634 (13)	0.559 (12)	0.634 (13)	0.595 (14)	0.283 (17)	0.320 (17)
chx.dou	0.395 (11)	0.583 (14)	0.506 (13)	0.626 (15)	0.622 (12)	0.309 (14)	0.280 (20)

Table 3: Top 15 Results for crosslingual setting.

	Crosslingual		
User	EN-ES	EN-CA	EN-EU
Cong666	0.644 (1)	0.643 (1)	0.632 (1)
colorful	0.618 (3)	0.562 (7)	0.584 (2)
gmorio	0.628 (2)	0.607 (3)	0.527 (4)
whu_stone	0.604 (5)	0.596 (4)	0.512 (7)
QiZhang	**0.551** (10)	**0.615** (2)	**0.530** (3)
Fadi	0.589 (6)	0.593 (5)	0.516 (6)
hades_d	0.617 (4)	0.544 (10)	0.522 (5)
lys_acoruna	0.570 (7)	0.554 (8)	0.509 (8)
rafalposwiata	0.564 (8)	0.586 (6)	0.444 (12)
KE_AI	0.561 (9)	0.552 (9)	0.463 (11)
etms.kgp	0.542 (11)	0.506 (11)	0.431 (13)
jylong	0.375 (12)	0.474 (12)	0.504 (9)
ouzh	0.375 (12)	0.474 (12)	0.504 (9)
SPDB_Inn...	0.356 (13)	0.470 (13)	0.486 (10)
gerarld	0.321 (14)	0.269 (14)	0.303 (14)

et al., 2021) to use Graph F1 scores to evaluate structured sentiment extraction and classification.

In detail, Table 2 shows the results of the monolingual setting. For English, the top 15 participants lie between 62.6% and 76.0%, 30.9% and 44.7%, 28.0% and 49.4% F1 score on the OpeNer$_{\text{en}}$, MPQA, DS$_{\text{Unis}}$ datasets. For the Spanish dataset, the top 15 F1 scores lie between 62.2% and 72.2%. The participants in the middle of the table are quite close to each other. For Catalan, F1 scores range from 58.3% to 72.8%. For the Basque language, F1 scores range from 50.6% to 73.9% , with a gap of 20 points between first and last place. For the Basque dataset, F1 scores range from 50.6% to 73.9%, and most are in mid 60%. In this setting, we obtain ninth place based on the average score of all the datasets.

Table 3 shows the performance of crosslingual setting, which trains on English datasets and tests on the target languages including MultiBooked datasets and the OpeNER Spanish dataset. For Spanish, the top 15 F1 scores between 32.1% and 64.4%. For Catalan, F1 scores between 26.9% and 64.3%, which span a wide range. For the Basque dataset, the F1 scores range from 30.3% to 63.2%. Particularly, our model obtains the second and third places on the MultiB$_{\text{CA}}$ and MultiB$_{\text{EU}}$ languages.

5 Conclusions

In this paper, we propose a simple and unified model for both the monolingual and crosslingual structured sentiment analysis tasks. Different from the previous studies, we transform this task into an event extraction task. Mainly, we first design an expression extraction for extracting the expression, just like extracting the trigger words in the event extraction task. Then we predict the holder and target based on the extraction results of the previous step. The model performs well on seven datasets in five languages.

Acknowledgements

The authors wish to thank the reviewers for their helpful comments and suggestions. This research is funded by the Science and Technology Commission of Shanghai Municipality (No. 19511120200 and 21511100402). This research is also funded by the National Key Research and Development Program of China (No. 2021ZD0114002), the National Nature Science Foundation of China (No. 61906045), and Shanghai Science and Technology Innovation Action Plan International Cooperation project "Research on international multi language online learning platform and key technologies (No.20510780100)". The computation is performed in ECNU Multi-functional Platform for Innovation (001).

References

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of ACL*, pages 3685–3694.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 431–439.

Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. 2017. Modulating early visual processing by language. *Advances in Neural Information Processing Systems*, 30.

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

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of ACL*, pages 504–515.

Arzoo Katiyar and Claire Cardie. 2016. Investigating lstms for joint extraction of opinion entities and relations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 919–929.

Robin Kurtz, Stephan Oepen, and Marco Kuhlmann. 2020. End-to-end negation resolution as graph parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 14–24.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Eric S Tellez, Sabino Miranda-Jiménez, Mario Graff, Daniela Moctezuma, Ranyart R Suárez, and Oscar S Siordia. 2017. A simple approach to multilingual polarity classification in twitter. *Pattern Recognition Letters*, 94:68–74.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Xiaojun Wan. 2008. Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 553–561.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Bishan Yang and Claire Cardie. 2012. Extracting opinion expressions with semi-markov conditional random fields. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1335–1345.

Meishan Zhang, Qiansheng Wang, and Guohong Fu. 2019. End-to-end neural opinion extraction with a transition-based model. *Information Systems*, 80:56–63.

Jie Zhou, Jimmy Xiangji Huang, Qin Chen, Qinmin Vivian Hu, Tingting Wang, and Liang He. 2019. Deep learning for aspect-level sentiment classification: survey, vision, and challenges. *IEEE access*, 7:78454–78483.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020a. Modeling multi-aspect relationship with joint learning for aspect-level sentiment classification. In *International Conference on Database Systems for Advanced Applications*, pages 786–802. Springer.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020b. Sk-gcn: Modeling syntax and knowledge via graph convolutional network for aspect-level sentiment classification. *Knowledge-Based Systems*, 205:106292.

ZHIXIAOBAO at SemEval-2022 Task 10: Apporoaching Structured Sentiment with Graph Parsing

Yangkun Lin[*], **Chen Liang**[*], **Jing Xu**, **Chong Yang**[†], **Yongliang Wang**
Ant Group
Hangzhou, China
{linyangkun.lyk, liangchen.liangche, jill.xj,
yangchong.yang, yongliang.wyl}@antgroup.com

Abstract

This paper presents our submission to task 10, Structured Sentiment Analysis of the SemEval 2022 competition. The task aims to extract all elements of the fine-grained sentiment in a text. We cast structured sentiment analysis to the prediction of the sentiment graphs following (Barnes et al., 2021), where nodes are spans of sentiment holders, targets and expressions, and directed edges denote the relation types between them. Our approach closely follows that of semantic dependency parsing (Dozat and Manning, 2018). The difference is that we use pre-trained language models (e.g., BERT and RoBERTa) as text encoder to solve the problem of limited annotated data. Additionally, we make improvements on the computation of cross attention and present the suffix masking technique to make further performance improvement. Substantially, our model achieved the **Top-1** average Sentiment Graph F1 score on seven datasets in five different languages in the monolingual subtask.

1 Introduction

SemEval 2022 task 10 is a structured sentiment analysis task, aiming to predict all of the opinion tuples in a text. Each opinion O is a tuple (t, h, e, p), where h is a holder who expresses a polarity p towards a target t through a sentiment expression e. In practical, the task of structured sentiment analysis can help machines understand how people perceive ideas, policy etc.

This paper describes the system developed by the team ZHIXIAOBAO for SemEval-2022 Task 10. We follow the work of (Barnes et al., 2021) to cast the task as dependency graph parsing problem. The predicted opinion tuples are denoted by a directed graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for each sentence. As shown in Figure 1, all tokens in a sentence are presented as nodes and there are directed edges between the nodes to represent their relations. Each node in $\mathcal{V}$ can point to multiple nodes, and can have multiple incoming edges too. Sentiment expressions are regarded as roots in the structured sentiment graph (e.g., the token "5" and "don't" in Figure 1). Notice that not all nodes connect to the other nodes (e.g., the token "give" in Figure 1). The isolated tokens are the none-sentiment elements in the sentence, thus we should be able to predict the edge type of "null" in our model. Original edge types are defined as *holder*, *target*, and *expression*. Following the work of (Barnes et al., 2021), we also tried the **"+inlabel"** style of definition, where none-"null" edge types consist of $holder_{in}$, $holder_{out}$, $target_{in}$, $target_{out}$, $expression_{in}$, $expression_{out}$. Foot markers *in*, *out* denotes the in-span and out-span edges respectively. For example, the edge from "5" to "some" belongs to $holder_{out}$, while the edge from "Some" to "others" belongs to $holder_{in}$.

As demonstrated in previous work (Barnes et al., 2021), formulating the task as a graph structure prediction problem is superior to that of solving it by the span extraction and relation prediction approaches. The former can better extract overlapping spans than the latter. Thus, our model mainly follows the solution of dependency parsing to directly predict between-word relations. The model consists of a text encoder to extract contextual features of tokens, and a classifier to predict edges between each pair of tokens. A bilinear or biaffine cross attention is applied in the classifier layer to make multiplicative interactions between the features of a pair of tokens. In our model, pre-trained language models (e.g., BERT and RoBERTa) are used as the text encoder. We discover that fine-tuning the pre-trained language model brings huge enhancements in our experiments. In addition, as the meaning of in-span and out-span edge types are totally different, we leverage two cross attention for in-span edge types and out-span edge types prediction respectively. We also present a suffix mask-

[*] The first two authors contributed equally to this work.
[†] Corresponding author.

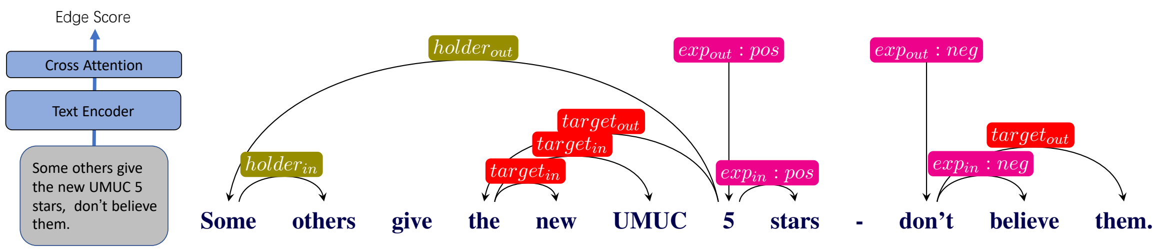


Figure 1: The framework of our model.

Dataset	Language	train	dev	test
NoReC$_{\text{Fine}}$	Norwegian	8634	1531	1272
MultiB$_{\text{CA}}$	Catalan	1174	168	336
MultiB$_{\text{EU}}$	Basque	1064	152	305
OpeNER$_{\text{ES}}$	Spanish	1438	206	410
OpeNER$_{\text{EN}}$	English	1746	249	499
MPQA	English	4500	1622	1681
DS$_{\text{Unis}}$	English	2253	232	318

Table 1: Summary of the datasets.

ing technique to reduce noise in the data. These techniques greatly improve the performance of our model compared with the original dependency parsing model.

Our model ranked **1st** out of 32 participating teams on monolingual subtask and got the highest average F1 score on 4 datasets.

2 Task Description

SemEval task 10 focuses on predicting all elements of the structured sentiment in a text, represented by opinion tuples (t, h, e, p), where h is a holder who expresses a polarity p towards a target t through a sentiment expression e. The evaluation is on seven datasets in five languages, the statistics of which are shown in Table 1.

NoReC$_{\text{Fine}}$ (Øvrelid et al., 2019) is the largest structured sentiment multi-domain dataset with professional reviews in Norwegian. **MultiB**$_{\text{EU}}$ and **MultiB**$_{\text{CA}}$ (Barnes et al., 2018) are hotel reviews datasets in Basque and Catalan, respectively. **OpeNER**$_{\text{EN}}$ and **OpeNER**$_{\text{ES}}$ (Agerri et al., 2013) are polarity-enhanced datasets with customer reviews in Spanish and English respectively. **MPQA** (Wiebe et al., 2005) annotates news articles in English from the world press. Finally, **DS**$_{\text{Unis}}$ (Toprak et al., 2010) is an annotated English reviews dataset of online universities and e-commerce.

Previous shared tasks on Aspect-Based Sentiment Analysis (ABSA) focus on extracting sentiment targets and classifying the polarity directly. Most previous methods follow the information extraction pipeline, which firstly extract the span of holders, targets, expressions and subsequently predict the relations. However, splitting structured sentiment analysis into subtasks may cause the error propagation problem. We follow the work that solving the problem by dependency graph parsing (Barnes et al., 2021) to achieve better performance in our model.

3 System Overview

The overview of our system is shown in Figure 1. We use a pre-trained language model to extract text information as the node features in the graph. Then, a cross attention layer is used to compute the predicted score of each edge type. After we get the edge score for each token pair, a graph parsing algorithm is presented to transform the predicted score to opinion tuples.

3.1 Text Encoder

We use the pre-trained language models, **BERT** and **RoBERTa**, to generate the contextual features of the text in multiple languages. Both of them are Transformer-based language models using a huge amount of text with a masked language model objective. These pre-trained language models have shown great superiority in a low-resource scenario like this task. Compared with BERT, RoBERTa removes next sentence prediction(NSP) loss and applies larger batch size and sequence length during the pre-training step, leading to a better performance in most cases. We tried the monolingual version of **RoBERTa**$_{\text{LARGE}}$ (Liu et al., 2019a) and **BERT**$_{\text{LARGE}}$ (Devlin et al., 2019) for each dataset, as our feature extractor. Our experimental results demonstrate that **RoBERTa**$_{\text{LARGE}}$ performs better than **BERT**$_{\text{LARGE}}$ in all the datasets.

3.2 Discrete Cross Attention

After extracting the text features, we can then use bilinear or biaffine attention to produce a score for each pair of tokens. The score includes multiplicative interactions among pairs, and can be used to predict edge types. Different from the previous work (Barnes et al., 2021), we model heads and dependents of in-span and out-span separately, because we think it is better to cast the in-span and out-span label prediction as two "different" tasks. According to our observation, they have different properties in the corpus. Thus, inspired by the multi-task learning framework, we propose to use **Discrete Cross Attention (DCA)** to make them share same bottom features in the text encoder, but have non-shared parameters in the computation of cross attention.

The contextual features C extracted from text encoder are processed with four layers of the feed-forward neural networks(FNN), FNN$^{in}_{head}$, FNN$^{in}_{dep}$, FNN$^{out}_{head}$ and FNN$^{out}_{dep}$ creating representations of potential heads and dependents for in-span and out-span respectively. And then a bilinear score is computed for each kind of edge types using a trainable parameter matrix A. The discrete cross attention can be formulated as bellow,

$$h_i^{in} = FNN_{head}^{in}(c_i) \tag{1}$$

$$d_i^{in} = FNN_{dep}^{in}(c_i) \tag{2}$$

$$score_{ij}^{in} = h_i^{inT} A_{in} d_j^{in} \tag{3}$$

$$h_i^{out} = FNN_{head}^{out}(c_i) \tag{4}$$

$$d_i^{out} = FNN_{dep}^{out}(c_i) \tag{5}$$

$$score_{ij}^{out} = h_i^{outT} A_{out} d_j^{out} \tag{6}$$

$$score_{ij} = softmax(score_{ij}^{in} \oplus score_{ij}^{out}) \tag{7}$$

$score_{ij}$ represents the final score list for each edge type, which is the softmax score of the concatenation of in-span edge scores $score_{ij}^{in}$ and out-span scores $score_{ij}^{out}$.

3.3 Graph Parsing

We set a threshold θ to determine whether the edge exists, i.e., if $max(score_{ij}) > \theta$, we set the predicted edge type to be $argmax(score_{ij})$, or we make the predicted edge to be "null". We

Algorithm 1: Graph parsing

```
Input: Sentiment graph $\mathcal{G}$
Output: Opinion Tuples $(H, T, E)$
Data: Opinion set $Op_{set}$
1  for $e_{r,i}$ in $\mathcal{G}$ do
2      if $e_{r,i} \in ExpTypes$ then
3          E ← $FindSpan(i, Type(e_{r,i}))$;
           new $H_{set}, T_{set}$;
4          for $e_{i,j}$ in $\mathcal{G}$ do
5              if $Type(e_{i,j}) \in HolTypes$ then
6                  $H_{set} \leftarrow FindSpan(j, hol)$
7              if $Type(e_{i,j}) \in TrgTypes$ then
8                  $T_{set} \leftarrow FindSpan(j, trg)$
9          for $H$ in $H_{set}$ do
10             for $T$ in $T_{set}$ do
11                 $Op_{set} \leftarrow (H, T, E)$
```

set $\theta = 0.5$ in our experiments. We use two kinds of graph parsing representations, head-first and head-final, following (Barnes et al., 2021). For head-first, we use the first token in the target/holder/expression spans as the head of the span and the other tokens within the span as the dependent. For head-final, we take the opposite way, i.e., set the final token of the target/holder/expression spans as the heads.

The algorithm of converting structured sentiment graph to opinion tuples (H, T, E) is in shown in **Algorithm 1**. H, T, E denote $holder$, $target$ and $expression$ respectively. $ExpTypes$, $HolTypes$, $TrgTypes$ are the edge type sets for $expression$, $holder$, $target$ respectively. $e_{i,j}$ denotes the predicted edge type between token i and token j, and r is the root nodes. $FindSpan(\cdot)$ is a function to find the complete span for a certain edge type, which can be simply implemented by merging linked tokens with the same edge type. As shown in **Algorithm 1**, we should first find the expressions, and then add the linked holders and targets for each expression to the opinion tuples. Notice that if we cannot find holder or target spans for an expression, we shall append an empty token into H_{set} or T_{set}.

3.4 Suffix Masking Trick

In both training and predicting procedure, a sentence is first tokenized by byte-pair encoding (BPE) before it is inputted into the text encoder, i.e., BERT

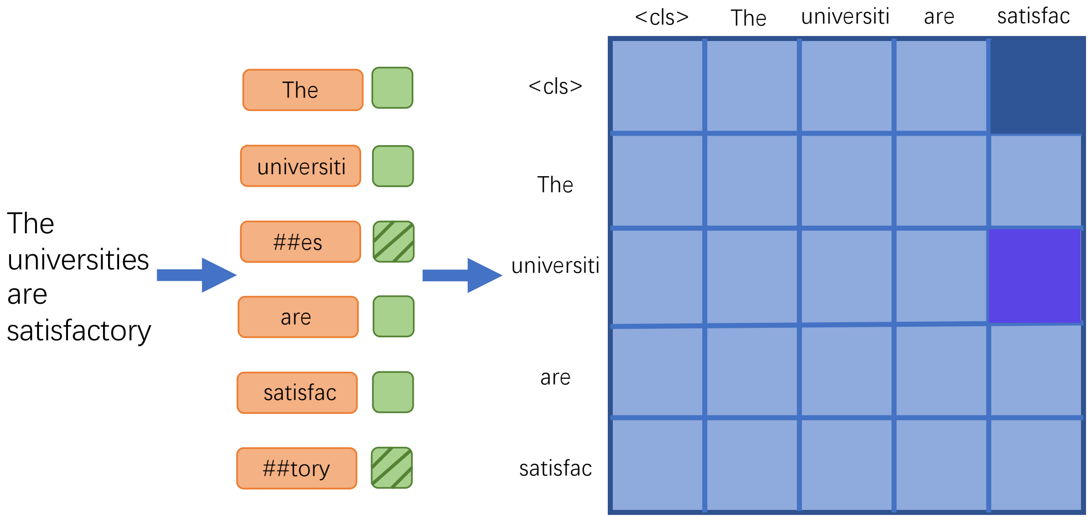


Figure 2: Suffix masking.

and RoBERTa. Some words are splitted into prefixes and suffixes in the procedure. For example, "universities" is tokenized to "universiti", "##es" as shown in Figure 2. We think these suffixes are often noises and provide few supervision signals for the edge prediction, since they are shared by many distinguished words. Inspired by this intuition, we mask these suffixes in the computation of edge scores. As shown in Figure 2, we mask the suffix, "##es" and "##tory", before the prediction of edges. In this way, the edges only exist between the pair of $(< cls >, satisfac)$ and $(universiti, satisfac)$.

4 Experiments

The experiments details and main results are shown in this section.

4.1 Experiment Details

The implementation of our model depends on pytorch and huggingface. In our experiments, **BERT**$_{\text{LARGE}}$ represents the monolingual version for each dataset, which all can be found in the huggingface website. As for **RoBERTa**$_{\text{LARGE}}$, the version of *xlm-roberta-large* (Conneau et al., 2019) is used on models for **MultiB**$_{\text{CA}}$, **MultiB**$_{\text{EU}}$ and **OpeNER**$_{\text{ES}}$, and *roberta-large-en-cased* (Liu et al., 2019b) is used on the English datasets, i.e., **MPQA**, **OpeNER**$_{\text{EN}}$ and **DS**$_{\text{Unis}}$.

We use Adam as our optimizer with the learning rate to be 1e-5 for the fine-tuning of pre-trained language models and 1e-4 for the other parameters in the model. The batch size is set to 12 with the gradient accumulation steps to be 48. The dropout rate is 0.3 and the hidden state size of FNN layers is set to 256. Our models are run on a maximum of 1000 epochs. We train all the models with 5 different seeds on the training set released by the orginizer and choose the best results based on the performance on development datasets. The training run on two Tesla V100 GPUs with 32G memory.

It has to be noted that we add a “<cls>” token when encoding the sentences and set the “<cls>” token as the root of the sentiment graph. In Figure 1, there actually exists an edge between <cls> token and the head of expression spans.

4.2 Metrics

To measure how well a system is able to capture the full sentiment graph, submitted systems are evaluated on sentiment graph F_1 (SF_1) following (Barnes et al., 2021). A true positive is defined as an exact match at graph-level, weighting the overlap in predicted and gold spans for each sentiment element, averaged across all three kinds of spans, i.e., expression, holder, target. For precision we weight the number of correctly predicted tokens divided by the total number of predicted tokens (for recall, we divide instead by the number of gold tokens).

4.3 Main Results

The main experimental results are shown in Table 2. It can be seen that using monolingual **BERT**$_{\text{LARGE}}$ pre-trained on larger language-specific corpus as text encoder is better than the multilingual **BERT**$_{\text{base}}$ used in (Barnes et al., 2021), and fine-tuning the pre-trained language models brings more improvements than freezing the parameters. An in-

Methods	NoReC$_{\text{Fine}}$	MultiB$_{\text{CA}}$	MultiB$_{\text{EU}}$	OpeNER$_{\text{ES}}$	OpeNER$_{\text{EN}}$	MPQA	DS$_{\text{Unis}}$
mBERT$_{\text{base}}$ w/o fine-tune + lstm (Barnes et al., 2021)	39.4	55.8	57.4	-	-	18.8	27.3
BERT$_{\text{LARGE}}$ w/o fine-tune + lstm	42.9	63.7	62.1	65.4	65.0	33.5	36.7
BERT$_{\text{LARGE}}$ fine-tune + lstm	50.4	69.1	65.2	67.1	69.2	42.1	40.8
BERT$_{\text{LARGE}}$ fine-tune	50.8	70.7	65.7	68.6	70.8	43.5	42.4
BERT$_{\text{LARGE}}$ fine-tune + inlabel	50.9	70.5	65.9	68.4	71.5	43.3	42.8
BERT$_{\text{LARGE}}$ fine-tune + inlabel + DCA	51.7	71.1	67.3	69.1	73.1	43.6	44.5
BERT$_{\text{LARGE}}$ fine-tune + inlabel + DCA + mask	**52.9**	71.7	70.5	71.6	75.4	43.9	47.2
RoBERTa$_{\text{LARGE}}$ fine-tune + inlabel + DCA + mask	-	**72.8**	**73.9**	**72.2**	**76.0**	**44.7**	**49.4**

Table 2: Main results. **mBERT**$_{\text{base}}$ denotes the multilingual BERT (Xu et al., 2019). "+lstm" denotes adding an LSTM layer after the text encoder. "+inlabel", "DCA" and "mask" denote the "+inlabel" style of edge types, the discrete cross attention and the suffix masking technique presented in last section.

teresting discovery is that adding an LSTM layer between text encoder and cross attention leads the decreasing of SF_1 score. Thus, we remove the LSTM layer in our final submitted models. In addition, we can see that the **"+inlabel"** style definition of edge types is indeed helpful in this task. Furthermore, the presented discrete cross attention and suffix masking technique significantly improve the performance of our model.

The results prove the effectiveness of fine-tuning **RoBERTa** in this task. As shown in the Table 2, methods with **RoBERTa**$_{\text{LARGE}}$ as the text encoder on six datasets achieve the best performance. The best SF_1 scores on **MPQA**, **OpeNER**$_{\text{EN}}$ and **DS**$_{\text{Unis}}$ are 44.7, 76.0, and 49.4, where *roberta-large-en-cased* is used in the model. For **MultiB**$_{\text{CA}}$, **MultiB**$_{\text{EU}}$ and **OpeNER**$_{\text{ES}}$, *xlm-roberta-large* (Conneau et al., 2019) outperforms **BERT**$_{\text{LARGE}}$ and we have 72.8 on **MultiB**$_{\text{CA}}$, 73.9 on **MultiB**$_{\text{EU}}$ and 72.2 on **OpeNER**$_{\text{ES}}$. The results demonstrate that a pre-trained language model with more parameters and trained on the larger corpus performs very well in downstream tasks as feature extractor. For **NoReC**$_{\text{Fine}}$, we use *nb-bert-large* (Kummervold et al., 2021) and the SF_1 score is 52.9. We do not get results on **NoReC**$_{\text{Fine}}$ in the last line of the table because monolingual **RoBERTa**$_{\text{LARGE}}$ for Norwegian is not available at the time of our experiments. We guess that the model on **NoReC**$_{\text{Fine}}$ using **RoBERTa**$_{\text{LARGE}}$ will achieve a better result.

There are some failed attempt during the period of competition, too. We tried to enrich the contextual features of a sentence with word embedding, POS tag embedding, lemma embedding by using tools like SpaCy (Honnibal et al., 2020), Stanza (Qi et al., 2020) and UDPipe (Straka and Straková, 2017). But, we find that it is not superior to directly fine-tuning the pre-trained language model. We also tried to pre-train the RoBERTa-large on a Norwegian corpus, then the SF_1 score continuously grows with the increasing of training steps. However, due to the limitation of computation resources and time, we only trained the model with 2M steps. The final version does not outperform that using **BERT**$_{\text{LARGE}}$ because of the inadequate training.

5 Conclusion

In this paper, we have presented the implementation of the ZHIXIAOBAO system submitted to the SemEval-2022 Task 10. We propose an enhanced dependency parsing model for sentiment graph analysis. We leverage the fine-tuning technique of pre-trained language models, **BERT**$_{\text{LARGE}}$ and **RoBERTa**$_{\text{LARGE}}$ to increase the ability of model generalization. Furthermore, we present the discrete cross attention and suffix masking technique to achieve a significant performance improvement. Our model ranked **1st** out of 32 participating teams on the monolingual subtask with the highest SF_1 score on 4 datasets.

6 Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments.

References

Rodrigo Agerri, Montse Cuadros, Seán Gaines, and German Rigau. 2013. Opener: Open polarity enhanced named entity recognition. *Procesamiento Del Lenguaje Natural*, 51:215–218.

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. *arXiv preprint arXiv:2105.14504*.

Jeremy Barnes, Patrik Lambert, and Toni Badia. 2018. Multibooked: A corpus of basque and catalan hotel reviews annotated for aspect-level sentiment classification. *arXiv: Computation and Language*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

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

Per E Kummervold, Javier de la Rosa, Freddy Wetjen, and Svein Arne Brygfjeld. 2021. Operationalizing a national digital library: The case for a norwegian transformer model.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019a. Roberta: A robustly optimized bert pretraining approach.

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

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2019. A fine-grained sentiment dataset for norwegian. *arXiv preprint arXiv:1911.12722*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Milan Straka and Jana Straková. 2017. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis.

Hitachi at SemEval-2022 Task 10: Comparing Graph- and Seq2Seq-based Models Highlights Difficulty in Structured Sentiment Analysis

Gaku Morio, Hiroaki Ozaki, Atsuki Yamaguchi, and **Yasuhiro Sogawa**
Research and Development Group, Hitachi, Ltd.
Kokubunji, Tokyo, Japan
`{gaku.morio.vn, hiroaki.ozaki.yu, atsuki.yamaguchi.xn, yasuhiro.sogawa.tp}@hitachi.com`

Abstract

This paper describes our participation in SemEval-2022 Task 10, a structured sentiment analysis. In this task, we have to parse opinions considering both structure- and context-dependent subjective aspects, which is different from typical dependency parsing. Some of the major parser types have recently been used for semantic and syntactic parsing, while it is still unknown which type can capture structured sentiments well due to their subjective aspects. To this end, we compared two different types of state-of-the-art parser, namely graph-based and seq2seq-based. Our in-depth analyses suggest that, even though graph-based parser generally outperforms the seq2seq-based one, with strong pre-trained language models both parsers can essentially output acceptable and reasonable predictions. The analyses highlight that the difficulty derived from subjective aspects in structured sentiment analysis remains an essential challenge.

1 Introduction

SemEval-2022 Task 10 (Barnes et al., 2022) aims at extracting structured sentiment from a given sentence. Different from other sentiment analysis tasks, structured sentiment analysis is formulated as an information extraction problem with at least three elements, namely a holder, a target and a sentiment expression. The shared task has two subtasks. In Subtask 1 (monolingual), we evaluate the performance on seven monolingual corpora, i.e., `MPQA` (Wiebe et al., 2005), `OpeNER`, `OpeNER_es` (Agerri et al., 2013), `DSu` (Toprak et al., 2010), `MultiB_ca`, `MultiB_eu` (Barnes et al., 2018) and `NoReC` (Øvrelid et al., 2020). In Subtask 2 (crosslingual), we evaluate the zero-shot prediction performance on three non-English corpora, i.e., `OpeNER_es`, `MultiB_ca` and `MultiB_eu`, generated by a model trained with an English corpus. One significant difference between syntactic parsing and structured sentiment analysis is that

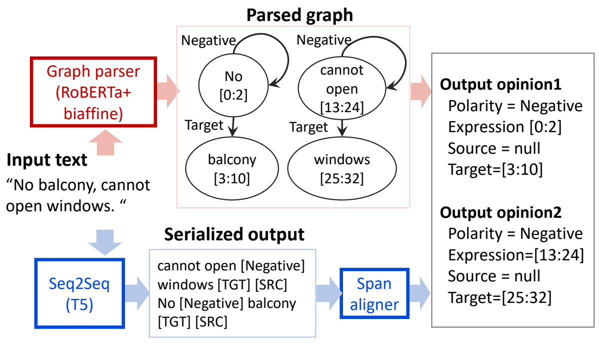


Figure 1: Overview of the two parsers: (Top) Graph and (Bottom) Seq2Seq.

the former is based on certain well-defined rules, whereas the latter forms an information structure on the basis of subjective opinions. Given the nature of the sentiment structure, Barnes et al. (2021) formulated the problem as a dependency parsing task.

Our motivation here is to throw light upon how subjective opinions affect parsing performance and how well recent strong parsing models can capture them. To this end, we compare two types of parsing model: graph- and generation-based[1] models. Though these parsers have shown competitive performance under various conditions (Oepen et al., 2020; Ozaki et al., 2020; Samuel and Straka, 2020), each of them has its own advantages and disadvantages. Graph-based parsers (McDonald et al., 2005) directly model token-to-token relations; therefore, they are suitable for modeling structured sentiment with surface anchors. On the other hand, generation-based parsers output a sequence to reconstruct the graph structure of the target meaning representation. Recent advances in deep neural networks have allowed us to generate a serialized graph directly from an input sentence by using pre-trained generation models (Ozaki et al.,

[1]We regard a transition-based parser (Dyer et al., 2015) to be a kind of seq2seq-based parser, as it generates an action sequence based on its states.

2020; Procopio et al., 2021). Specifically, we focus on state-of-the-art seq2seq-based models like T5 (Raffel et al., 2020). Because pre-trained generation models (e.g. text summarization models) have been trained to generate a summarized statement from a given input sentence, it may be easier for the generation-based parsers to model semantic relations including subjective opinions than the graph-based ones.

In this paper, we briefly introduce our approaches together with a comparison of the two different parsers: (i) **Graph**: We use a model that jointly solves two different problems, namely span identification, and relation extraction of spans. (ii) **Seq2Seq**: We design a serialization for sentiment structures generated by our seq2seq-based parser. We simply use pre-trained text generation models and fine-tune them with the serialized sentiment structures.

Experiments showed that Graph and Seq2Seq achieved reasonable levels of performance on both subtasks. Our submitted systems are based on Graph and ranked third in both the monolingual and cross-lingual tasks. We further conducted in-depth analyses for the two parsers. Our findings are twofold:

1. Multilingual performance tends to depend more on the type of pre-trained model used than on the model architecture, while Graph outperformed Seq2Seq in non-English corpora.

2. Graph is somewhat recall-focused, whereas Seq2Seq is more precision-focused in specific corpora.

The first finding suggests that Graph with a strong pre-trained model has an advantage for multilingual training when compared to Seq2Seq. Given that mT5 (Xue et al., 2021) is not trained on any supervised tasks such as translation or summarization, it may be difficult for multilingual Seq2Seq to generate (and possibly copy) sentiment tokens from the input text. Other considerations such as the decoder architecture and training time of Seq2Seq seem to favor Graph.

Regarding the second finding, we found that Graph performs well on complex structured opinions. However, on the other side of the coin, it suggests that Graph sometimes causes over-detection (though these over detected opinions may be acceptable to humans). We found that this happens particularly for `MPQA` polar expressions. This could be due to the nature of `MPQA`, which usually includes context-dependent expressions.

As a result, we argue that it is difficult to decide which type of parser is better because: (i) the decision criteria rely on how we define the sentiment structure, e.g., structured sentiments in some corpora are semantically complex and context-dependent, while those in other corpora are not, making the corpus less context-dependent. (2) Whether to use Graph or Seq2Seq depends on whether we want to cover as much of the structured sentiments as possible or whether we want to emphasize precision. (3) In regard to metric-dependent choices, Seq2Seq is at a disadvantage in the shared task because the metric requires anchoring to the surface of the input text. We also have to decide which parser to use from various other perspectives, such as (4) whether or not we are targeting English, and (5) whether the training speed is important. These considerations make structured sentiment analysis challenging.[2]

2 Models

This section explains the Graph and Seq2Seq methods used in this work.

2.1 Graph model

We formulate the problem as a joint task of span identification and relation extraction. This approach can be classified as a graph-based approach (Dozat et al., 2017; Falenska et al., 2020), which is known to perform well in fields such as syntactic dependency parsing. Although there are various approaches to this problem, we simply use the architecture of Morio et al. (2022). The architecture generates BIO tags to predict spans using pre-trained language models such as Longformer (Beltagy et al., 2020). The span representation of each predicted span is generated by average pooling of the predicted span and subsequently fed into biaffine classifiers (Dozat and Manning, 2017) to predict relations.

Because the architecture was originally designed to predict argument structures (Lawrence and Reed, 2019), so-called argument mining, it needs a little tweaking. We thus designed a dedicated encoding for structured sentiment analysis. As shown in Figure 1 (Top), the opinions for the input text can

[2]We plan to release our code at `https://github.com/hitachi-nlp/graph_parser`

be represented as a graph. We represent a representative polarity expression as a root node, and its polarity as a self-loop (e.g., a *no* node and its negative self-loop). We represent the other polarity expressions, target and source via child nodes linked to the polarity nodes (e.g., *balcony* is a child node for the *no* node). Overlapping spans are divided up using the start and end indexes of the overlap, and each span can have multiple labels (e.g., *Positive_Source* is a combined label for positive and source spans). Although this graph encoding cannot fully represent structured sentiments[3], we confirmed that the data reconstructed from the graph encoding achieves about 99% F-score (so there is practically no problem.) We convert all the given datasets using this graph encoding, and given a text input, the model is trained to parse the graph.

2.2 Seq2Seq model

Seq2Seq Generation Here, we formulate the problem as a summarization task to output serialized tuples of structured sentiment. We preferably utilize pre-trained summarization language models, such as T5 (Raffel et al., 2020), and fine-tune them with the serialized tuples.

Serialization system A seq2seq model outputs serialized tuples of structured sentiment regardless of their position on the surface (see Figure 1). For example, when an input text is “No balcony, cannot open windows”, we have two tuples of structured sentiment; each of them is serialized as follows:

```
No[NEG] balcony[TGT] [SRC]
cannot open[NEG] windows[TGT] [SRC]
```

We serialize a tuple into a polar expression, a target, and a holder order. The polar expression ends with `[NEG]`, `[NEU]` or `[POS]` tokens according to its polarity. The target and holder expressions end with `[TGT]` and `[SRC]`, respectively. If multiple spans are in a tuple, we concatenate them with a special separator (`[SEP]`) token. Since each expression is marked with these special tokens, we can simply concatenate all tuples without confusing them with each other.

Reconstruction from serialization Although our serialization preserves semantically sufficient information on sentiment structures, there is a piece of missing information, i.e., anchors. To reconstruct the anchor information, we utilize a word aligner based on a pre-trained language model: SimAlign (Jalili Sabet et al., 2020). Because SimAlign provides a zero-shot alignment model, we utilize it for obtaining the alignment between the input text and its serialized sentiment structure. For ease of explanation, we define *span* $s = [t_i, t_{i+1}, \ldots, t_{i+j}]$ as a single phrase appearing as a holder, a target, or a polar expression, where t is a generated token. First, we pick a single opinion; then, we extract all tokens in the opinion and form a token sequence $[t_1, \ldots, t_n]$. Second, we calculate the token alignments $[a_1, \ldots, a_n]$ between the input text and the token sequence to point out the corresponding tokens in the input, where $a > 0$ and $a \in \mathbb{N}$. Lastly, we recover the span $s = [t_i, \ldots, t_{i+j}]$ location by $[\min(a_i, \ldots, a_{i+j}), \max(a_i, \ldots, a_{i+j})]$ [4].

Furthermore, we add a heuristic procedure to improve the reconstruction accuracy. When an expression extracted from a reconstructed anchor is different from its original generated expression, we apply a greedy search to find the part where the original expression appears *as is* in the input text. If we find the part, we use an anchor that points to the part instead of the reconstructed anchor. With this heuristic, we achieved an F-score of around 97% between gold and reconstructed opinions from the serialized outputs[5].

3 Experiments

3.1 Experimental setup

Implementation details We implemented Seq2Seq and Graph with PyTorch (Paszke et al., 2019) and the Huggingface Transformers library (Wolf et al., 2020). All models were trained with a fixed number of steps (about 10,000 steps). We used a learning rate of 2e-5 for Graph and 5e-5 for Seq2Seq, with a warmup (Howard and Ruder, 2018) ratio of 0.1. The batch size was set to 16 for Graph and 32 for Seq2Seq. We set the beam width to 5 for Seq2Seq. We did not conduct hyperparameter tuning or model selection and did not use any development data during training and validation.

Pre-trained models To fully utilize the representative power of Graph and Seq2Seq, we used

[3]Our graph encoding cannot distinguish a case where one polar expression forms a single opinion with multiple targets from a case where a polar expression forms multiple opinions with a single target.

[4]Because actual anchors are character-level, we need to convert the spans from the token level to the character level.

[5]Without this heuristic, the F-score is about 91%.

Dev		DSu	OpeNER	MPQA	OpeNER_es	MultiB_eu	MultiB_ca	NoReC
		en	en	en	es	eu	ca	no
Graph	RoBERTa-large	**38.6**	**72.3**	**42.8**				
	InfoXLM-large				**71.1**	**64.7**	**71.0**	**50.7**
Seq2Seq	T5-large	38.0	69.4	42.3				
	mT5-large				66.0	61.4	67.9	48.8
Test								
Graph	RoBERTa-large	42.2	75.0	39.3				
	+ensemble (submitted)	**46.3**	**75.6**	40.2				
	InfoXLM-large				71.7	70.5	69.8	51.2
	+ensemble (submitted)				**73.2**	**71.5**	**70.9**	**53.3**
Seq2Seq	T5-large	40.5	67.1	**40.9**				
	mT5-large				65.6	66.2	65.5	48.0
Best team		49.4	76.0	44.7	72.2	73.9	72.8	52.9

Table 1: Evaluation results of Subtask 1 (monolingual) in Sentiment Graph F1 (SF_1) for the development and test sets. Graph and Seq2Seq represent graph-based and seq2seq-based parsers, respectively. We submitted the InfoXLM-large+ensemble model in the evaluation phase. Note that en=English, es=Spanish, eu=Basque, ca=Catalan, and no=Norwegian.

Dev		OpeNER_es	MultiB_eu	MultiB_ca
		en→es	en→eu	en→ca
Graph	InfoXLM-large	**62.8**	**46.2**	**62.3**
Seq2Seq	mT5-large	56.9	41.3	53.5
Test				
Graph	InfoXLM-large	61.9	51.6	60.1
	+ensemble (submitted)	**62.8**	**52.7**	**60.7**
Seq2Seq	mT5-large	57.4	44.7	53.5
Best team		64.4	63.2	64.3

Table 2: Evaluation results of Subtask 2 (crosslingual zero-shot) in SF_1. We submitted the InfoXLM-large+ensemble version in the evaluation phase.

pre-trained language models based on Transformer (Vaswani et al., 2017). For Graph, we used RoBERTa-large (Liu et al., 2019) in Subtask 1 (monolingual) and InfoXLM-large (Chi et al., 2021) in Subtask 2 (cross-lingual). RoBERTa is a well-tuned model based on BERT (Devlin et al., 2019), and it has shown state-of-the-art performance in various classification tasks. InfoXLM is a recently proposed model, which is pre-trained with contrastive learning. For Seq2Seq, we used T5-large (Raffel et al., 2020) in Subtask 1 and mT5-large (Xue et al., 2021) in Subtask 2. T5 uses a unified text-to-text framework to deal with various text-based tasks.

Submitted models In our preliminary experiments, we found that the development scores of the monolingual Graph models were slightly better than those of the Seq2Seq ones (the reasons will be discussed later). Thus, we only used Graph models for our submission. However, to discuss Graph and Seq2Seq in detail, we show the results of both Graph and Seq2Seq below.

3.2 Main results

Table 1 shows the overall results of Subtask 1 (i.e., monolingual), including the scores for the development and test data in Sentiment Graph F1 (SF_1) (Barnes et al., 2021). We tried three different seeds for each model to minimize the effects of random seeds and averaged the scores. For the ensemble methods, the scores are those from the ensemble of models with the three seeds. For reference, we also include the results of the best-performing team for the test data. Overall, Graph mostly outperformed Seq2Seq with significant differences observed in non-English corpora, such as OpeNER_es and MultiB_eu. This suggests that Graph has an advantage for multilingual training when compared to Seq2Seq. That is, while T5 is trained on summarization and its related tasks, mT5 is not. This difference might cause a disadvantage

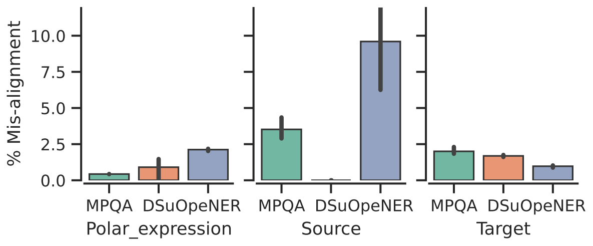


Figure 2: Misalignment ratio of Seq2Seq, where misalignment means Seq2Seq predicted correct surface tokens, but the aligned spans for the original text were incorrect.

wherein Seq2Seq generates (and possibly copies) sentiment tokens from the input text.

Although not surprising, Table 1 indicates that the ensemble models of Graph outperform the non-ensemble ones. The submitted ensemble models have comparable SF_1 to those of the best team on the test set of some corpora.

Subtask 2 In this subtask, we used `OpeNER` English for training and conducted zero-shot prediction for `OpeNER_es`, `MultiB_eu`, and `MultiB_ca`. Table 2 shows SF_1 of Subtask 2 (i.e., cross-lingual zero-shot). Overall, it seems that Graph is still a better choice than Seq2Seq. There is a significant difference in `MultiB_eu`, i.e., Seq2Seq (44.7) and Graph (51.6). This difference may be due to the pre-trained language models (i.e., InfoXLM vs. mT5) and the misalignment problem of Seq2Seq. On the other hand, Graph and Seq2Seq performed poorly compared with the top-performing team, so neither model seems suitable for the zero-shot setting.

3.3 Analysis and Discussion

This section compares Seq2Seq and Graph and points out that one is not superior to the other. Our argument is supported by an analysis of the alignment errors of Seq2Seq and the structural/semantic properties of the parsers. To simplify the discussion, we focus only on non-ensemble and monolingual models for the English corpora (i.e., `MPQA`, `DSu` and `OpeNER`).

3.3.1 Does Graph really outperform Seq2Seq?

Alignment error in Seq2Seq A major drawback of Seq2Seq is the alignment error caused by the aligner. That is, Seq2Seq can produce the correct surface tokens of the polar expression, source or target, but the aligner may align incorrect spans

Figure 2 shows the misalignment ratio (i.e., the ratio of predicted elements where the surface tokens generated by Seq2Seq are correct, but the spans generated by the aligner are incorrect). There is a certain amount of alignment error in `MPQA`, `DSu` and `OpeNER`. We can see that the ratio of `OpeNER` is larger than those of the others. We explain this in Table 3. In case #1, a polar expression *chic!* was correctly predicted, but the aligner did not include the exclamation mark. In #2, *minutes from numerous...* was correctly predicted, but the beginning word *minutes* was unfortunately not included in the span. We found that the large misalignment ratio of the source in `OpeNER` was caused by pronoun words, as illustrated in #3, where our system could not resolve which *I* (i.e., the former or the latter in the two *I* tokens) to align. Case #4 shows a similar phenomenon for the term *hate*.

If we had remedied some of the misalignments, Seq2Seq would have produced 41.8 SF_1 and 42.6 SF_1 on the test data of `DSu` and `MPQA`, respectively. These results are comparable or better than those of the Graph models shown in Table 1; thus, we can not simply conclude that Graph is better than Seq2Seq. Moreover, Seq2Seq might be the best choice for an evaluation metric that does not emphasize anchoring spans of the input sentence (which may be enough for practical purposes). On the other hand, it seems that Seq2Seq for `OpeNER` still has a significant performance gap against Graph, as shown in Table 1. We focus on this aspect below.

3.3.2 How do Graph and Seq2Seq differ from each other?

Here, we discuss the differences between Graph and Seq2Seq on the basis of structural and semantic complexity. We also discuss the limitations of the parsers.

Complexity of sentiment structure We suppose that the number of opinions in an input text can be used as a proxy metric showing the complexity of the sentiment structure. Figure 3 shows the relationship between the number of opinions in an input sentence and SF_1. Overall, the two parsers exhibit similar trends; that is, the more opinion numbers there are, the harder the prediction becomes. However, the performance of Graph seems to be less dependent on the number of opinions, while Seq2Seq generally exhibits a negative trend, especially for `OpeNER`. These results suggest that Graph is a good choice for handling complicated

#	Corpus	Text	Gold span (and its text)	Generated surface by Seq2Seq	Mis-aligned span (and its text)
1	`OpeNER`	hotel chic !	[6:12] (chic !)	chic!	[6:10] (chic)
2	`OpeNER`	It is minutes from numerous restaurants , bars , etc. and centrally located between the Prado and the Palace · · great for walking .	[6:53] (minutes from numerous restaurants, bars, etc.)	minutes from numerous restaurants, bars, etc.	[14:53] (from numerous restaurants , bars , etc.)
3	`OpeNER`	I was forced to stay in this area due to my business reason , but oterwise I would not suggest to come here to spend your holidys .	[75:76] (I)	I	[0:1] (I)
4	`MPQA`	" We do n't hate the sinner , " he says , " but we hate the sin . "	[51:55] (hate)	hate	[12:16] (hate)

Table 3: Case study of misalignments where Seq2Seq produced correct surface tokens but the reconstruction system aligned incorrect spans.

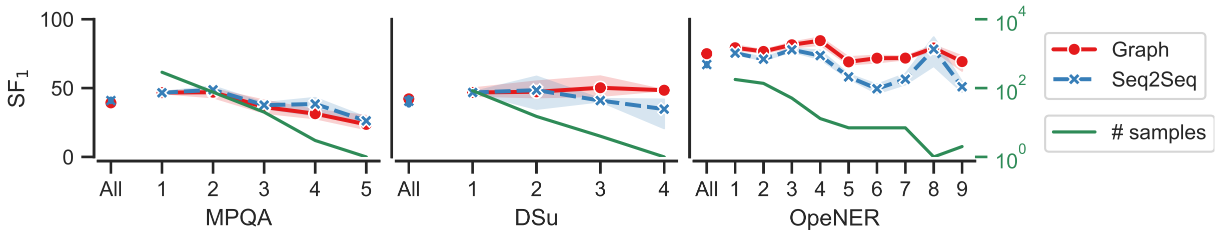


Figure 3: Relationship between the number of opinions included in a sentence (X-axis) and SF$_1$ (Y-axis). The green line shows the total number of samples in the evaluation set; it is evident that the larger the number of opinions is, the fewer the number of samples is. Note that *All* indicates the evaluation for full sentences, for reference.

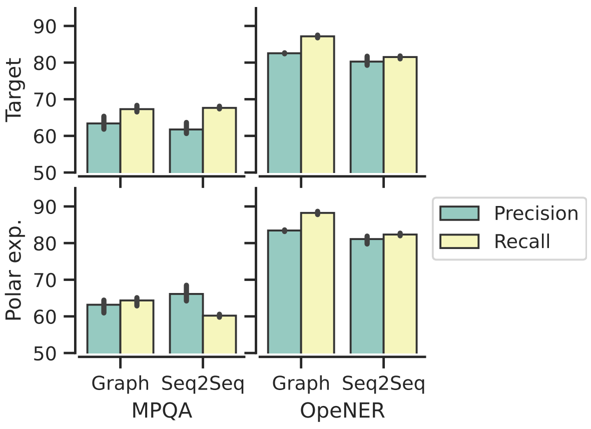


Figure 4: Precision and recall of Graph and Seq2Seq for target (top) and polar expression (bottom) outputs.

structures.

However, Graph has another disadvantage: over-detection. Figure 4 shows the precision and recall values for target and polar expression in `MPQA` and `OpeNER`. As shown, there is a possible trend that Graph is more recall-focused, and Seq2Seq is more precision-focused. In other words, Graph may over-detect opinion fragments for specific corpora. Case #1 in Table 4 shows an example of over-detection. The over-detected phrases are a *means* of "threatened" and may not be a target.

However, we find that, even for complex structures with multiple opinions, both Graph and Seq2Seq produce reasonable predictions. In case #1 in Table 4, the *means* of "threatened" is semantically a target; thus, this would be an acceptable case. Case #2 shows a long polar expression which does not appear in the gold standard, i.e., *let alone the 4 stars*, while the prediction seems to be correct in a sense. Through these observations, both Graph and Seq2Seq seem to try to output plausible outputs.

Semantic complexity Here, we begin by focusing on the difference between corpora (i.e., `MPQA` and `OpeNER`), because it is difficult to evaluate the semantic complexity directly. `MPQA` is annotated on the basis of the private state frame and distinguishes subjective information from material presented as fact (Wiebe et al., 2005). This makes `MPQA` semantically complex and context-dependent. On the other hand, `OpeNER` project originally focuses on lexicon creation (Agerri et al., 2013), making the corpus probably less context-dependent.

Again, let us investigate the precision and recall in Figure 4. A significant score gap between precision and recall in the figure can be found in

#	Corpus	Text	Method	Polarity	Polar exp.	Source	Target
1	MPQA	But after the Chinese side released all the US crew members , major US political figures changed their stance immediately and threatened to use human rights , trade , and Olympics hosting issues to " retaliate " against China .	Gold	Negative	threatened	major US political figures	China
			Graph	Negative	threatened	major US political figures	1. China, 2. trade , and Olympics hosting
			Seq2Seq	Negative	threatened	major US political figures	China
2	OpeNER	I would never ever come back to this hotel even if they paid me. simply it 's not worth the money , let alone the 4 stars .	Gold	Negative	would never ever come back	I	this hotel
				Negative	not worth		the money
			Graph	Negative	would never ever come back	I	this hotel
				Negative	not worth		the money
			Seq2Seq	Negative	never ever come back	I	this hotel
				Negative	not worth	I	the money
				Negative	let alone the 4 stars		it
3	MPQA	AOL would never have existed if it had been founded here , I am sure , since its employees would have been mocked into obscurity by the digerati .	Gold	Negative	mocked into obscurity	digerati	its employees
			Graph	Negative	mocked	the digerati	its employees
			Seq2Seq	⊥	⊥	⊥	⊥
4	MPQA	In the complaint , Hobeika had not yet been called by name .	Gold	Negative	the complaint		
			Graph	Negative	complaint		
			Seq2Seq	⊥	⊥		
5	DSu	I had a programming class with no lectures which is always fun for beginners to try to learn concepts without any sort of interactivity .	Gold	Negative	1.always, 2.fun for beginners	everyone	no lectures
			Graph	⊥	⊥	⊥	⊥
			Seq2Seq	⊥	⊥	⊥	⊥

Table 4: Case study of outputs by graph-based and seq2seq-based models. Magenta colored text indicates incorrect outputs. For visibility, we have omitted some of the outputs. ⊥ represents a false-negative prediction.

the polar expression of MPQA; i.e., for Seq2Seq, the polar expression's precision of MPQA is quite higher than its recall. Interestingly, the gap was not so large in the polar expression of OpeNER. We presume that this is due the semantic complexity (or context-dependent nature) of MPQA. That is, since Seq2Seq always refers to the context of the input side when generating an output, the output could be more context-dependent, and as a result, Seq2Seq may not output less-confident opinions.

Since it is difficult to test the hypothesis that Seq2Seq may not output less-confident opinions in a statistical manner, we present several case studies. Cases #3 and #4 in Table 4 show errors where Graph predicted correct or incorrect opinions but Seq2Seq did not. In #3, the term *mocked* is a negative word *as is*, and can be predicted only with a lexical perspective. However, the polar expression is located in a fictional speculation in insubstantial text, which may have confused Seq2Seq. In #4, Graph predicted *complaint* as a negative polar expression, since the complaint could be a negative lexicon *as is*, while Seq2Seq did not output any opinions. Considering the context in the entire article, it might be difficult to for Seq2Seq to determine if the complaint is an opinion. In this way, Seq2Seq might enable more context-aware predictions, but may not be suitable for structured sentiment analysis based on lexicons. Graph seems to be good at handling lexicons and context-independent phrases.

Limitation of both parsers Another semantically complex case illustrates an interesting error that neither Graph nor Seq2Seq could parse. Case #5 in Table 4 shows an irony expression that might illustrate a limitation of pre-trained models.

Which is superior? As discussed above, each model has its own advantages depending on the perspective. However, we cannot decide which is better, because the decision criteria rely on how we define the sentiment structure. MPQA defines structured sentiment on the basis of private state frames, while OpeNER focuses more on lexical

↓ Sec. per **step**		DSu	OpeNER	MPQA
		en	en	en
Graph	RoBERTa-large	**0.2**	**0.2**	**0.3**
Seq2Seq	T5-large	2.0	2.0	1.8
↓ Sec. per **epoch**				
Graph	RoBERTa-large	**34.0**	**26.9**	**92.8**
Seq2Seq	T5-large	138.6	105.3	332.1

Table 5: Training speed (in sec.) for each parser.

		SF_1
(Barnes et al., 2021)	mBERT	31.2
(Peng et al., 2021)	mBERT	31.9
PERIN NC (Samuel et al., 2022)	XLM-R-base	39.3
PERIN LE (Samuel et al., 2022)	XLM-R-base	40.4
PERIN OT (Samuel et al., 2022)	XLM-R-base	41.6
Graph (ours)	XLM-R-base	**44.8**

Table 6: Comparison with state-of-the-art methods for `NoReC` test data. Note that NC, LE, and OT mean the node-centric, labeled edge, and opinion-tuple variants of PERIN.

semantics. This would make a difference with syntactic parsing, which is based on a certain type of well-defined grammar, and would cause the parsing performance to be lower than that of syntactic parsers. To cover the variety of definitions of structured sentiment, an abstraction of the sentiment structure might be needed, as is done in abstract meaning representation (AMR; Banarescu et al. 2013).

3.3.3 Energy efficiency approximated by training time

Finally, we discuss the energy efficiency of Graph and Seq2Seq as an estimate of their financial and environmental impact (Strubell et al., 2019). We evaluated the approximated energy consumption in terms of the training time on NVIDIA V100 GPUs. Table 5 shows step-normalized and epoch-normalized training speed in seconds, given that a different batch size was used for Graph and Seq2Seq. As shown, Graph is usually faster than Seq2Seq on all English corpora. We suppose this is because Seq2Seq has a decoder part, which has a computational cost. The results suggest that Graph is preferable in terms of energy efficiency.

4 Related work and state-of-the-art

Sentiment analysis, such as aspect-based sentiment analysis (Chen and Qian, 2020), is a popular research area in natural language processing. Most recently, we have seen attention focusing on parsing full representations of sentiment from text, i.e., structured sentiment analysis (Barnes et al., 2021), as we have tackled in this study. Most methods (Barnes et al., 2021; Peng et al., 2021) for this task are motivated by graph-based parsers that can parse the structured sentiment using a technique similar to dependency parsing. On the other hand, the state-of-the-art graph-based parser, PERIN (Samuel et al., 2022), utilizes the idea of meaning representation parsing (Oepen et al., 2020; Samuel and Straka, 2020), which remedies the lossy dependency graphs of the previous work (Barnes et al., 2021). Our graph-based method (i.e., Graph) is one such study. The differences between Graph and the related literature are in the graph encoding method and model architecture.

To directly compare our Graph with the state-of-the-art methods, we trained Graph with XLM-RoBERTa-base (XLM-R; Conneau et al. (2020)). We tried three different seeds and averaged the scores. Because the corpora provided in the shared task were slightly modified from those used in the related studies, it is difficult to make a direct comparison. Thus, we evaluated only on the unaffected `NoReC`. Table 6 suggests that our Graph outperforms state-of-the-art baselines; however, we cannot conclude that our method is state-of-the-art based solely on an evaluation on `NoReC`. We hope that more extensive studies will clarify this situation.

On the other hand, an alternative to the traditional graph-based methods, we proposed a generation-based method (i.e., Seq2Seq) that showed promising results. Generation-based methods have been recently utilized in meaning representation parsing (Ozaki et al., 2020; Procopio et al., 2021); this framework offers more research options in terms of architecture and graph encodings for structured sentiment analysis. Our study can be positioned within this framework.

5 Conclusion

This paper showed two different parsers (i.e., graph-based and seq2seq-based parsers) for SemEval-2022 Task 10, structured sentiment analysis. The parsers were compared in various aspects such as complexity in structure and semantics. Experiments and analyses showed that both parsers output reasonable predictions, but that it is hard to decide which is better. This could be because the decision criteria rely on how the sentiment structure is defined. This makes structured sentiment analysis challenging. To deal with this difficulty, it may be helpful to apply an abstract representation of structured sentiment.

Acknowledgements

The computational resources of the AI Bridging Cloud Infrastructure (ABCI) provided by the National Institute of Advanced Industrial Science and Technology (AIST) were used. We would like to thank Dr. Masaaki Shimizu for maintenance and management of the large computational resources.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Laura Ana Maria Oberländer, Enrica Troiano, Andrey Kutuzov, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, and Erik Velldal. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

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

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694, Online. Association for Computational Linguistics.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, Beijing, China. Association for Computational Linguistics.

Agnieszka Falenska, Anders Björkelund, and Jonas Kuhn. 2020. Integrating graph-based and transition-based dependency parsers in the deep contextualized era. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 25–39, Online. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, and Kohsuke Yanai. 2022. End-to-end argument mining with cross-corpora multi-task learning. *Transactions of the Association for Computational Linguistics*, pages accepted, to appear.

Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Tim O'Gorman, Nianwen Xue, and Daniel Zeman. 2020. MRP 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 1–22, Online. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Hiroaki Ozaki, Gaku Morio, Yuta Koreeda, Terufumi Morishita, and Toshinori Miyoshi. 2020. Hitachi at MRP 2020: Text-to-graph-notation transducer. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 40–52, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Letian Peng, Zuchao Li, and Hai Zhao. 2021. Sparse fuzzy attention for structured sentiment analysis.

Luigi Procopio, Rocco Tripodi, and Roberto Navigli. 2021. SGL: Speaking the graph languages of semantic parsing via multilingual translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 325–337, Online. Association for Computational Linguistics.

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

David Samuel, Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2022. Direct parsing to sentiment graphs. In *arXiv (to appear in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022))*.

David Samuel and Milan Straka. 2020. ÚFAL at MRP 2020: Permutation-invariant semantic parsing in PERIN. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 53–64, Online. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

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

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

UFRGSent at SemEval-2022 Task 10: Structured Sentiment Analysis using a Question Answering Model

Lucas Rafael Costella Pessutto
Institute of Informatics
UFRGS – Brazil
lrcpessutto@inf.ufrgs.br

Viviane P. Moreira
Institute of Informatics
UFRGS – Brazil
viviane@inf.ufrgs.br

Abstract

This paper describes the system submitted by our team (UFRGSent) to SemEval-2022 Task 10: Structured Sentiment Analysis. We propose a multilingual approach that relies on a Question Answering model to find tuples consisting of holder, target, and opinion expression. The approach starts from general questions and uses the extracted tuple elements to find the remaining components. Finally, we employ an aspect sentiment classification model to classify the polarity of the entire tuple. Despite our method being in a mid-rank position in the SemEval competition, we show that the question-answering approach can achieve good coverage retrieving sentiment tuples, allowing room for improvements in the technique.

1 Introduction

Opinions abound on the Internet nowadays. They are a valuable source of information since people often rely on them for making purchases. Companies can also benefit from this vast amount of opinions, as they do not need to conduct opinion polls or focus groups to measure the acceptance of a particular product (Liu, 2011). The large volume of opinions available becomes hard for humans to process. This leads to the study of ways of automating the processing of opinions, in order to summarize them.

Sentiment Analysis is the field of study which aims at processing the information conveyed by unstructured texts, providing structured information that facilitates the understanding of the opinions, attitudes, or emotions towards a particular entity (Liu, 2011). Sentiment Analysis can be performed at different levels of granularity (entire review, sentence, or aspect). Aspect-Based Sentiment Analysis (ABSA) aims to identify and rate the features (or aspects) of the entity being evaluated. Typically, ABSA involves the following phases: (i) identify and extract entities in reviews; (ii) identify and extract the aspects of an entity; (iii) cluster similar aspects; and (iv) determine the polarity of the sentiment over the entities and the aspects. Most of the research in Sentiment Analysis focuses on solving only one of these phases at a time.

Task 10 in SemEval 2022 – Structured Sentiment Analysis (Barnes et al., 2022) proposes a new approach to tackle the Sentiment Analysis problem, where the elements that constitute an opinion are identified together, in a structured way through a graph. Thus, Structured Sentiment Analysis can be seen as an information extraction task since we want to find the text spans where opinions about a particular feature are expressed (Barnes et al., 2021).

In this paper, we describe UFRGSent, a multilingual approach that relies on a question answering system to find the elements of an opinion present in review texts and a fine-tuned model to classify the polarity of the sentiment tuple. Our average results ranked 20th out of 31 participating systems. Nevertheless, we believe there is room for improvement in our technique.

2 Background and Related Work

An opinion can be defined as a tuple $O = (h, t, e, p)$, where h represents the opinion holder (person who emits the opinion), t is the aspect target of the entity being reviewed, e is the opinion that is being expressed, and p is the sentiment related to the aspect expressed on the review (Liu, 2012).

While many works treat each opinion component separately, some approaches extract them all together, taking advantage of the components being interconnected. Graph neural networks (Barnes et al., 2021; Qian et al., 2021), transition-based neural models (Zhang et al., 2019), and multi-task learning (Chen and Qian, 2020) can be used to accomplish this task. There are also works applying co-extraction to find correlated opinion components, such as aspect words and corresponding polarities (Luo et al., 2019; He et al., 2019), aspect and opinion terms (Wu et al., 2020), or aspects, opinions, and polarities (Wang et al., 2017; Chen et al., 2021).

Question Answering is a challenging and well-studied problem in the Natural Language Processing field, gaining attention in the last years due to the use of pre-trained Language Models. One of the most popular subtasks of question answering is the Machine Reading Comprehension task. This task consists of, from a text piece (also known as context) and a question, finding the answer to the question in context (Zeng et al., 2020). Chen et al. (2021) proposed using a machine reading comprehension system to solve the problem of aspect sentiment triplet extraction. They use three-turn questions to extract aspects and opinions (in the first two turns) and sentiments (in the last turn), achieving state-of-the-art performance on standard Aspect-Based Sentiment Analysis (ABSA) datasets.

3 UFRGSent

3.1 Task Description

The Structured Sentiment Analysis task consists in identifying a sentiment graph from a review text r. Such graph can be seen as tuple $t = (h, t, e, p)$, composed by the the holder (h), the target (t), the opinion expression (e), and the sentiment polarity (p). Components h, t, and p are text spans over r, while $s \in \{\text{Positive}, \text{Negative}, \text{Neutral}\}$. A review r can contain none or multiple sentiment tuples.

3.2 Solution Overview

An overview of UFRGSent can be seen in Figure 1. A two-phase process was employed to identify sentiment tuples in a review text r. First, we use a pre-trained question answering model fine-tuned with opinionated texts in order to identify the spans in r that correspond to holder, targets, and opinion expressions. Next, we use another pre-trained aspect sentiment classification model fine-tuned on the training datasets, in order to predict the polarity of each extracted sentiment tuple.

The extraction of sentiment tuples from the review text was made using a question answering model. We iteratively submit three kinds of questions to the model, in order to extract candidates to sentiment tuples.

Tier 1 Questions: the following questions were the first questions posed to the model. As shown,

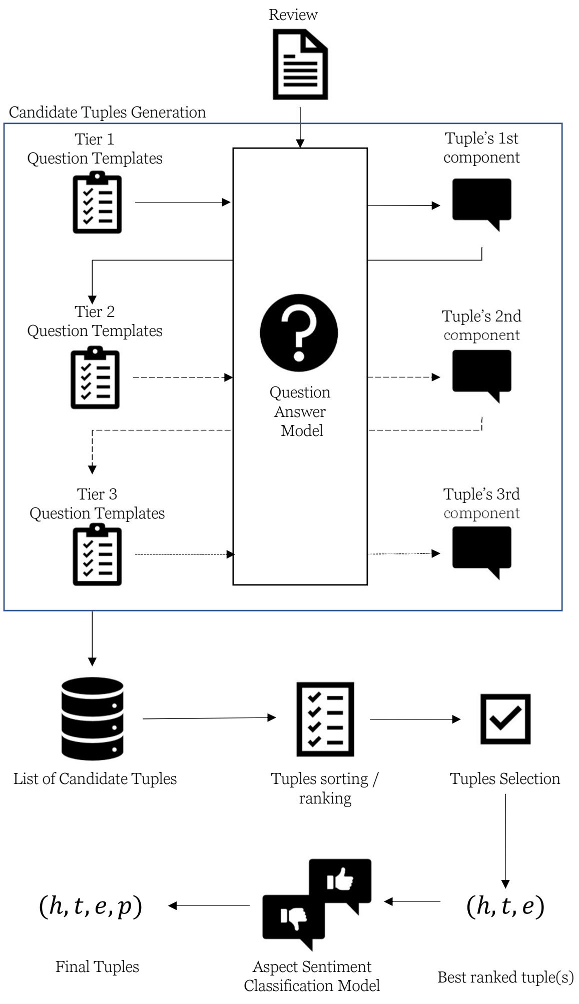


Figure 1: Overview of UFRGSent

each of the questions allows us to obtain one component of the sentiment tuple.

```
Who is the holder?
  Answer: (h, __, __)
What is the aspect expression?
  Answer: (__, t, __)
What is the opinion word?
  Answer: (__, __, e)
```

We extract the n most likely answers predicted by the model for each question. The model can also predict that the question has no answer on r. When this happens, we interpret it as the absence of sentiment in the sentence and produce the null tuple $(\_\_, \_\_, \_\_)$ as a candidate.

Tier 2 Questions: after the first extraction iteration, we obtain a list of candidate tuples containing just one component. This step aims to extract the second tuple component based on the existing one. These are the templates of questions used in this step, and the candidate tuples that were generated in this phase.

```
Who has an opinion about <target>?
  Answer: (h, t, __)
What is the feeling about <target>?
  Answer: (__, t, e)
What is <holder> opining about?
  Answer: (h, t, __)
What is the opinion expressed by
<holder>?
  Answer: (h, __, e)
What is <opinion> about?
  Answer: (__, t, e)
Who thinks <opinion>?
  Answer: (h, __, e)
```

We do not use the null tuple in this iteration. If the QA Model returns that a question has no answer in this phase, we create a tuple without the remaining components, which will not be used in the next phase.

Tier 3 Questions: finally, we use the candidate tuples obtained in the previous step, with two components of the tuple, to obtain the remaining expression. These are the templates of questions for this step.

```
How <holder> feels about <target>?
Who thinks <target> is <opinion>?
What <holder> expressed <opinion>
about?
```

There are two possible outcomes for this phase – a complete tuple (h, t, e) or a tuple with two components, for the cases in which the question produces no answer.

Candidate Ranking and Selection: At the end of the Tier 3 questions, UFRGSent produces a list of candidate tuples, containing all answers generated by the iterative procedure previously described. The next step is ranking these tuples according to some criteria. Finally, based on the final ranking, we select the top-k answers to find the subset of tuples that best represent the structured sentiment on that review. If the null tuple were selected as the best answer in this phase, we conclude that the review has no sentiment.

Aspect Sentiment Classification: After determining the first three components of the sentiment tuple, we use an aspect sentiment classification model. This model receives the review and the aspect-phrase as inputs and outputs the polarity of the aspect in the review.

4 Experimental Setup

4.1 Question Answering Model

We fine-tuned BERT multilingual (Devlin et al., 2019) for the Question Answering task using the training data provided. To convert the original data into a question-answer dataset, we employed the following technique: Each sentence becomes a context. For each sentence, we generate the questions following the templates presented in Section 3.2. If a sentence does not have any sentiment annotation, we only generate Tier-1 questions with the null answer. Otherwise, we generate the three tiers of questions for the sentences containing structured sentiment annotations. For the sentences whose tuples do not contain some component (*i.e.,* part of the sentiment graph is not in the sentence), we do not generate questions with the missing component. For example, we only include the question `Who has an opinion about <target>?` for the sentences that have the aspect annotations.

The QA model was fine-tuned for two epochs, using the script provided by HuggingFace Transformers (Wolf et al., 2020)[1]. We tested the final model over dev datasets provided in the task, obtaining an F-1 score of 74.68.

4.2 Aspect Sentiment Classification Model

For the polarity prediction task, we used LCF-BERT (Zeng et al., 2019), which is provided by PyABSA[2]. We used an existing trained model created over 14 datasets in two languages (English and Chinese) as our base model. The base model was fine-tuned for ten epochs on the training dataset. The tests over the dev datasets yielded an F1 score of 75.13% and an Accuracy of 88.54%. The model only accepts as input a review and an aspect. In the case of sentences which not contain the target component, we feed the model with the opinion expression. The holder information is not used in this task.

4.3 Ranking and Tuple Selection

Our team employed two simple heuristics to sort the candidate tuples and select the final tuples. To sort the candidate tuples, we ranked them according to the number of times that the tuple was generated by the question answering procedure. Since we make multiple questions, it is common for an answer to be generated many times.

Once the candidate tuples are sorted, we selected the most frequent answers as our final tuples. If the most frequent was the null answer, we assume

[1]https://github.com/huggingface/transformers/tree/main/examples/pytorch/question-answering

[2]https://github.com/yangheng95/PyABSA

the review has no sentiment. On the other hand, if our technique generated more than one answer, we prune the answer set by removing the occurrence of the null tuple and removing overlapping tuples, *i.e.,* the tuples that have conflicting spans over the review. For example, in the sentence "I love the food." the spans "food" and "the food" overlap. In that case, we only keep the first tuple that appears in the ranking.

4.4 Datasets

We evaluated UFRGSent using seven datasets, namely NoReC (Øvrelid et al., 2020) that contains professional reviews in Norwegian; MultiBooked_eu and MultiBooked_ca (Barnes et al., 2018) with hotel reviews in Basque and Catalan, respectively; OpeNER_en and OpeNER_es (Agerri et al., 2013) that contain hotel reviews in English and Spanish, respectively; MPQA (Wiebe et al., 2005) a dataset of news wires in English; and Darmstadt Service Reviews (Toprak et al., 2010) with English reviews from online universities. Table 1 shows statistics of the datasets.

Dataset	Lang	# sent	# h	# t	# e
NoReC	NO	11,437	1,128	8,923	11,115
MultiBooked_eu	EU	1,521	296	1,775	2,328
MultiBooked_ca	CA	1,678	235	2,336	2,756
OpeNER_es	ES	2,057	255	3,980	4,388
OpeNER_en	EN	2,494	413	3,850	4,150
MPQA	EN	10,048	2,279	2,452	2,814
Darmstadt_unis	EN	2,803	86	1,119	1,119

Table 1: Statistics of Datasets (obtained from `https://github.com/jerbarnes/semeval22_structured_sentiment`).

4.5 Evaluation Metric

The evaluation metric used to assess the quality of the participating systems was the Sentiment Graph F_1 (Barnes et al., 2021). This metric evaluates an entire tuple (h, t, e, p). A true positive is an exact match between the predicted and golden graphs, weighting the overlaps of the spans for each tuple component, averaged across the three spans. *Precision* is calculated by weighting the number of correctly predicted tokens divided by the total predicted tokens, while *Recall* is the ratio between the number of correctly predicted tokens and the number of golden tokens.

5 Results

Table 2 shows the official results of UFRGSent. We varied two parameters of our method during the runs – the types of questions used in the question-answering model to extract tuple candidates and the number of answers retrieved for each question. The run that achieved the best average result uses the target and opinion questions and one answer to generate the candidate tuples. The same result was obtained when we generated five or ten answers. This run produced the best result in four out of seven datasets – MultiBooked_ca, NoReC, OpeNER_es, and OpeNER_en.

Considering the Darmstadt_unis dataset, the best result was when we just considered the aspect questions and generated one answer. On the other hand, the best results for MPQA and MultiBooked_eu datasets were obtained using the three types of questions and one, five, or ten answers.

We noticed a significant loss in performance when generating three answers. We conclude that this happened because the first answer generated is the expected response most of the time. Producing additional answers tends to introduce noise in the candidate tuples. However, increasing the number of answers makes the correct answer be generated more times, yielding the right tuple choice.

In comparison with other participants, our average score was the 20^{th} result out of 31 participating systems. The dataset in which we achieved the best rank was MultiBooked_ca (15^{th}), while our worst results were Darmstadt_unis and MPQA datasets (20^{th} out of 31). Although our team did not submit results for the cross-lingual task, we emphasize that our solution is entirely multilingual – there is only one model, which extracted sentiment tuples for all datasets simultaneously. Therefore, our solution can be extended to any other language among the 104 languages present in multilingual BERT.

In order to assess the quality of our candidate tuple extraction, we measured the coverage of the question-answering results (*i.e.,* the percentage of gold tuples present in the set of candidate tuples). This measurement was done on the development dataset, for which we know beforehand the gold sentiment graphs. The results of the experiment can be seen in Table 3.

We set the hyper-parameter k to one, extracting only one answer per question. The evaluation was made in two ways – the exact match between gold tuple and candidates and the overlap, in which a pair of tuples containing an overlap between their tokens is considered a correct match. The experiment was repeated, varying the type of questions

Configuration		Dataset							Avg. Score
Question Types	# Ans	Darmstadt_unis	MPQA	MultiBooked_ca	MultiBooked_eu	NoReC	OpeNER_en	OpeNER_es	
H - T - E	1	0.230	**0.232**	0.505	**0.467**	0.251	0.431	0.399	0.359
	3	0.061	0.071	0.217	0.226	0.088	0.151	0.133	0.135
	5	0.230	**0.232**	0.505	**0.467**	0.251	0.431	0.399	0.359
	10	0.005	**0.232**	0.505	**0.467**	0.251	0.431	0.399	0.327
T - E	1	0.242 (20)	0.217 (20)	**0.521 (15)**	0.463 (17)	**0.270 (19)**	**0.452 (18)**	**0.427 (19)**	**0.370 (20)**
	3	0.082	0.042	0.284	0.286	0.135	0.232	0.204	0.18
	5*	0.242	0.217	**0.521**	0.463	**0.270**	**0.452**	**0.427**	**0.370**
	10*	0.242	0.217	**0.521**	0.463	**0.270**	**0.452**	**0.427**	**0.370**
T	1	**0.283**	0.231	0.456	0.374	0.241	0.399	0.338	0.332
	3	0.007	0.009	0.050	0.038	0.019	0.041	0.035	0.029
	5*	0.283	0.231	0.456	0.374	0.241	0.399	0.338	0.029
	10*	0.283	0.231	0.456	0.374	0.241	0.399	0.338	0.029
E	1	0.244	0.206	0.486	0.459	0.252	0.417	0.383	0.35
	3	0.029	0.011	0.141	0.148	0.100	0.063	0.086	0.083
	5*	0.244	0.206	0.486	0.459	0.252	0.417	0.383	0.35
	10*	0.244	0.206	0.486	0.459	0.252	0.417	0.383	0.35

Table 2: Official Results of UFRGSent in terms of Sentiment Graph F_1. Best results for a given dataset are in bold. $\star$ denotes that the results were obtained in the post-evaluation phase. The numbers between parentheses indicate the position achieved by UFRGSent in the competition.

Dataset	H - T - E		T		E		H	
	Exact	Overlap	Exact	Overlap	Exact	Overlap	Exact	Overlap
Darmstadt_unis	65.7%	76.2%	58.9%	67.7%	56.9%	67.3%	58.9%	63.3%
MPQA	75.1%	87.3%	72.2%	82.9%	71.2%	82.5%	72.7%	83.9%
MultiBooked_ca	41.1%	77.9%	30.2%	61.1%	32.6%	64.9%	10.9%	24.9%
MultiBooked_eu	42.1%	77.4%	29.8%	58.3%	33.6%	58.7%	13.6%	33.2%
NoReC	42.0%	79.7%	33.1%	64.9%	31.9%	66.3%	32.4%	42.2%
OpeNER_en	37.9%	76.5%	28.6%	57.2%	29.7%	58.5%	12.2%	42.8%
OpeNER_es	35.8%	72.8%	27.7%	50.4%	27.4%	57.8%	0.1%	26.0%

Table 3: Coverage of Extraction for Question-Answering System

used to extract the candidates. We first use the three types of questions together, and then we evaluate the coverage achieved by each type of question individually.

The results show that our question-answering technique provides good coverage of sentiment tuples. For darmstadt_unis and mpqa, we have an exact match with over 65% coverage. We also improved the measure on datasets with less coverage in the exact match experiments, considering tuple overlap. All datasets achieved at least 70% coverage in the overlap experiments.

Using only one component to extract candidate tuples reduces coverage for the question answering for all datasets. While the mpqa dataset was less affected by removing the components (loss of 2.9% in coverage using targets, 3.9% using opinion expressions, and 2.4% using holder questions). Other datasets had significant drops in coverage, especially when just holder questions were considered.

6 Conclusion

In this paper, we described our system submitted to SemEval-2022 Task 10. We designed a multilingual approach that relies on a QA system and an ASC model to find the Sentiment Graphs. The key idea is that the joint and incremental extraction of holder, target, and opinion helps to achieve a good coverage for UFRGSent.

In these preliminary experiments, we could not establish the quality of our tuple ranking and selection methods and we leave it for future work. Additionally, we are interested in understanding how well our multilingual model performs against a monolingual version of our technique, and how other state-of-the-art QA models can improve the extraction of sentiment tuples.

Acknowledgements.

This work was partially supported by CNPq/Brazil and by CAPES Finance Code 001.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402.

Jeremy Barnes, Andrey Kutuzov, Laura Ana Maria Oberländer, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. *arXiv preprint arXiv:2103.07665*.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694.

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

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515.

Bing Liu. 2011. Opinion mining and sentiment analysis. In *Web data mining: exploring hyperlinks, contents, and usage data*, 2 edition, chapter 11. Springer Science & Business Media.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Huaishao Luo, Tianrui Li, Bing Liu, and Junbo Zhang. 2019. DOER: Dual cross-shared RNN for aspect term-polarity co-extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 591–601.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033.

Yong Qian, Zhongqing Wang, Rong Xiao, Chen Chen, and Haihong Tang. 2021. SGPT: Semantic graphs based pre-training for aspect-based sentiment analysis. *arXiv preprint arXiv:2105.12305*.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

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

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2576–2585.

Biqing Zeng, Heng Yang, Ruyang Xu, Wu Zhou, and Xuli Han. 2019. LCF: A local context focus mechanism for aspect-based sentiment classification. *Applied Sciences*, 9(16).

Changchang Zeng, Shaobo Li, Qin Li, Jie Hu, and Jianjun Hu. 2020. A survey on machine reading comprehension—tasks, evaluation metrics and benchmark datasets. *Applied Sciences*, 10(21).

Meishan Zhang, Qiansheng Wang, and Guohong Fu. 2019. End-to-end neural opinion extraction with a transition-based model. *Information Systems*, 80:56–63.

OPI at SemEval-2022 Task 10: Transformer-based Sequence Tagging with Relation Classification for Structured Sentiment Analysis

Rafał Poświata
National Information Processing Institute, 00-608 Warsaw, Poland
rposwiata@opi.org.pl

Abstract

This paper presents our solution for SemEval-2022 Task 10: Structured Sentiment Analysis. The solution consisted of two modules: the first for sequence tagging and the second for relation classification. In both modules we used transformer-based language models. In addition to utilizing language models specific to each of the five competition languages, we also adopted multilingual models. This approach allowed us to apply the solution to both monolingual and cross-lingual sub-tasks, where we obtained average Sentiment Graph F1 of 54.5% and 53.1%, respectively. The source code of the prepared solution is available at https://github.com/rafalposwiata/structured-sentiment-analysis.

1 Introduction

Structured Sentiment Analysis (SSA) can be formulated as an information extraction task in which one attempts to find all of the opinion tuples $O = O_i, ..., O_n$ in a text. Each opinion O_i is a tuple (h, t, e, p) where h is a **holder** who expresses a **polarity** p towards a **target** t through a **sentiment expression** e, implicitly defining pairwise relationships between elements of the same tuple (Barnes et al., 2021). An example of such tuples as a structure sentiment graph was shown in Figure 1. This problem is relatively new and there has been little work published on the subject to date. To stimulate interest in this issue among the NLP community the SemEval-2022 Task 10: Structured Sentiment Analysis (Barnes et al., 2022) competition was organized. The contest consisted of two sub-tasks: monolingual and cross-lingual. In the monolingual sub-task, the systems were trained and then tested on the datasets in the same languages. In the cross-lingual sub-task, systems had to be prepared for Catalan, Basque and Spanish datasets, while data in these languages could not be used for training. This setup is often known as zero-shot cross-lingual transfer (Hu et al., 2020).

In this paper we present our system for this competition. We mainly focused on the solution for the monolingual track, however, it has also been successfully applied to the cross-lingual. The rest of the paper is organized as follows. Section 2 briefly describes related work. Section 3 shows an overview of used datasets. Section 4 elaborates on our solution. Experiments showing the effectiveness of the created system performed on development and test sets are presented in Section 5. The next section briefly describes the mistakes and limitations of our system. Finally, Section 7 concludes this paper.

2 Related Work

Structured Sentiment Analysis can be broken down into five sub-tasks: a) expression (opinion) extraction, b) target (aspect) extraction, c) holder extraction, d) defining the relationship between these elements, and e) assigning polarity (Barnes et al., 2021).[2]

A few years ago, the main focus was on **Aspect-Based Sentiment Analysis (ABSA)**, which only concerned on targets extraction (task b) and classifying the polarity towards them (task e) (Pontiki et al., 2014, 2015, 2016). Sequence tagging solutions have proven to be effective in this issue (Li et al., 2019a). An extension of this problem was **End2End Aspect-Based Sentiment Analysis (E2E-ABSA)**, which adds the issue of expression extraction (task a). He et al. (2019) propose an interactive multi-task learning network (IMN) which is able to jointly learn multiple related tasks simultaneously, to resolve this problem. Chen and Qian (2020) also use multi-task learning, but with relation propagation mechanisms and create Relation-Aware Collaborative Learning (RACL) framework. Tagging-based solutions also work well in this case

[1]Picture based on figure from Barnes et al. 2021.

[2]Phrases in parentheses indicate alternative names used interchangeably in the sentiment analysis literature.

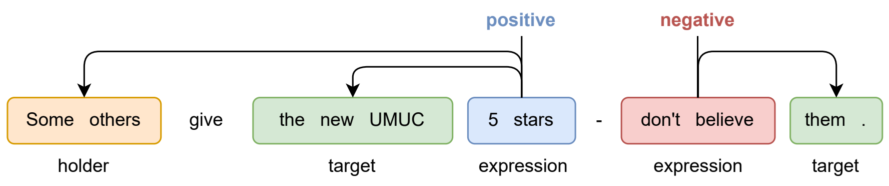


Figure 1: SSA example as a structure sentiment graph.[1]

		# sentences							# tags				
Dataset		all	w/o opinion	w/ one opinion	w/ two or more opinions	w/ mixed tags	w/ nested tags	w/ opposite polarity exp.	holders	targets	expressions		
											neg.	neu.	pos.
MPQA	train	5873	4619	917	337	92	108	0	1425	1481	698	337	671
	dev	2063	1647	304	112	49	38	0	406	494	215	124	231
	test	2113	1724	289	100	31	36	0	434	462	229	124	165
DS$_{\text{Unis}}$	train	2253	1572	583	98	3	0	1	63	806	364	102	340
	dev	232	150	69	13	0	0	0	9	98	54	15	29
	test	318	214	84	20	0	0	0	12	130	62	12	56
OpeNER$_{\text{en}}$	train	1744	344	638	762	0	0	0	266	2679	783	0	2101
	dev	249	51	83	115	0	0	0	49	371	116	0	284
	test	499	92	178	229	0	0	0	98	793	269	0	596
OpeNER$_{\text{es}}$	train	1438	186	500	752	0	0	0	176	2748	570	0	2472
	dev	206	32	77	97	0	0	0	23	363	70	0	317
	test	410	48	159	203	0	0	0	56	849	189	0	768
MultiB$_{\text{ca}}$	train	1174	172	508	494	0	0	0	169	1705	716	0	1273
	dev	167	27	79	61	0	2	0	15	211	107	0	151
	test	335	54	143	138	0	0	0	53	434	204	0	319
MultiB$_{\text{eu}}$	train	1063	164	478	421	0	0	0	205	1277	278	0	1401
	dev	152	32	68	52	0	0	0	33	152	36	0	167
	test	305	65	126	114	0	0	0	58	331	65	0	372
NoReC$_{\text{Fine}}$	train	8634	4079	2406	2149	802	472	173	898	6778	2753	0	5695
	dev	1531	710	441	380	119	87	32	120	1152	444	0	988
	test	1272	598	353	321	123	79	14	110	993	359	0	876

Table 1: Statistics of the datasets. Mixed tags means a situation where a given term in different opinions plays a different role, e.g. once it is a target and once it is a holder. Nested tags are when a term in one opinion is part of a term in another opinion. Opposite polarity expressions refers to the case where a sentence contains an expression that has a different sentiment depending on the opinion.

(Li et al., 2019b; Hu et al., 2019). The tasks listed above did not require resolving relationships between extracted tags.

The recently proposed, **Aspect Sentiment Triplet Extraction (ASTE)** fill this gap (Peng et al., 2020). The task is to extracting all aspects terms with their corresponding opinion terms and sentiment polarity (tasks a, b, d and e). Peng et al. (2020) propose two stage model. In the first stage, it extracts opinions and aspects along with sentiment using sequence tagging based on the unified BIO scheme. The second stage pairs up the predicted terms from the first stage to output triplets. ASTE is most similar to SSA, missing only the holder extraction.

For SSA, the subject of the competition, there are few solutions. Barnes et al. (2021) cast the structured sentiment problem as dependency graph parsing. Peng et al. (2021) extend this work and propose a sparse and fuzzy attention scorer with pooling layers which improves parser performance.

3 Datasets

Seven structured sentiment datasets in five languages were selected for the competition. The **MPQA** dataset (Wiebe et al., 2005) contains news documents from the world press in English. **DS$_{\text{Unis}}$** (Toprak et al., 2010) are English reviews of online universities and e-commerce. **OpeNER$_{\text{en}}$** and **OpeNER$_{\text{es}}$** (Agerri et al., 2013) consist of hotel reviews in English and Spanish, respectively. **MultiB$_{\text{eu}}$** and **MultiB$_{\text{ca}}$** (Barnes et al., 2018) are also hotel reviews, but in Basque and Catalan. The last dataset is **NoReC$_{\text{Fine}}$** (Øvrelid et al., 2020), a multi-domain dataset of professional reviews in Norwegian. The statistics of each dataset are sum-

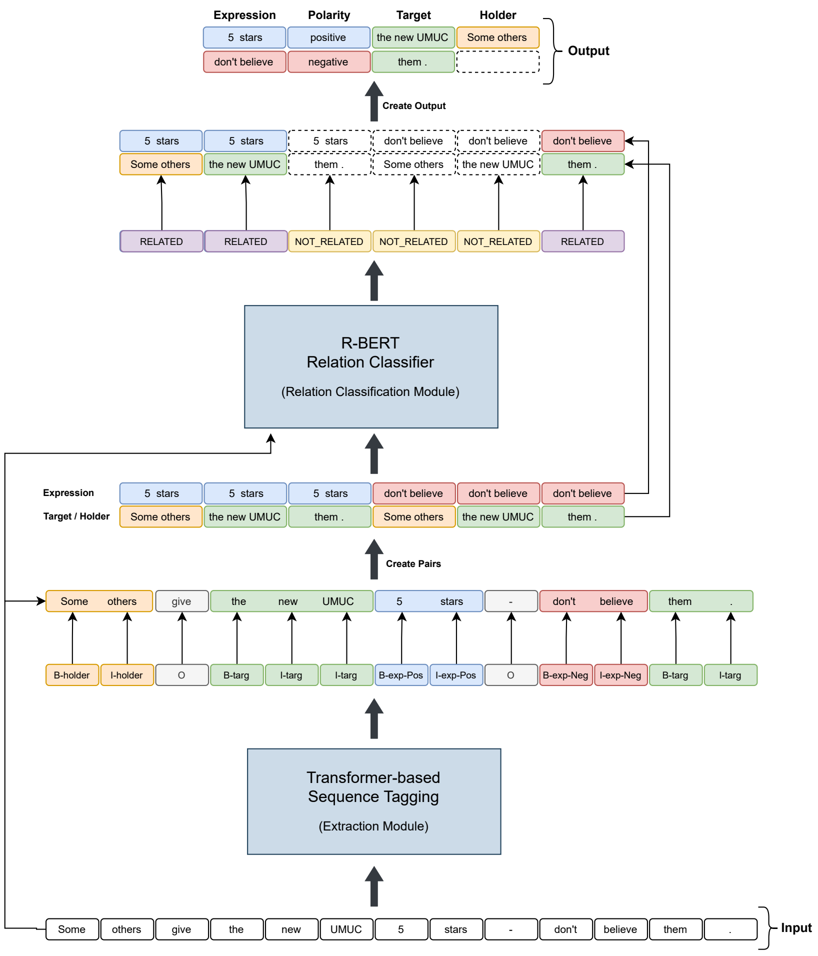


Figure 2: Architecture of the proposed solution.

marized in Table 1.

4 System Overview

The architecture of our solution is shown in Figure 2. This solution was inspired by the works of Li et al. (2019a,b); Hu et al. (2019), and especially the work of Peng et al. (2020). It consists of two main components: Extraction Module and Relation Classification Module. The first module is based on sequence tagging and is used to extract targets, holders and expressions with polarity. This is accomplished by using a suitable tagset which is a modification of the BIO scheme, consisting of the following tags: *{B-holder, B-targ, B-exp-Neg, B-exp-Neu, B-exp-Pos, I-holder, I-targ, I-exp-Neg, I-exp-Neu, I-exp-Pos, O}*. Transformer-based Language Model with a linear classification layer was used as an implementation. Having already extracted entities, the role of the second module is to classify whether there is a relationship between them. Specifically, it is about verifying that there is a holder and/or target associated with a particular expression. We utilized the R-BERT (Wu and He, 2019) model to accomplish this task. Based on a sentence with two appropriately marked entities (expression and holder/target), it determines whether or not they are related.[3] Entities that are related are combined and form an output. Extraction and Relation Classification modules are trained independently.

5 Experiments

5.1 Experimental Setup

To conduct the experiments, we first utilized the Simple Transformers library (Rajapakse, 2019) for the implementation of the Extraction Module. For the Relation Classification Module we modify publicly available source code of R-BERT.[4] The hyperparameters used in learning each of these modules are presented in Table 2. All models were run five times on a single GPU Tesla V100.

[3]For all the details, we would refer you to Wu and He 2019 paper.

[4]https://github.com/monologg/R-BERT

Parameter	Extraction	Relation Classification
Optimizer	AdamW	AdamW
Learning rate	5e-5	2e-5
Batch size	32	16
Dropout	0.1	0.1
Epochs	10	12
Validation after no. steps	200	200

Table 2: Parameter used for Extraction and Relation Classification modules during training.

5.2 Pretrained Language Models

We chose two types of language models based on transformer architecture for experiments: monolingual (at least one for each of the five competition languages) and multilingual. The use of multilingual models allowed us to obtain a more general solution and was necessary for the cross-lingual sub-task. Table 3 gives a brief summary of the models used. All models were downloaded from the Hugging Face hub[5].

Language	Model	Size	Source
English	BERT	base	Devlin et al. 2019
	RoBERTa	large	Liu et al. 2019
	XLNet	large	Yang et al. 2019
Spanish	BERTIN	base	de la Rosa et al. 2021
	RoBERTa-BNE	large	Gutiérrez-Fandiño et al. 2021
Catalan	Catalan-BERTa	base	Armengol-Estapé et al. 2021
Basque	BERTeus	base	Agerri et al. 2020
Norwegian	NorBERT	base	Kutuzov et al. 2021
	NB-BERT	large	Kummervold et al. 2021
Multilingual	mBERT	base	Devlin et al. 2019
	XLM-R	large	Conneau et al. 2020

Table 3: Transformer-based language models used in experiments.

5.3 Metrics

Following the works on Named Entity Recognition problem (Akbik et al., 2018; Yamada et al., 2020; Zhou and Chen, 2021), we used micro-average F1 score as our main measure for the Extraction Module. In addition for this module we added a detailed measure for each tag type i.e. F1 score for holders, targets and expressions with sentiment classes, separately. For the Relation Classification Module, we used Accuracy and macro-average F1 measures. Evaluation of the overall system was based on the official competition metric i.e. Sentiment Graph F1.

[5]https://huggingface.co/models

5.4 Development Results

Table 4 shows the results on the development sets for each module. For the Extraction Module, the **XLM-R** model was the best on five of the seven datasets. In only two cases (**MPQA** and **$\mathbf{DS_{Unis}}$**) language-specific models were found to be superior: **XLNet** and **RoBERTa**, respectively. For the Relation Classification Module, we only used models based on the BERT architecture, following the original R-BERT work (Wu and He, 2019). The **mBERT** usually proved to be the best (5/7 cases), except for two cases (**$\mathbf{MultiB_{eu}}$** and **$\mathbf{NoReC_{Fine}}$**) where **BERTeus** and **NB-BERT** were the best. The best models for each module were used to test the overall system. A summary of this experiment can be found in Table 5. The average Sentiment Graph F1 was 55.0%.

5.5 Test Results

The best models verified on the development sets were used on the test sets which are the official competition sets. For the monolingual sub-task, we used exactly the same configuration of models as in Table 5. For the cross-lingual sub-task, we used models trained on the **$\mathbf{OpeNER_{en}}$** set, namely **XLM-R** for extraction and **mBERT** for relation classification. There were two reasons for this choice. First is the use of multilingual models in both modules. Second, from the fact that the results on the development sets were high compared to the results for other models trained on English language sets. The results are summarized in Table 6. We achieved average SF1 scores of 54.5% and 53.1% for the monolingual and cross-lingual sub-tasks, respectively. This allowed us to rank 11th and 9th out of the 32 teams in these sub-tasks.

6 Errors Analysis

As a result of the used architecture, most errors are due to incorrect tagging. In particular, this is relevant to expressions where a correct sentiment is additionally required. The results were significantly worse for expressions limited in a given set, e.g., neutrals in the MPQA or DS_{Unis} sets. Furthermore, by using a single extraction model, the solution is not able to correctly handle more complicated cases such as mixed or nested tags or opposite polarity expressions. This is most noticeable in the **$\mathbf{NoReC_{Fine}}$** dataset.

		Extraction						Relation Classification	
Dataset	**Model**	**Holder F1**	**Target F1**	**Exp. F1**			**F1**	**Acc**	**F1**
				neg.	**neu.**	**pos.**			
MPQA	BERT	50.4	39.3	44.3	16.9	43.6	41.9	82.1	79.0
	RoBERTa	**58.8**	48.4	51.0	17.9	45.8	48.8	-	-
	XLNet	57.9	**49.1**	**51.7**	**25.5**	**48.5**	**49.9**	-	-
	mBERT	49.3	41.5	40.0	15.7	44.4	42.0	**82.6**	**79.4**
	XLM-R	56.8	46.9	50.8	17.4	47.8	48.3	-	-
DS$_{\text{Unis}}$	BERT	22.2	42.6	38.2	13.8	47.1	39.7	86.3	77.2
	RoBERTa	50.0	47.4	**44.1**	**14.3**	57.1	**46.1**	-	-
	XLNet	**66.7**	47.5	43.1	6.2	53.0	44.6	-	-
	mBERT	18.2	44.3	34.4	13.3	49.4	39.3	**92.1**	**88.2**
	XLM-R	28.6	**47.8**	40.6	6.5	**58.5**	44.1	-	-
OpeNER$_{\text{en}}$	BERT	70.1	73.2	56.9	-	67.7	68.2	94.6	94.0
	RoBERTa	68.4	76.1	61.1	-	**73.3**	72.0	-	-
	XLNet	68.9	73.8	63.2	-	72.5	71.3	-	-
	mBERT	**71.6**	70.8	53.5	-	68.7	67.2	**94.8**	**94.3**
	XLM-R	71.4	**77.2**	**66.1**	-	72.6	**73.3**	-	-
OpeNER$_{\text{es}}$	BERTIN	**77.4**	66.7	39.5	-	60.0	61.1	-	-
	RoBERTa-BNE	71.4	69.7	44.0	-	62.7	64.0	-	-
	mBERT	66.7	68.2	38.4	-	59.5	61.3	**92.6**	**90.9**
	XLM-R	75.0	**73.1**	**46.8**	-	**65.1**	**66.9**	-	-
MultiB$_{\text{ca}}$	Catalan-BERTa	69.2	72.8	55.0	-	74.5	69.2	-	-
	RoBERTa-BNE	47.1	69.3	48.7	-	74.7	65.7	-	-
	mBERT	61.5	70.1	57.5	-	73.4	67.9	**94.0**	**92.7**
	XLM-R	**72.7**	**73.4**	**63.7**	-	**79.0**	**73.0**	-	-
MultiB$_{\text{eu}}$	BERTeus	**71.7**	**76.7**	46.0	-	59.3	64.1	**89.5**	**89.3**
	RoBERTa-BNE	52.6	59.8	21.1	-	48.0	49.4	-	-
	mBERT	54.9	64.1	36.1	-	53.0	55.0	87.3	87.2
	XLM-R	69.4	74.1	**48.7**	-	**65.8**	**66.9**	-	-
NoReC$_{\text{Fine}}$	NorBERT	62.0	51.6	23.4	-	36.6	39.8	85.5	85.3
	NB-BERT	**64.6**	56.0	28.1	-	40.6	44.1	**88.0**	**87.8**
	mBERT	54.7	51.2	18.2	-	31.2	36.2	86.2	86.0
	XLM-R	63.4	**60.3**	**32.3**	-	**40.7**	**46.5**	-	-

Table 4: Results for the Extraction and Relation Classification modules on development sets. Underlined and bolded numbers indicate the best result for the metric and dataset.

Dataset	**Extraction Model**	**Relation Classification Model**	**SF1**
MPQA	XLNet	mBERT	37.7
DS$_{\text{Unis}}$	RoBERTa	mBERT	34.5
OpeNER$_{\text{en}}$	XLM-R	mBERT	69.1
OpeNER$_{\text{es}}$	XLM-R	mBERT	66.5
MultiB$_{\text{ca}}$	XLM-R	mBERT	65.7
MultiB$_{\text{eu}}$	XLM-R	BERTeus	64.7
NoReC$_{\text{Fine}}$	XLM-R	NB-BERT	47.0
		Average score	**55.0**

Table 5: Overall system results on development sets.

Dataset	**Monolingual**	**Cross-lingual**
MPQA	32.6	-
DS$_{\text{Unis}}$	39.5	-
OpeNER$_{\text{en}}$	67.0	-
OpeNER$_{\text{es}}$	66.3	56.4
MultiB$_{\text{ca}}$	65.0	58.6
MultiB$_{\text{eu}}$	65.3	44.4
NoReC$_{\text{Fine}}$	45.9	-
Average score	**54.5**	**53.1**

Table 6: Overall system results on test sets (official results of the competition).

7 Conclusion

In this paper, we presented a solution to the SemEval-2022 Task 10: Structured Sentiment Analysis. A simple architecture based on sequence tagging and relation classification achieved good results. The use of multilingual language models enabled the solution to be used for monolingual and cross-lingual sub-tasks. At the same time it can be easily extended e.g. by using an additional CRF layer (Souza et al., 2019) in the Extraction

Module or by using other multilingual language models e.g. InfoXLM (Chi et al., 2021).

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Rodrigo Agerri, Iñaki San Vicente, Jon Ander Campos, Ander Barrena, Xabier Saralegi, Aitor Soroa, and Eneko Agirre. 2020. Give your text representation models some love: the case for basque. In *Proceedings of the 12th International Conference on Language Resources and Evaluation*.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jordi Armengol-Estapé, Casimiro Pio Carrino, Carlos Rodriguez-Penagos, Ona de Gibert Bonet, Carme Armentano-Oller, Aitor Gonzalez-Agirre, Maite Melero, and Marta Villegas. 2021. Are multilingual models the best choice for moderately under-resourced languages? A comprehensive assessment for Catalan. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4933–4946, Online. Association for Computational Linguistics.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Oberländer Laura Ana Maria Kutuzov, Andrey and, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694, Online. Association for Computational Linguistics.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Javier de la Rosa, Eduardo González, Paulo Villegas, Pablo González de Prado, Manu Romero, and María Grandury. 2021. BERTIN project.

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

Asier Gutiérrez-Fandiño, Jordi Armengol-Estapé, Marc Pàmies, Joan Llop-Palao, Joaquín Silveira-Ocampo, Casimiro Pio Carrino, Aitor Gonzalez-Agirre, Carme Armentano-Oller, Carlos Rodriguez-Penagos, and Marta Villegas. 2021. Spanish language models.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515, Florence, Italy. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546, Florence, Italy. Association for Computational Linguistics.

Per E Kummervold, Javier De la Rosa, Freddy Wetjen, and Svein Arne Brygfjeld. 2021. Operationalizing a national digital library: The case for a Norwegian transformer model. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 20–29, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Andrey Kutuzov, Jeremy Barnes, Erik Velldal, Lilja Øvrelid, and Stephan Oepen. 2021. Large-scale contextualised language modelling for Norwegian. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 30–40, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019a. A unified model for opinion target extraction and target sentiment prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6714–6721.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019b. Exploiting BERT for end-to-end aspect-based sentiment analysis. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 34–41, Hong Kong, China. Association for Computational Linguistics.

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

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8600–8607.

Letain Peng, Zuchao Li, and Hai Zhao. 2021. Sparse fuzzy attention for structured sentiment analysis. *ArXiv*, abs/2109.06719.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Haris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

T. C. Rajapakse. 2019. Simple transformers. `https://github.com/ThilinaRajapakse/simpletransformers`.

Fábio Souza, Rodrigo Nogueira, and Roberto de Alencar Lotufo. 2019. Portuguese named entity recognition using bert-crf. *ArXiv*, abs/1909.10649.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 2361–2364, New York, NY, USA. Association for Computing Machinery.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

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

Wenxuan Zhou and Muhao Chen. 2021. Learning from noisy labels for entity-centric information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5381–5392, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

ETMS@IITKGP at SemEval-2022 Task 10: Structured Sentiment Analysis Using A Generative Approach

R Raghav* and **Adarsh Vemali*** and **Rajdeep Mukherjee**
Indian Institute of Technology Kharagpur, India
{rraghav5600, adarsh.vemali, rajdeep1989}@iitkgp.ac.in

Abstract

Structured Sentiment Analysis (SSA) deals with extracting opinion tuples in a text, where each tuple (h, e, t, p) consists of h, the holder, who expresses a sentiment polarity p towards a target t through a sentiment expression e. While prior works explore graph-based or sequence labeling-based approaches for the task, we in this paper present a novel unified generative method to solve SSA, a SemEval-2022 shared task. We leverage a BART-based encoder-decoder architecture and suitably modify it to generate, given a sentence, a sequence of opinion tuples. Each generated tuple consists of seven integers respectively representing the indices corresponding to the start and end positions of the holder, target, and expression spans, followed by the sentiment polarity class associated between the target and the sentiment expression. We perform rigorous experiments for both Monolingual and Cross-lingual subtasks, and achieve competitive Sentiment F1 scores on the leaderboard in both settings.

1 Introduction

Structured Sentiment Analysis (SSA) is the task of extracting structured information around sentiment expressions present in text in the form of opinion tuples $O = \{O_1, O_2, ..., O_n\}$, where each opinion tuple $O_i = (h, t, e, p)$ consists of h, the *holder* (or *source*, used interchangeably) who expresses a sentiment polarity p towards a *target* (or *aspect*) t using an opinion or sentiment expression e (Barnes et al., 2021a). Prior works (Liu, 2012; Peng et al., 2020) have highlighted the importance of addressing sentiment analysis as a structured prediction problem in order to capture the complete information around various opinions expressed in the text. The task of SSA thus expects to exploit the pairwise interactions between the members of the same opinion tuple during the extraction process.

*Equal contribution

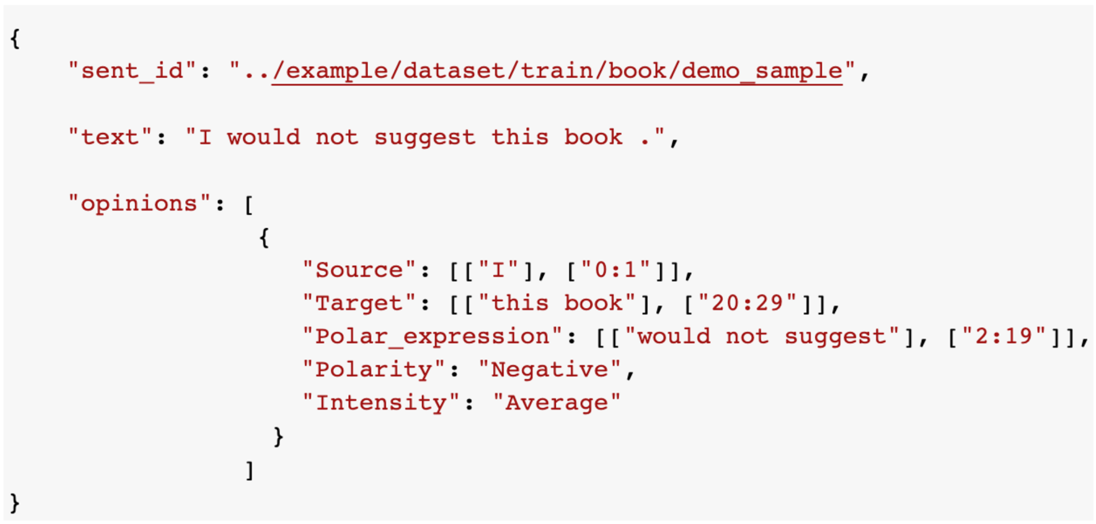
With the exponential growth of online marketplaces and user-generated content therein, SSA or near similar tasks of aspect-sentiment-opinion triplet extraction (Peng et al., 2020; Mukherjee et al., 2021a; Yan et al., 2021), and aspect-category-sentiment-opinion quad extraction (Cai et al., 2021), the newest additions under the broader umbrella of aspect-based sentiment analysis (ABSA) (Pontiki et al., 2014a,b) have become more important than ever (Mukherjee et al., 2021b). In the face of ever-expanding choices, it becomes a challenging necessity to take educated explainable decisions from past user reviews. SSA guides the learning in the proper direction by facilitating an automated way to focus on major sentiment or opinion indicators. As a result, the task has wide applications in various market segments, such as e-commerce, food delivery, healthcare, ride sharing, travel and hospitality, to name a few.

Previous efforts on SSA have primarily focused on two approaches: sequence labeling-based (He et al., 2019), and graph-based (Barnes et al., 2021b). The former tries to first predict the presence/absence of targets and expressions in the text by sequentially labeling each text token using BIOES[1] tags, before modeling their interaction to predict the sentiment polarity. The latter models the task as a dependency graph parsing problem, where the sentiment expression is considered as the root node, and the other elements are connected via arcs that represent their relationships. Different from these, we present a novel generative approach to solve SSA. More specifically, we take motivation from a unified generative framework recently proposed by Yan et al. (2021) to solve several ABSA tasks. We suitable modify their BART-based encoder-decoder architecture to adapt it for SSA. Given a sentence, the model is

[1]BIOES is a tagging scheme commonly used for sequence labeling tasks. B, I, E, O, and S respectively denote the *begin, inside, outside, end*, and *single* tags corresponding to an entity.

Dataset Name	Language	% Null (Train, Dev)	Size (Train, Dev, Test)
NoReC_fine (Øvrelid et al., 2020)	Norwegian	(47.24%, 46.37%)	(8634, 1531, 1272)
MultiBooked_eu (Barnes et al., 2018)	Basque	(15.43%, 21.05%)	(1064, 152, 305)
MultiBooked_ca (Barnes et al., 2018)	Catalan	(14.65%, 16.17%)	(1174, 168, 336)
OpeNER_es (Agerri et al., 2013)	Spanish	(12.93%, 15.53%)	(1438, 206, 410)
OpeNER_en (Agerri et al., 2013)	English	(19.72%, 20.48%)	(1744, 249, 499)
MPQA (Wiebe et al., 2005)	English	(77.92%, 79.18%)	(5873, 2063, 2112)
Darmstadt_unis (Toprak et al., 2010)	English	(69.77%, 64.66%)	(2253, 232, 318)

Table 1: Dataset Statistics



```
{
    "sent_id": "../example/dataset/train/book/demo_sample",

    "text": "I would not suggest this book .",

    "opinions": [
                  {
                     "Source": [["I"], ["0:1"]],
                     "Target": [["this book"], ["20:29"]],
                     "Polar_expression": [["would not suggest"], ["2:19"]],
                     "Polarity": "Negative",
                     "Intensity": "Average"
                  }
                ]
}
```

Figure 1: Annotation Format. The value of "opinions" in the JSON is a list of opinion tuples present in the "text". Each item in the list is a dictionary, with keys being the tuple elements, and the values corresponding to their representation in the text. *Source, Target* and *Polar_expression* are annotated with the the actual word spans appearing in the text along and their respective character indices. *Polarity* represents the sentiment expressed in the tuple, and *Intensity* represents its strength.

trained to generate a sequence of tuples, each consisting of seven integer outputs corresponding to the start and end indices of the holder, target and sentiment expression spans appearing in the text, and finally the polarity class. An example is shown and described in Figure 2.

We participate in the SemEval 2022 Task 10: Structured Sentiment Analysis (Barnes et al., 2022) hosted on CodaLab. In order to demonstrate the efficacy of our proposed solution, we attempt both the *monolingual* and *cross-lingual* subtasks and achieve competitive performance on the leaderboard in both settings. As part of the (sub)tasks, we testify our approach on multiple datasets spanning across five different languages - *English* (Darmstadt_unis, OpeNER_en, MPQA), Basque (MultiBooked_eu), *Catalan* (MultiBooked_ca), *Norwegian* (NoReC_fine) and *Spanish* (OpeNER_es). A summary of dataset statistics is reported in Table 1. While the evaluation scripts were made available to us by the task organizers to analyze our performance, the final leaderboard scores were obtained on a hidden test set.

2 Task Overview

2.1 Task Definition

SSA aims to predict all the structured sentiment graphs present in a given text. A graph is formally represented by opinion tuples $O = O_1, O_2, ..., O_n$, where each opinion tuple O_i consists of a quadruple of the holder h, the target t, the sentiment expression e, and the sentiment polarity p.

2.2 Datasets

As summarized in Table 1, we are provided with a total of 7 datasets, as part of the shared task, spanning across 5 different languages. Each dataset is a collection of sentences, along with their corresponding annotated opinion tuples, each consisting of (*Source, Target, Polar Expression*, and *Polarity*).

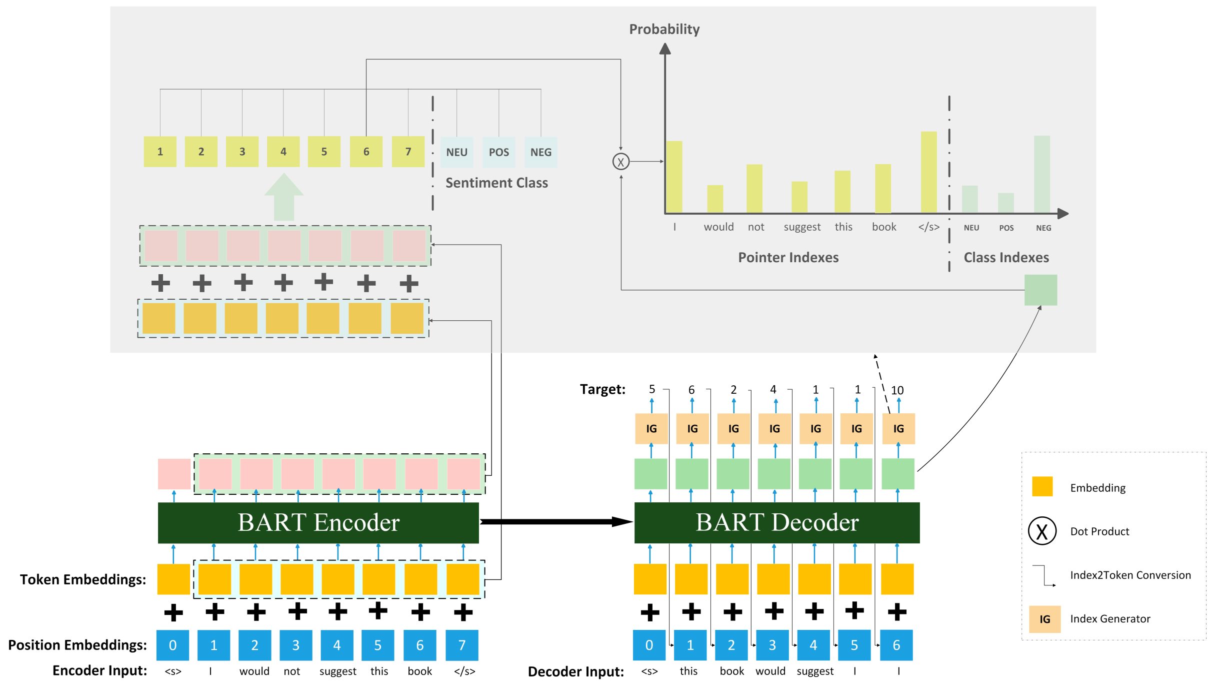


Figure 2: Model Architecture. The above figure shows an example where the input is "<s> I would not suggest this book </s>", and the corresponding output is "5, 6, 2, 4, 1, 1, 10" (Only partial decoder sequence is shown. Here, 7 (</s>) should be the next generation index). The "Index2Token Conversion" module converts the pointer indices back to the corresponding tokens in the source text, and the class index to the corresponding sentiment polarity.

While the *Intensity* of the expressed sentiment is also provided as part of the annotations, intensity classification/regression is not included as part of the task. An example is shown in Figure 1. All the data is provided through CodaLab, and GitHub.

3 Methodology

3.1 Task Formulation

We take a generative approach to formulate SSA as a structured prediction problem. We note here that predicting the holder (source), target (aspect), and polar expression spans correspond to extraction tasks, whereas sentiment polarity prediction is a classification task. Following (Yan et al., 2021), we model both these tasks in a unified generative framework by representing span entities with their start and end pointer indices corresponding to the text, and sentiment polarity with a class index.

We denote the holder, target, polar expression and sentiment polarity as *h*, *t*, *pe*, and *sp* respectively. The start and end index of each term is represented using superscripts ^{s} and ^{e}. For an input $X = [x_1, ..., x_n]$, where x_i is the i^{th} word in the text, the target $Y = [t_i^s, t_i^e, pe_i^s, pe_i^e, h_i^s, h_i^e, sp_i, ...]$ is defined as a sequence of tuples, each consisting of seven indices corresponding to an opinion tuple (h, t, pe, sp). An example sentence along with its target sequence are shown in Figure 2.

3.2 System Overview

Our model, as shown in Figure 2, consists of an encoder-decoder architecture with BART (Lewis et al., 2019) as its backbone. Given an input $X = [x_1, ..., x_n]$, the model is trained to produce an output $Y = [y_1, ..., y_m]$ (with y_0 representing the start-of-sequence token, $< s >$). The probability distribution is modeled as:

$$P(Y|X) = \prod_{t=1}^{m} P_t \quad (1)$$

Here $P_t = P(y_t|X, Y_{<t})$ represents the index probability distribution for the t^{th} time step.

3.2.1 Encoder

BART comprises of a bi-directional encoder. We denote the encoded vector of the input sentence X as H^e. For the sake of simplicity, we ignore the start-of-sequence token ($< s >$) in the equations.

$$H^e = BARTEncoder([x_1, ..., x_n]) \quad (2)$$

Here $H^e \in R^{n \times d}$, with d as the hidden dimension.

Dataset	% Null Instances
NoReC_fine	47.24% (As in the dataset)
MultiBooked_eu	10%
MultiBooked_ca	14.65 % (As in the dataset)
OpeNER_es	4%
OpeNER_en	14%
MPQA	50%
Darmstadt_unis	69.77 % (As in the dataset)

Table 2: % null instances in processed train sets.

3.2.2 Index2Token Conversion

Since the entity spans are decoded as corresponding start and end indices, and the sentiment polarity is decoded as corresponding class index, the indices need to be converted back to tokens before the BART Decoder can use them along with the encoder hidden state H^e for generating the next token (index) in the t^{th} time step. For each $y_t \in Y_{<t}$, we therefore use the following conversion strategy:

$$\hat{y}_t = \begin{cases} X_{y_t} & \text{if } y_t \text{ is a pointer index,} \\ Pol_{y_t - n} & \text{if } y_t \text{ is a class index} \end{cases} \quad (3)$$

where $Pol = [p_1, p_2, p_3]$ is the list of polarity classes. In our implementation, $y_t \in [1, n+3]$. The first sentence token x_1 has the pointer index 1.

3.2.3 Decoder

Our BART decoder now uses H^e and the converted decoder outputs $\hat{Y}_{<t}$ to obtain the t^{th} decoder hidden state:

$$H_t^d = BARTDecoder(H^e, \hat{Y}_{<t}) \quad (4)$$

where $H_t^d \in R^d$. Finally, H_t^d is used to predict the token probability distribution P_t. We request our readers to refer to (Yan et al., 2021) for additional details.

3.2.4 Training and Workflow

Teacher forcing with negative log likelihood as the loss function is used to train the model. During inference, beam search is used to generate the target sequence Y in an auto-regressive manner. Finally, the generated sequence is translated back into the phrase spans and sentiment polarity. As shown in Figure 2, we now illustrate the working of our proposed method using an example sentence:
I would not suggest this book .

1. The input <s> I would not suggest this book </s> is sent as input to the BART Encoder. As specified in Section 3.2.2, the word "I" is mapped to position index 1. Accordingly, </s> is mapped to index 7. Thereafter, each sentiment polarity is assigned a class index in sequence. In this case, the polarity values neutral, positive, and negative, are assigned class indices 8, 9 and 10 respectively.

2. The BART Decoder is trained to generate a sequence of indices till the end-of-sequence index (here 7) is generated. Corresponding to each opinion tuple, the decoder respectively predicts the start and end word indices for the target, polar expression, and source and finally, the polarity class index.

3. In our case, the expected target sequence is 5, 6, 2, 4, 1, 1, 10, 7. Here, (5, 6) represents the *target* phrase "this book", (2, 4) represents the *polarity expression* phrase "would not suggest", (1, 1) represents the *holder* phrase "I", and 10 represents the *negative* polarity class.

4. During inference, a decoding algorithm is employed making use of the *Index2Token Conversion* module to respectively convert the indices back to the text tokens and polarities before presenting to the end user.

4 Experiments

4.1 Data Preprocessing

In addition to fully annotated sentences, we observe the following two kinds of instances in all the datasets: (a) no opinion tuples at all - hereby referred to as the null examples, and (b) few empty entities in a single opinion tuple. We note here that our tuple representation scheme expects position indices of words appearing in the sentence. In order to accommodate for the above mentioned cases, we add a string *None* in the beginning of every sentence. This helps us to map the missing entities (e.g. holder or aspect in an opinion tuple) to a phrase present in the sentence.

After rigorous experiments, we set an optimal threshold for the proportion of null examples to be used for training in each of the datasets as reported in Table 2. During training, we found that limiting the proportion of null examples to the reported values significantly helped us in achieving the best performance on the respective datasets across monolingual and cross-lingual settings.

Dataset	% null	Batch Size	LR	F1 Score	Best Epoch
NoReC_fine	10	16	1E-04	0.320	35
NoReC_fine	10	16	2E-05	0.310	8
NoReC_fine	10	16	5E-05	0.323	11
OpeNER_es	10	8	1E-04	0.351	33
OpeNER_es	10	8	2E-05	0.482	36
OpeNER_es	10	8	5E-05	0.566	33
OpeNER_en	10	8	1E-04	0.674	43
OpeNER_en	10	8	2E-05	0.661	32
OpeNER_en	10	8	5E-05	0.675	7
Darmstadt_unis	10	16	1E-04	0.259	30
Darmstadt_unis	10	16	2E-05	0.276	34
Darmstadt_unis	10	16	5E-05	0.289	19

Table 3: Learning Rate Tuning

Dataset	% null	Batch Size	LR	F1 Score	Best Epoch
NoReC_fine	15	8	5E-05	0.317	8
NoReC_fine	30	16	5E-05	0.295	17
NoReC_fine	As in Dataset	16	5E-05	0.357	28
MultiBooked_eu	5	8	5E-05	0.422	20
MultiBooked_eu	10	8	5E-05	0.427	21
MultiBooked_eu	As in Dataset	8	5E-05	0.401	19
MultiBooked_ca	5	8	5E-05	0.552	35
MultiBooked_ca	10	8	5E-05	0.545	32
MultiBooked_ca	As in Dataset	8	5E-05	0.556	46
OpeNER_es	4	8	5E-05	0.572	31
OpeNER_es	8	8	5E-05	0.57	4
OpeNER_es	As in Dataset	8	5E-05	0.571	25
OpeNER_en	7	16	5E-05	0.677	24
OpeNER_en	14	16	5E-05	0.678	25
OpeNER_en	As in Dataset	16	5E-05	0.674	39
MPQA	25	8	5E-05	0.355	20
MPQA	50	16	5E-05	0.366	44
MPQA	As in Dataset	16	5E-05	0.359	42
Darmstadt_unis	25	16	5E-05	0.268	42
Darmstadt_unis	45	16	5E-05	0.277	33
Darmstadt_unis	As in Dataset	16	5E-05	0.312	36

Table 4: Null Parameter Tuning for each dataset

4.2 Experimental Setup

For the *monolingual* setting, we used the train, validation, and test splits of the same datasets. While experimenting on the *English* datasets (Darmstadt_unis, MPQA, and OpeNER_en), we use BART-base[2] as the backbone. For the *Non-English* datasets (OpeNER_es, Multibooked_eu, Multibooked_ca, and NoReC_fine), we use BART-large-MNLI [3] as the backbone. In the *cross-lingual* setting, we trained our models using the combined training data from all *English* datasets and evaluated them on the test sets of respective *Non-English* datasets (NoReC_fine not included as part of this setting). Here, we used BART-large-MNLI as the backbone for all our cross-lingual experiments.

Although predicting the intensities of sentiment polarities was not included as part of the shared task, we hypothesized that additionally learning the intensity prediction task would help the model in predicting the other entities (h, t, e, p) better in a multi-task setting. We performed additional experiments to verify our hypothesis. However, we observed little to no difference in the final results. Accordingly, we excluded intensity prediction from further consideration while performing our final experiments. We make our code repository publicly available at https://github.com/Sherlock-Jerry/SSA-SemEval.

4.3 Hyperparameter Tuning

We train all our models on Tesla P100-PCIE 16GB GPU. We perform extensive tuning experiments to

[2]https://huggingface.co/facebook/bart-base

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

Parameters	English	Non-English
Batch Size	16	8
Learning Rate	5E-05	5E-05
Epochs	50	50
BART Model	Base	Large-MNLI
% Null	Varies with Dataset	

Table 5: Final set of hyperparameters

Dataset	Ours	Barnes et al. (2021b)
NoReC_fine	0.351	0.312
MultiBooked_eu	0.438	0.547
MultiBooked_ca	0.508	0.568
OpeNER_es	0.544	Not available
OpeNER_en	0.626	Not available
MPQA	0.327	0.188
Darmstadt_unis	0.330	0.265

Table 6: Monolingual SubTask: Test Set SG-F1 Scores.

obtain the optimal set of hyperparameters. To determine the optimal learning rate, we ran monolingual experiments on two English and Non-English datasets respectively, as elucidated in Table 3. Based on our observations, we fixed a common learning rate of $5e-5$ for our final experiments across both the settings, monolingual as well as cross-lingual. For obtaining the optimal proportion of null instances to be used for training the final models, we perform three iterations of monolingual experiments on each dataset, each time with a different proportion of null instances used for training the models, as reported in Table 4. The final null thresholds are reported in Table 2. Table 5 summarizes the set of hyperparameters used for reporting our final results for both the subtasks. For all our experiments, the model selected according to the best F1 score on the validation data was used to evaluate on the test data.

4.4 Evaluation Metrics

Sentiment Graph F1 (SG-F1) is used to evaluate the models. *True Positive* is defined as an exact match (including polarity) at graph-level, weighted by the token-level overlap between the gold and predicted spans for holder, target, and polar expression, averaged across all three spans. *Precision* is calculated by weighting the number of correctly predicted tokens divided by the total number of predicted tokens. *Recall* is calculated by dividing the number of correctly predicted tokens by the number of gold tokens, thereby allowing for empty holders and targets which exist in the gold standard.

5 Results

5.1 Monolingual Subtask

We report the results for our monolingual experiments in Table 6 and compare them with the existing state-of-the-art (SOTA) results reported in Barnes et al. (2021b). Amongst the given datasets, our model performs the best on OpeNER_en and OpeNER_es, and has a relatively poor performance on MPQA, Darmstadt_unis and NoReC_fine. Despite this, we comfortably outperform the existing SOTA on these datasets. On the public leaderboard hosted on CodaLab, we achieved 18th rank out of 32 entries for this task.

5.2 Crosslingual Subtask

We report the results for our crosslingual experiments in Table 7. In this paradigm, we used all the English datasets for training our model, and tested our best trained models on the test sets of the respective Non-English datasets.

Dataset	SG-F1
EN-EU (MultiBooked_eu)	0.431
EN-CA (MultiBooked_ca)	0.506
EN-ES (OpeNER_es)	0.542

Table 7: Corsslingual SubTask: Test Set SG-F1 Scores.

Here, we achieved 11th rank out of 32 entries.

5.3 Qualitative Analysis

- In the monolingual subtask, we observed that our model performs poorly on the datasets with large proportions of empty opinion tuples (null instances). As can be confirmed from Table 2, MPQA, Darmstadt_unis, and NoRec_fine have high empty tuple proportion as against OpeNER_en, and OpeNER_es with low proportion of null instances.

- We observed that for datasets having lengthy sentences, our model performs relatively poor. A comparison between the distribution of test sentence lengths and the Sentiment Graph F1 scores for each dataset is shown in Figure 3.

- We also observed annotation errors in the datasets. For instance, given the test sentence

Label Type	Source	Target	Polar Expression	Polarity
Gold	-	The size of room	reasonable	Positive
Prediction	-	The size	reasonable	Positive
Gold	-	walls	in very poor conditions	Negative
Prediction	-	walls	very poor	Negative
Gold	-	floor	in very poor conditions	Negative
Prediction	-	floor	in very poor conditions	Negative
Gold	-	ceiling	in very poor conditions	Negative
Prediction	-	ceiling	in very poor conditions	Negative

Table 8: Ground truth opinion tuples and model predictions for the sentence: "The size of room is reasonable , but floor , walls and ceiling are in very poor conditions".

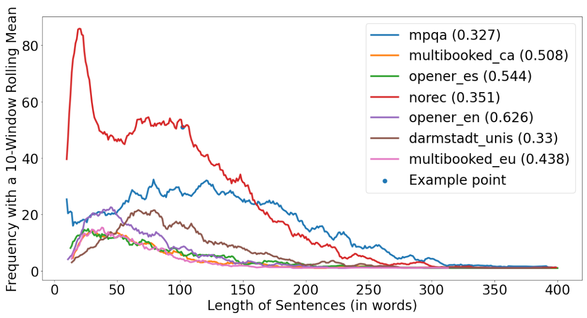


Figure 3: Comparing the distribution of test sentence lengths with best-obtained SG-F1 scores for each dataset. The "Example point" shows that there 51 sentences in the test set for NoRec_fine with length 100.

"So wonderful to see people go to work smiling and leave work still smiling and happy.", our trained generative model correctly predicts an opinion tuple with "see people go to work" as the target, and "So wonderful" as the opinion expression with a "Positive" sentiment. However, no ground truth opinion tuples are associated with the sentence.

- As reported in Table 8, we found a few instances where our model correctly predicts the necessary entities; but due to ambiguity in labelling (even at human level), we saw a mismatch. Here, our model predicts "The size" as the *target* whereas the gold standard expects "The size of room". Similar is the case with "in very poor conditions" (gold standard) versus the predicted phrase "very poor".

6 Related Work

Previous efforts on SSA have primarily focused on two approaches: sequence labeling-based (He et al., 2019), and graph-based (Barnes et al., 2021b). The corresponding scores for both these approaches are considered as baselines by the task organizers.

6.1 Sequence Labelling

In this approach, (He et al., 2019) propose a pipeline of sequence labelling and relation classification tasks. More specifically, three different sequence labellers based on BIOES tags are trained to predict the three span-based opinion entities, i.e. the holder, the target, and the polar expression. Finally, their relationship is exploited using a separate classification layer on top to predict the connecting sentiment polarity. However, such an approach inherently suffers from error propagation between the steps. Also, the inter-dependency especially between the target and the polar expression is not captured when the spans are predicted in isolation.

6.2 Dependency Graph Parsing

(Barnes et al., 2021b) have treated this task as a bilexical dependency graph prediction problem. They present two different versions of their proposed approach - (a) head-first and (b) head-final, as shown in Figure 4.

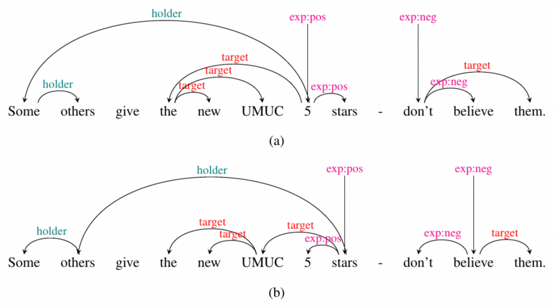


Figure 4: Dependency Graph Parsing

In both cases, the sentiment expression is considered as the root node, and the other elements are connected via arcs that represent their relationships. This approach builds upon the Dozat and Manning parser, implemented in (Kurtz et al., 2020).

7 Conclusion

Different from prior methods, we in this work present a novel generative approach to tackle the task of Structured Sentiment Analysis. We formulate the task as a structured prediction problem. Our BART-based encoder-decoder architecture is trained to predict a sequence of indices corresponding to each opinion tuple present in the text. The generated indices suitably represent the holder, target, and polar expression spans by their start and end token positions, and the sentiment polarity by its corresponding class. As part of SemEval 2022 Task 10, we participate in both monolingual and crosslingual subtasks, and achieve competitive performance on the leaderboard for both settings. In future, we would like to explore paraphrasing-based generative methods for the task.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021a. Structured sentiment analysis as dependency graph parsing. *arXiv preprint arXiv:2105.14504*.

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021b. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Andrey Kutuzov, Laura Ana Maria Oberländer, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Hongjie Cai, Rui Xia, and Jianfei Yu. 2021. Aspect-category-opinion-sentiment quadruple extraction with implicit aspects and opinions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 340–350, Online. Association for Computational Linguistics.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515, Florence, Italy. Association for Computational Linguistics.

Robin Kurtz, Stephan Oepen, and Marco Kuhlmann. 2020. End-to-end negation resolution as graph parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 14–24, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Rajdeep Mukherjee, Tapas Nayak, Yash Butala, Sourangshu Bhattacharya, and Pawan Goyal. 2021a. PASTE: A tagging-free decoding framework using pointer networks for aspect sentiment triplet extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9279–9291, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rajdeep Mukherjee, Shreyas Shetty, Subrata Chattopadhyay, Subhadeep Maji, Samik Datta, and Pawan Goyal. 2021b. Reproducibility, replicability and beyond: Assessing production readiness of aspect based sentiment analysis in the wild. In *Advances in Information Retrieval*, pages 92–106, Cham. Springer International Publishing.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *AAAI*.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014a. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014b. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Hang Yan, Junqi Dai, Xipeng Qiu, Zheng Zhang, et al. 2021. A unified generative framework for aspect-based sentiment analysis. *arXiv preprint arXiv:2106.04300*.

SLPL-Sentiment at SemEval-2022 Task 10: Making Use of Pre-Trained Model's Attention Values in Structured Sentiment Analysis

Sadrodin Barikbin
Speech and Language Processing Lab
Sharif University of Technology

barikbin@ce.sharif.edu

Abstract

Sentiment analysis is a useful problem which could serve a variety of fields from business intelligence to social studies and even health studies. Using SemEval 2022 Task 10 formulation of this problem and taking sequence labelling as our approach, we propose a model which learns the task by finetuning a pretrained transformer, introducing as few parameters (150k) as possible and making use of precomputed attention values in the transformer. Our model improves shared task baselines on all task datasets.

1 Introduction

Sentiment analysis has many applications in different fields. From social and political studies (Rodríguez-Ibáñez et al., 2021) to business intelligence and even health studies (Alamoodi et al., 2021). SemEval 2022 Task 10 (Barnes et al., 2022) formulates this problem as to extract a graph of sentiment-related entities and names that, **Structured Sentiment Analysis**. An example of such graph is depicted in figure2 Specifically given an input sentence, one should extract a list of quadruples. Each quadruple consists of a sentiment expression, a target expression, a holder expression and the polarity. SemEval 2022 Task 10 has two subtasks, monolingual and cross-lingual. We participated in the first track. Shared task data is consisted of seven datasets in English, Spanish, Norwegian, Catalan and Basque. We trained and evaluated our model on each of these datasets.[1]

2 Related Work

Structured sentiment analysis aims to extract sentiment, target and holder expressions along with their relations and also sentiment polarities. (Barnes et al., 2021) solves this problem by taking dependency graph parsing approach. It uses BiLSTM and multilingual BERT (Devlin et al., 2019) to encode input sentence and a neural dependency parser to jointly predict expressions and relations. Every sentiment, target or holder expression in the sentence would become a star-shaped subtree in the output graph, with last token of expression as its root and its edges labelled with type of entity. Relation between a sentiment expression and a target or holder expression is also represented with an edge from the former's root to that of the latter. This work evaluates its model on five datasets in four languages namely, NoReC$_{\text{Fine}}$ (Øvrelid et al., 2020), Multibooked$_{\text{EU}}$, Multibooked$_{\text{CA}}$ (Barnes et al., 2018), MPQA (Wiebe et al., 2005) and Darmstadt$_{\text{Unis}}$ (Toprak et al., 2010). Its main evaluation metrics are Targeted F1 and Sentiment Graph F1. First one measures exact prediction of target expression and polarity, and the second one measures exact match at graph level, weighting overlap between gold and predicted entities, averaged across three entity types.

A less comprehensive formulation of sentiment analysis is **End2End** sentiment analysis which aims to predict target and sentiment expression along with polarity but does not consider relation between expressions. (He et al., 2019) uses this formulation. Taking sequence labelling as its approach, it predicts target expression together with sentiment expression by one module and polarity by another one. Both modules along with a feature extraction module make use of a CNN to make their specific and shared latent vectors respectively. Furthermore, polarity prediction module has a self-attention layer which gets predicted probability of tokens' being in sentiment expression from the other module and incorporates it in computing attention values so as to tokens with higher proba-

[1] Our code is available at https://github.com/sadra-barikbin/novel-solutions-for-sentiment-analysis

bility get more attention. Beside its main task, this work employs two auxiliary document-level tasks to enjoy the benefits of multi-task learning. This work also uses a mechanism called message passing in which, predicted probabilities of all modules is iteratively fed back into model to make the model like a RNN. For data, they use review datasets from (Pontiki et al., 2014) and (Pontiki et al., 2015) which are annotated with sentiment expressions by (Wang et al., 2016, 2017). Among their evaluation metrics, F1-I is equivalent with Targeted-F1 of (Barnes et al., 2021).

(Chen and Qian, 2020), similarly attempts to solve end2end sentiment analysis and takes sequence labelling approach. It introduces a layer called RACL which consists of three separate but interconnected modules for sentiment and target extraction and polarity prediction. Each module makes a module-specific representation of input using a CNN and information is exchanged among them through attention mechanism. The work uses a stack of RACL layers as its model in which features in a layer is fed to upper layer and finally prediction is done by average pooling over prediction of all layers. This work has used same data and evaluation metrics as (He et al., 2019).

Finally (Li et al., 2019) solves **targeted sentiment analysis** which aims to extract only target expression and polarity. To this end, it has used BERT (Devlin et al., 2019) along with a task specific layer. Here BERT makes a contextual and rich representation of input words and the task-specific layer performs sequence labelling. For this layer, different types such as fully connected, CRF (Lafferty et al., 2001), GRU and self-attention layer are examined that the last two outperformed the others.

3 Datasets and Evaluation

Shared task is on seven datasets in five languages. We trained and evaluated our model on each one separately. General and detailed information of datasets is shown in table1 and table2 respectively. **NoReC**$_{\text{Fine}}$ is norwegian professional reviews in multiple domains. **MultiB**$_{\text{CA}}$, **MultiB**$_{\text{EU}}$, **Opener**$_{\text{EN}}$ and **Opener**$_{\text{ES}}$ (Agerri et al., 2013) are hotel reviews in Catalan, Basque, English and Spanish respectively. **Darmstadt**$_{\text{Unis}}$ consists of English online university reviews and finally **MPQA** contains annotated news articles in English. Evaluation metric of the task is ***Sentiment Graph F1***. Prediction and gold answer of problem are a list of quadroples $q = (q_s, q_t, q_h, q_{pol})$ in which the first three entities are sets of tokens for sentiment, target, and holder expressions respectively. q_pol is polarity with value among $Negative, Neutral, Positive$. Match score of two given source and target quadroples namely, $score(src, tgt)$ is as follows.

$$\frac{\sum_{e\in\{s,t,h\}} \frac{|src_e \cap tgt_e|}{|src_e|}}{3} * 1\{src_{pol} = tgt_{pol}\} \quad (1)$$

As it can be seen, this is a weighted match over amount of overlap between entities, averaged across three entity types. Polarities should be strictly equal. Denominator $|src_e|$ is for score to be comparable with another one having different source quadrople. As target and holder expressions could be empty, $|src_e|$ is replaced with 1 in the case src_e is empty. $|src_e \cap tgt_e|$ is also set to 1 when $|src_e|$ and $|tgt_e|$ are empty. Given N input sentences, $pred_n$ as the list of predicted quadroples for n-th sentence and $gold_n$ as its gold counterpart, ***Precision*** is computed as follows.

$$\frac{\sum_{n=1}^{N} \sum_{p\in pred_n} \max_{q\in gold_n} score(p, q)}{\sum_{n=1}^{N} |pred_n|} \quad (2)$$

max is to select score of best matching gold quadrople for a given predicted one. Similarly, ***Recall*** is computed as follows.

$$\frac{\sum_{n=1}^{N} \sum_{q\in gold_n} \max_{p\in pred_n} score(q, p)}{\sum_{n=1}^{N} |gold_n|} \quad (3)$$

Finally, ***Sentiment Graph F1*** is measured as follows.

$$GraphF1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

4 System Description

Our model is comprised of a base pre-trained model that is supposed to extract rich contextualized features and two separate modules for extracting expressions and predicting edges. For english datasets (Opener$_{\text{EN}}$, Darmstadt$_{\text{Unis}}$ and MPQA) we used RoBERTa$_{\text{base}}$ (Liu et al., 2019) model and for non-english ones (Opener$_{\text{ES}}$, Norec$_{\text{Fine}}$,

		#	avg.	max
NoReC$_{\text{Fine}}$	train	8634	21.3	125
	dev	1531	21.4	86
MultiB$_{\text{CA}}$	train	1174	18.7	238
	dev	167	16.3	129
MultiB$_{\text{EU}}$	train	1063	14.5	105
	dev	152	14.6	60
MPQA	train	5873	25.6	134
	dev	2063	25.5	228
Darmstadt$_{\text{Unis}}$	train	2253	21.6	226
	dev	232	19.5	80
Opener$_{\text{EN}}$	train	1744	15.6	131
	dev	249	15.2	129
Opener$_{\text{ES}}$	train	1438	19.3	175
	dev	206	18.9	108

Table 1: General information of shared task datasets. First five datasets are used in (Barnes et al., 2021). Max and average are measures of tokens' counts.

Multibooked$_{\text{CA}}$ and Multibooked$_{\text{EU}}$), we used LaBSE (Feng et al., 2020) as model's base.

We treat quadruples in a sentence as a graph which its nodes are expressions and its edges connect expressions within a quadruple. We name the module for extracting expression, **Node Extractor** and the module for determining edges, **Edge Predictor**. System structure is depicted in figure 1. Firstly, input sentence is fed into model base. Then node extractor gets the encoded input and for each one of three entity types, predicts expressions in the sentence. Polarities are also predicted by this module. Edge predictor, then, gets the expressions and for every pair of sentiment-target and sentiment-holder expressions, predicts if there is an edge between them. Finally a sentiment expression plus its polarity along with all of target and holder expressions connected thereto would become one of predicted quadroples.

4.1 Node Extractor

This module consists of three feed forward neural networks for predicting label of each token in BIO scheme, each network for an entity type. Using these three networks, sequence tokens are labeled for sentiment, target and holder expressions independently. In this networks, ReLU is used as activation and each network has less than 50k parameters. Loss function of the module is cross entropy. We integrate predicting polarity of sentiment expressions in sequence labelling task by replicating BIO labels for negative, neutral and positive polarities.

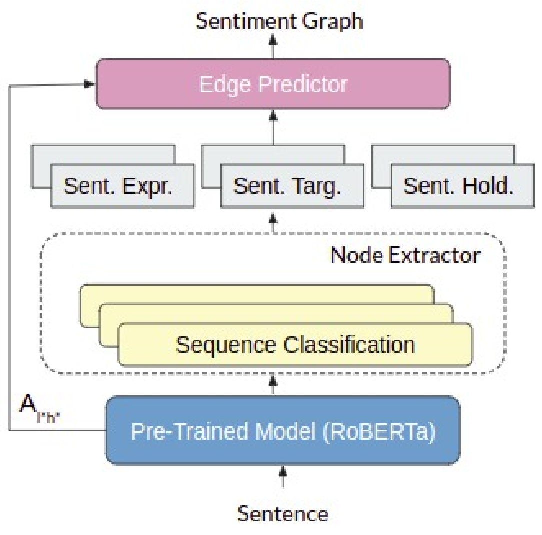


Figure 1: System architecture

Note that node extractor and jointly predicting sentiment expression and polarity is inspired of what is being used in one of our baselines, the one using sequence labelling approach.

4.2 Edge Predictor

Aimed for leveraging pretrained model's knowledge as much as possible, we simply used its computed attentions to predict edges. To this end, we examined different settings that are as follows:

Base: This is the base setting of edge predictor. We use attention values of a predetermined head in a specific layer in this way that we compute sum of attentions of two given expressions to each other and apply sigmoid to predict being an edge between them. More specifically, we choose head 7 in layer 8 of RoBERTa$_{\text{base}}$ for english datasets and head 9 layer 11 of LaBSE for others. This choice was based on the observation that those heads were more semantic-aware in the sense that sentiment expression tokens attend more to target expression tokens. Those heads are also less position dependant, meaning that a token's attention distribution does not lean toward a specific position in the sequence.

Consider two nodes a and b which span intervals (a_{begin}, a_{end}) and (b_{begin}, b_{end}) in the sentence respectively. Probability of being an edge being between a and b, $P_{e_{ab}}$ in the is computed as follows:

$$P_{e_{ab}} = \sigma\Big(\sum_{i=a_{begin}}^{a_{end}} \sum_{j=b_{begin}}^{b_{end}} A_{l^*h^*ij} + A_{l^*h^*ji}\Big) \quad (5)$$

		holders			targets			sent. expr.			polarity		
		#	avg.	max	#	avg.	max	#	avg.	max	+	neu	-
NoReC$_{\text{Fine}}$	train	898	1.3	18	6778	3.0	44	8448	6.8	51	5695	0	2753
	dev	120	1.1	4	1152	3.0	22	1432	7	38	988	0	444
MultiB$_{\text{CA}}$	train	169	1.3	4	1705	3.0	20	1989	3.2	21	1273	0	716
	dev	15	1.8	8	211	2.9	11	258	3.2	12	151	0	107
MultiB$_{\text{EU}}$	train	204	1.4	8	1277	2.3	14	1679	3.2	14	1401	0	278
	dev	33	1.5	5	152	2.3	9	203	3.6	13	167	0	36
MPQA	train	1425	3.2	43	1479	7.0	52	1706	2.2	16	671	337	698
	dev	405	3.3	28	494	6.1	44	570	2.2	9	231	124	215
Darmstadt$_{\text{Unis}}$	train	63	1.2	4	806	1.5	6	806	2.2	25	340	102	364
	dev	9	1.3	2	98	1.7	4	98	2.4	13	29	15	54
Opener$_{\text{EN}}$	train	266	1.0	3	2679	1.9	11	2884	2.6	18	2101	0	783
	dev	49	1.0	2	371	2.0	10	400	2.6	14	284	0	116
Opener$_{\text{ES}}$	train	176	1.0	3	2748	2.4	13	3042	2.6	16	2472	0	570
	dev	23	1.0	2	363	2.6	11	387	2.6	17	317	0	70

Table 2: Details of shared task datasets. First five datasets are used in (Barnes et al., 2021). Max and average are measures of tokens' counts in expressions.

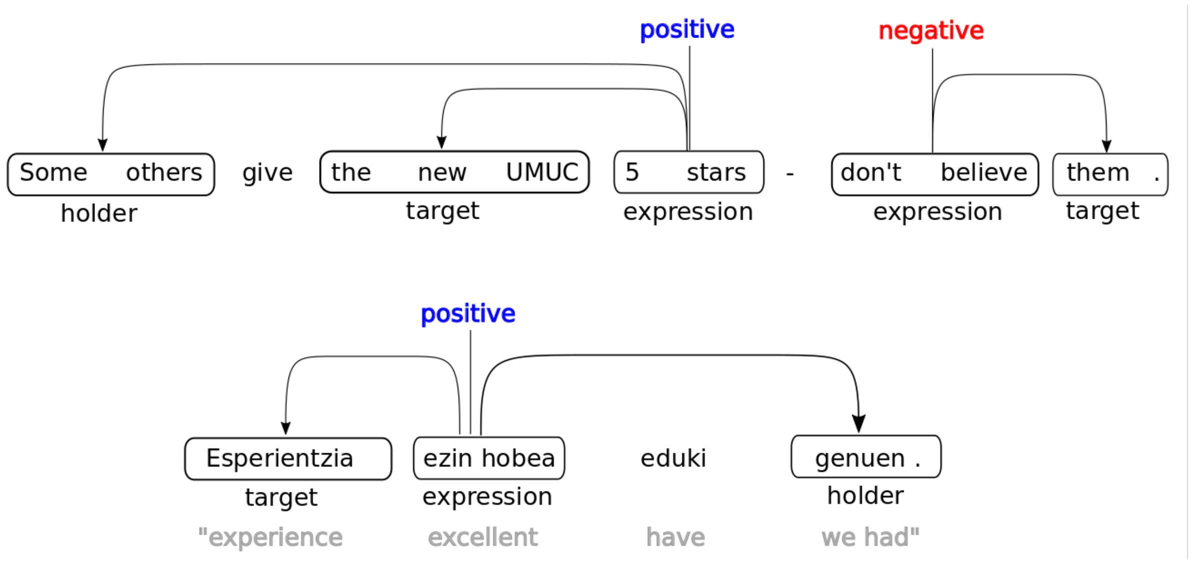


Figure 2: A sentiment graph example. Entities of a quadrople are connected with edges and sentiment expressions are labeled with polarity. Image is from (Barnes et al., 2022)

A is the pretrained model's computed attention. l^* and h^* determine prespecified head and layer respectively. σ is the sigmoid function. This equation simply means attention of two nodes to each other determines probability of being an edge between them.

+AvgH: In this setting we do not rely only on specific heads and use attention values of all heads in all layers. To this end we introduce new learnable weights, one for each head, and compute linear combination of attention values across all heads. Total number of new weights is number of layers times number of heads in each layer, which is 144 in both RoBERTa$_{\text{base}}$ and LaBSE. So in this setting, $P_{e_{ab}}$ becomes

$$\sigma(\sum_{l,h=1}^{n_l,h_l} \sum_{i=a_{begin}}^{a_{end}} \sum_{j=b_{begin}}^{b_{end}} w_{lh} * (A_{lhij} + A_{lhji})) \quad (6)$$

in which n_l and n_h are number of layers and heads in the pretrained Model respectively. w is the learned weight assigned to each head. At training start w is set to 1 in setting one head and 0 in other heads.

+SepH: We use separate heads for predicting edges for sentiment-target expression pair and sentiment-holder expression pair. Specifically, for predicting and edge between a sentiment and a target expression, we use head 7 in layer 8 for english datasets and head 9 in layer 11 for non-english

	$\text{Opener}_{\text{EN}}$	$\text{Opener}_{\text{ES}}$	$\text{Norec}_{\text{Fine}}$	$\text{Darmstadt}_{\text{Unis}}$	$\text{MultiB}_{\text{EU}}$	$\text{MultiB}_{\text{CA}}$	MPQA	Average
base	0.64	0.65	0.44	0.45	0.67	0.64	0.38	0.55
+AvgH	-0.03	-0.02	-0.05	-0.14	0.02	0.01	-0.31	-0.07
+SepH	0.03	0.02	-0.05	-0.03	-0.00	-0.02	-0.08	-0.02
+SepH+AvgH	0.01	-0.00	-0.03	-0.07	-0.02	-0.00	-0.24	-0.05

Table 3: Performance of different variants of our model. First row is the default variant and numbers in other rows show increase or decrease in comparison with the default variant. Results are based on evaluation on development data.

ones. For predicting an edge between a sentiment expression and a holder expression, we use head 10 in layer 11 for english datasets and head 4 in layer 11 for non-english ones.

+SepH+AvgH: This is the mixture of last two settings. We use all heads in all layers but we consider separate linear combinations of them for sentiment-target and sentiment-holder expression pairs. So the number of introduced weights is doubled to 288 in this setting.

5 Experiments

5.1 Baselines

We compare our model with two baselines proposed in the shared task. First one is basically (Barnes et al., 2021) which is described in section 2. The second one takes sequence labelling approach. It sets two modules for predicting expressions and relations respectively. Expression prediction module consists of three BiLSTMs which predict sentiment, target and holder expressions respectively. This module is fed by an embedding layer which is initialized with pretrained word embeddings. Relation prediction module, given two extracted expressions and the input sentence, uses separate BiLSTMs and max pooling to make contextualized representations from them. Then it feeds concatenation of three max pooling layers to a linear layer and sigmoid to predict if there is relation between two expressions or not. In the comparisons we call the baselines, **GP** (for graph parsing) and **SL-BiLSTM** (for sequence labelling) respectively.

5.2 Settings

We trained each variant of our model 5 times, each with different random seed on every dataset for at most 20 epochs. learning rate was 1e-4 and 1e-3 for pretrained model and newly introduced weights respectively. We did warm-up in epoch 1 and applied step LR scheduling with gamma as .1 and step size as 9. Baselines were also trained using script given by shared task organizers. Evaluation measure being used was Sentiment Graph F1. Evaluation was done using the script given by shared task organizers. We used PyTorch (Paszke et al., 2019) plus Pytorch-Ignite (Fomin et al., 2020) as training framework and WandB (Biewald, 2020) as Experiment Tracking tool.

Dataset	Attentionist	GP	SL-BiLSTM
$\textbf{NoReC}_{\text{Fine}}$	**0.32**	0.28	0.19
$\textbf{MultiB}_{\text{CA}}$	**0.61**	0.54	0.33
$\textbf{MultiB}_{\text{EU}}$	**0.59**	0.57	0.34
MPQA	**0.26**	0.15	0.02
$\textbf{Darmstadt}_{\text{Unis}}$	**0.25**	0.21	0.13
$\textbf{Opener}_{\text{EN}}$	**0.58**	0.52	0.31
$\textbf{Opener}_{\text{ES}}$	**0.55**	0.50	0.26

Table 4: Test results of our model and baselines. Evaluation metric is Sentiment Graph F1

6 Results

Results are shown in table4. **Attentionist** is our proposed model. For our model, only result of the best variant is shown. As it can be seen, our model outperforms the baseline in all datasets.

7 Ablation Study

In order to assess and compare performance of different variants of our model, we trained each variant four times using different random seeds on each dataset and compared them on development data. Results are depicted in table3.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

A.H. Alamoodi, B.B. Zaidan, A.A. Zaidan, O.S. Albahri, K.I. Mohammed, R.Q. Malik, E.M. Almahdi, M.A. Chyad, Z. Tareq, A.S. Albahri, Hamsa Hameed, and Musaab Alaa. 2021. Sentiment analysis and its applications in fighting covid-19 and infectious diseases: A systematic review. *Expert Systems with Applications*, 167:114155.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. *CoRR*, abs/2105.14504.

Jeremy Barnes, Oberländer Laura Ana Maria Kutuzov, Andrey and, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694, Online. Association for Computational Linguistics.

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

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT sentence embedding. *CoRR*, abs/2007.01852.

V. Fomin, J. Anmol, S. Desroziers, J. Kriss, and A. Tejani. 2020. High-level library to help with training neural networks in pytorch. `https://github.com/pytorch/ignite`.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. *CoRR*, abs/1906.06906.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019. Exploiting BERT for end-to-end aspect-based sentiment analysis. *CoRR*, abs/1910.00883.

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

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Margarita Rodríguez-Ibáñez, Francisco-Javier Gimeno-Blanes, Pedro Manuel Cuenca-Jiménez, Cristina Soguero-Ruiz, and José Luis Rojo-Álvarez. 2021. Sentiment analysis of political tweets from the 2019 spanish elections. *IEEE Access*, 9:101847–101862.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages

616–626, Austin, Texas. Association for Computational Linguistics.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

LyS_ACoruña at SemEval-2022 Task 10: Repurposing Off-the-Shelf Tools for Sentiment Analysis as Semantic Dependency Parsing

Iago Alonso-Alonso, David Vilares and Carlos Gómez-Rodríguez
Universidade da Coruña, CITIC
Departamento de Ciencias de la Computación y Tecnologías de la Información
Campus de Elviña s/n, 15071
A Coruña, Spain
{iago.alonso,david.vilares,carlos.gomez}@udc.es

Abstract

This paper addressed the problem of structured sentiment analysis using a bi-affine semantic dependency parser, large pre-trained language models, and publicly available translation models. For the monolingual setup, we considered: (i) training on a single treebank, and (ii) relaxing the setup by training on treebanks coming from different languages that can be adequately processed by cross-lingual language models. For the zero-shot setup and a given target treebank, we relied on: (i) a word-level translation of available treebanks in other languages to get noisy, unlikely-grammatical, but annotated data (we release as much of it as licenses allow), and (ii) merging those translated treebanks to obtain training data. In the post-evaluation phase, we also trained cross-lingual models that simply merged all the English treebanks and did not use word-level translations, and yet obtained better results. According to the official results, we ranked 8th and 9th in the monolingual and cross-lingual setups.

1 Introduction

Sentiment Analysis (SA, Pang and Lee, 2008) deals with the automatic processing of subjective information in natural language texts. Early work on SA focused on conceptually simpler tasks, such as polarity classification at the sentence or document level. With the advances in natural language processing (NLP), more fine-grained and complex tasks have been proposed, such as detecting the entity that expresses an opinionated chunk of text, or the entity that was targeted. More particularly, Barnes et al. (2021) consider sentiment analysis as a (graph) structured task, and discuss up to five subtasks: (i) sentiment expression extraction, (ii) sentiment target extraction, (iii) sentiment holder extraction, (iv) defining the relationship between these elements, and (v) assigning a polarity label. They discuss that although these tasks have been extensively studied by different authors (Turney, 2002; Pontiki et al., 2015; Zhang et al., 2019, *inter alia*), they are not addressed all together. They also discuss that such subdivision into subtasks might have a negative impact in the general analysis of the sentence, and that a joint analysis could translate into a holistic approach. To do so, they propose to encapsulate all these tasks in the form of a sentiment graph. Formally, the goal is to find the set of opinion tuples $\{O_1, \ldots, O_i, \ldots, O_n\}$ in a given text, where each opinion O_i is a tuple of the form (h, t, e, p) where h is a holder who expresses a polarity p towards a target t through a sentiment expression e, implicitly defining pairwise relationships between elements of the same tuple. We illustrate an example in Figure 1.

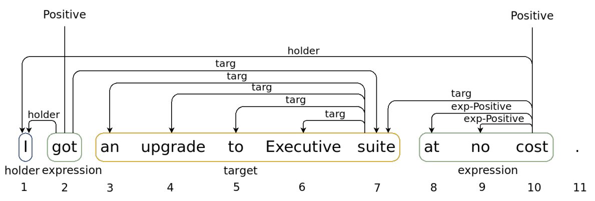


Figure 1: An example of sentiment graph as defined by Barnes et al. (2021). The sentence has a holder ('I'), two sentiment expressions ('got' and 'at no cost') and one target ('an upgrade to Executive suite')

More particularly, for the SemEval-2022 Task 10 (Barnes et al., 2022), the organizers proposed both a monolingual[1] and a cross-lingual (zero-shot) setup. They considered 5 languages (and 7 treebanks): English, Spanish, Catalan, Basque, and Norwegian. For the zero-shot setup Basque, Catalan, and Spanish were the target languages.

Our approach is based on the idea of viewing this task as semantic dependency parsing (Oepen et al., 2015), since both tasks are structurally similar even if the graphs have different meaning. More

[1]We use the term monolingual as it was the term used by the organizers, but this setup allowed the use of any resource, including resources in different languages.

specifically, we rely on a bi-affine graph-based parser (Dozat and Manning, 2018) and different large pre-trained language models (LM), such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) or XLM-R (Conneau et al., 2020). For the monolingual setup we train a semantic parsing model on single and merged treebanks, and compare the performance using different LMs. For the cross-lingual setup, we first do a word-level translation of the datasets in a different language than the target treebank, and then proceed similarly to the monolingual setup. Overall, the approach relies on off-the-shelf tools already available, but traditionally used for other purposes. We here re-purpose them for their use for sentiment analysis as graph-based parsing.

2 The role of parsing in SA

Parsing has been used in the past for SA, with different motivations, such as integrating syntactic knowledge as a component of the model's architecture, or producing structured sentiment outputs.

Polarity classification. Since early times, authors have studied the importance of language structure to deal with relevant linguistic phenomena for polarity classification, first focusing on simpler strategies such as the use of n-grams or lexical rules (Pang et al., 2002; Taboada et al., 2011). Later on, more complex syntactic structures were incorporated as well, both for rule-based and machine learning approaches.

For instance, for the rule-based paradigm, Poria et al. (2014) used dependency relations for concept-level sentiment analysis, so sentiment could flow from one concept to another to better contextualize polarity. Vilares et al. (2015a, 2017) proposed a model to compute the sentiment of sentences that was driven by syntax-based rules to deal with specific relevant phenomena in SA, and that could be easily re-purposed for any language for which a dependency parser was available. Kanayama and Iwamoto (2020) built on top of Vilares et al.'s idea, and proposed a multilingual syntax-based system that achieved a high precision for 17 languages.

From the machine learning perspective, Joshi and Rosé (2009); Vilares et al. (2015b) used dependency triplets to train data-driven (pre-neural) models and obtain slight improvements over purely lexical approaches. Socher et al. (2013) collected sentiment labels for phrases and sentences that were previously automatically represented as constituent (sub)trees, to then train a compositional model that used a recursive neural network. This work has some relevant resemblances with Barnes et al. (2021)'s proposal for structured sentiment analysis. Socher et al. were among the first to provide tree-shaped annotated sentiment data (in this case just for polarity classification), while most of previous work had focused on using tree knowledge as external information to the models, but with sentiment annotations only associated with plain texts. This publicly available data later encouraged many authors to design models that could exploit tree-shaped annotated data to obtain better performing models (Tai et al., 2015; Zhang and Zhang, 2019, *inter alia*).

Aspect-based sentiment analysis (ABSA). ABSA is a task that is particularly suitable for the integration of syntactic information, since its main goal is to associate sentiment with specific entities and aspects that occur in the sentence (Pontiki et al., 2015). Related to this, Popescu and Etzioni (2005) already used dependency trees to constrain an unsupervised sentiment analysis system that extracted a set of product features and their sentiment, given a particular item. More recently, with the wide adoption of neural networks in NLP, different authors have integrated syntactic knowledge and syntactic structures in different network architectures, such as long short-term memory networks (LSTMs, Tang et al., 2016), recursive neural networks (Nguyen and Shirai, 2015), convolutional networks (Xue and Li, 2018), and graph attention networks (Huang et al., 2020; Sun et al., 2019).

Overall, it is clear that parsing has had a high relevance in SA. Yet, the novelty of the shared task is in using graphs to represent richer annotations. This makes it possible to use parsing algorithms as sentiment models, i.e. not just to use them as a component of the model architecture, but as the model responsible of producing the whole sentiment structure of the chunk of text. Also, this is especially relevant in the era of large neural models, where the utility of parsers for downstream tasks is sometimes questioned, with some studies questioning its need in the presence of pretrained models that implicitly learn syntax (Tenney et al., 2019; Glavaš and Vulić, 2021; Dai et al., 2021) while others still achieve extra accuracy from their use in conjuntion with such models (Sachan et al., 2021; Xu et al., 2021; Li et al., 2021; Zhang et al., 2022). In any case, tasks like this one show that graph structures can be also useful to re-purpose traditional tasks such as SA, while taking advantage of research that the NLP community has done on parsing algorithms for decades.

3 Brief overview of the shared task

The goal of the task is to produce graph structures that reflect the sentiment of a sentence, as we showed in Figure 1. More particularly, the organizers released 7 treebanks in 5 different languages: OpeNER (Agerri et al., 2013, English and Spanish), MPQA (Wiebe et al., 2005, English), Darmstadt_unis (Toprak et al., 2010, English), MultiBooked (Barnes et al., 2018, Basque and Catalan), and NoReC_fine (Øvrelid et al., 2020, Norwegian).[2] Table 1 details the main statistics for the datasets.

Dataset	Language	# sents	# holders	# targets	# expr.
NoReC_fine	Norwegian	11437	1128	8923	11115
MultiBooked	Basque	1521	296	1775	2328
MultiBooked	Catalan	1678	235	2336	2756
OpeNER	Spanish	2057	255	3980	4388
OpeNER	English	2494	413	3850	4150
MPQA	English	10048	2279	2452	2814
Darmstadt_unis	English	2803	86	1119	1119

Table 1: General statistics of the treebanks used in the shared task.

The sentiment of a sentence is composed of all the opinions, O_i, that make it up. Each opinion can have up to four elements: a holder (h) who expresses a polarity (p) towards a target (t) through a sentiment expression (e). These four elements implicitly define the pairwise relationships between the elements of a tuple. The previous example, Figure 1, shows a sentence with two sentiment expressions (*got* and *at no cost*) that express the polarity (*Positive*) of the sentiment that a holder (I) has towards one target (*an upgrade to Executive suite*).

Preprocessing The organizers of the shared task proposed two possible ways to address the task: as a sequence labeling or as graph-based parsing problem. As mentioned above, we opted for the latter. We use the scripts available in the official repository to transform the JSON files to the CoNLL-U based format and *vice versa*, and we applied the needed changes to make it compatible with `supar` (see §4).[3] Under the graph-based paradigm, the problem is approached as a bilexical dependency graph prediction task, with some assumptions. To convert the data, the organizers suggest two possible conversions, namely *head-first* and *head-final*. In *head-first*, it is assumed that the first token of the sentiment expression is a root node, and that the first token of each holder or target spans is the head node of such span, while the other ones are dependents. Meanwhile, in *head-final*, the final token of the holder and target spans is set as the head of the span, and the final token of the sentiment expression as a root node (Figure 1 is a head-final example). In this work, we have chosen *head-final*, which is the default option for the shared task and also delivered better results than *head-first* in the experiments carried out by Barnes et al. (2021) (see Table 3 in that paper).

Subtasks More in detail, the challenge is divided into two subtasks:

1. Monolingual setup: When training and development data is available for the same treebank/language, i.e. the goal is to train one model per treebank. It was allowed to use extra resources or tools that could boost performance, even from different languages.

2. Cross-lingual, zero-shot setup: It is assumed that there is no gold training data in the language of the target treebank. The organizers specified that it is possible to use treebanks in other languages, translation tools, and any other resources that do not include sentiment annotations in the target language.

Metrics Each subtask is evaluated independently, and the ranking metric was *sentiment graph F1* (Barnes et al., 2021), where true positives are exact matches at the graph level, weighting the overlap between the predicted and gold spans for each element, and averaged across all three spans. To compute precision, it weights the number of correctly predicted tokens divided by the total number of predicted tokens, while for recall it weights the number of correctly predicted tokens divided by the total number of gold tokens. Also, as mentioned earlier, it is possible to have tuples with empty holders and targets.

[2]For more detailed information see `https://github.com/jerbarnes/semeval22_structured_sentiment`

[3]`https://github.com/MinionAttack/conllu-conll-tool`

4 Our model

We rely on the Dozat and Manning (2018) parser, a widely used state-of-the-art model both for syntactic and semantic dependency parsing. Inspired in previous graph-based parsers (McDonald et al., 2005; Kiperwasser and Goldberg, 2016), the parser first computes contextualized representations for each word using bidirectional LSTMs (biLSTMs; Hochreiter and Schmidhuber, 1997; Schuster and Paliwal, 1997). After that, the model computes a *head* and a *dependent* representation for each term, to establish through a bi-affine attention whether an edge exists between each pair of tokens, and if so, what is the semantic relationship between them. In particular, in this paper we follow the implementation used in the `supar`[4] package, as it has been widely adopted by the community and it is available for other flavors of parsing as well, such as constituent or dependency parsing. We preferred this implementation over the graph-based baseline provided in the SemEval repository, since early experiments showed a superior performance, and it also offered a simpler integration of large language models. We left the parser hyperparameters, except the learning rate, at their default value.

Pre-trained language models For each language we looked for avaliable monolingual and multilingual pre-trained LMs at `https://huggingface.co/`. Specifically, for each language, we included:

- `Basque`: berteus-base-cased, RoBasquERTa.
- `Catalan`: julibert, roberta-base-ca, calbert-base-uncased.
- `English`: bert-base-cased, bert-base-uncased, bert-large-cased, bert-large-uncased, roberta-base, roberta-large, albert-base-v2, albert-large-v2, xlnet-base-cased, xlnet-large-cased, electra-base-discriminator, electra-large-discriminator, electra-base-generator, electra-large-generator.
- `Norwegian`: norbert, nb-bert-base, nb-bert-large, electra-base-norwegian-uncased-discriminator.
- `Spanish`: bio-bert-base-spanish-wwm-uncased, bert-base-spanish-wwm-cased, roberta-base-bne, roberta-large-bne, selectra_medium, zeroshot_selectra_medium.

[4]`https://github.com/yzhangcs/parser`

With respect to the cross-lingual LMs, we considered: xlm-roberta-base and xlm-roberta-large.

4.1 Monolingual models

For this task, we use: (i) pre-trained language models, (ii) `supar`, and (iii) the official training and development files to build our models. Also note that we train end-to-end models, using words as the only input (later tokenized into subword pieces by the language models), but ignoring the part-of-speech tags and syntactic information provided in the sentiment treebanks. We did not use part-of-speech tags (or other morphosyntactic annotations) since these are not used in `supar` together with BERT encoders, and using them would require to adapt the code, which was exactly what we tried to avoid in this work.

Training procedure We fine-tuned parsing models considering for each treebank the proposed LMs, and combining them with `supar`. Since training is time-consuming, many model configurations are proposed, and the performance of `supar` is stable independently of the seed, we decided to train a single model per LM. Specifically, all models have been trained with the default seed used by `supar`, which is 1. The only parameter that was modified was the learning rate (l_r), as we observed that for some models (specially the larger language models) the fine-tuning process did not converge. We started with $5 \cdot 10^{-5}$, and did a small grid search down to $1 \cdot 10^{-6}$, where if a model still did not converge it was discarded.[5] Additionally, to train the parsing models, we considered three strategies:

1. Single monolingual training and development files: We train each model on a single treebank and validate its performance in the corresponding dev set, i.e., the standard monolingual training and development methodology.

For the best model obtained for each treebank according to strategy 1, we explored a couple of harmonized training strategies (harmonized in the sense that different treebanks follow the same annotation guidelines):

2. Merged training and development files from different treebanks: We considered to merge

[5]For both monolingual and cross-lingual subtasks, all the selected models used a l_r of $5 \cdot 10^{-5}$. The only exception is Norwegian in the monolingual subtask, for which we used $5 \cdot 10^{-6}$.

all the available training files and all the available development files, treating them as a single dataset. Thus, we trained a single model that could predict all test files, but with the disadvantage that model selection is based on multilingual performance, which could hurt the performance in this setup.

3. Merged training files, single development file: Similar to 2, but merging only the training files. For the development phase, we proceeded as in 1 and used each dataset's dev file for model selection. The idea was to have training data that can benefit from multilingual information, but that still considers a monolingual file for a language-dependent model selection, i.e., given n treebanks, we still need to train n models, one per treebank.

We detail the experimental results for the training/development phase in §5.

4.2 Cross-lingual (zero-shot) models

In this setup, we rely on two main components: (i) available translation systems to perform word-level translations from source language to target language treebanks, and (ii) both monolingual and cross-lingual language models. Our goal with (i) is to obtain noisy, unlikely-grammatical data, but that still can provide sentiment annotations for a given target language, exploring the viability of this approach. Regarding the learning rate, we used $5 \cdot 10^{-5}$ in all cases.

Auxiliary translation models From the CoNLLU converted files[6], we translated the sentences at the word level using the Helsinki-NLP translation models[7] (Tiedemann and Thottingal, 2020) available at `huggingface`. Table 2 lists the language pairs for which we could obtain translated versions for the cross-lingual setup.

Dataset	Language	Basque	Catalan	Spanish
NoReC_fine	Norwegian			✓
MultiBooked	Basque			✓
MultiBooked	Catalan			✓
OpeNER	Spanish	✓	✓	
OpeNER	English	✓	✓	✓
MPQA	English	✓	✓	✓
Darmstadt_unis	English	✓	✓	✓

Table 2: Treebanks and the languages to which they were translated for the cross-lingual experiments.

[6] https://github.com/MinionAttack/corpus-translator
[7] https://huggingface.co/Helsinki-NLP

Then, to train the models we proceeded similarly to strategy 2 used in the monolingual setup: we combined the translated training and validation files coming from treebanks in other languages, and used the micro-averaged F1-score on the translated development set for model selection.

Post-evaluation (and better) baseline After the deadline to submit proposals, we also tested a baseline consisting on training, using an XLM-RoBERTa LM as the base component, a cross-lingual model that uses all the English datasets (without any kind of translation) as the source data. We discuss these results as well in §5.2.

5 Results

Here, we detail and discuss the results that we got for both subtasks (see §5.1 and 5.2) on: (i) the official development sets, and (ii) the official test sets of the shared tasks.

5.1 Monolingual setup

Tables 3 and 4 show the results for the development phase on the English and non-English datasets, respectively, including different LMs and training setups.

With respect to the results on the English treebanks, an interesting trend is that despite being the monolingual setup, using cross-lingual language models, and in particular XLM-RoBERTa, performed surprisingly well. Combined with the training strategy 3 (merged training sets, single development set), such models obtained the best results for 2 out of 3 English corpora (OpeNER and Darmstadt), while they still ranked well in the other dataset (MPQA). Across monolingual LMs, we also observe trends: electra-base-discriminator and (both base and large) RoBERTa models obtain overall the best results. On the other hand, we did not obtain equally robust results with ALBERT, and to a lesser extent, with BERT architectures. This is not totally surprising, since among the tested LMs, BERT is among the oldest ones, and ALBERT is a lite BERT, so some computational power is lost and it is understandable that this translates into some performance loss too, compared to larger LMs.

With respect to the experiments on the non-English datasets, we observe certain similarities, although the number of available models is much smaller than in the English cases. Again, XLM-RoBERTa overall obtains the best results. The only

Corpus	Model	Strategy	F1
OpeNER_en	xlm-roberta-large	3	0.714
	electra-base-discriminator	1	0.710
	xlm-roberta-large	1	0.707
	xlm-roberta-base	2	0.686
	roberta-large	1	0.683
	xlnet-base-cased	1	0.681
	xlm-roberta-base	3	0.679
	xlm-roberta-base	1	0.673
	roberta-base	1	0.663
	electra-large-discriminator	1	0.662
	bert-large-uncased	1	0.660
	electra-large-generator	1	0.652
	xlm-roberta-large	2	0.643
	bert-base-uncased	1	0.640
	bert-large-cased	1	0.640
	xlnet-large-cased	1	0.639
	bert-base-cased	1	0.612
	electra-base-generator	1	0.612
	albert-base-v2	1	0.590
	albert-large-v2	1	0.297
MPQA	roberta-base	1	0.374
	roberta-large	1	0.365
	electra-base-discriminator	1	0.351
	xlm-roberta-large	1	0.346
	xlnet-base-cased	1	0.338
	xlm-roberta-large	3	0.327
	bert-base-cased	1	0.306
	xlm-roberta-base	2	0.303
	electra-large-generator	1	0.301
	xlm-roberta-large	2	0.298
	bert-large-uncased	1	0.297
	bert-base-uncased	1	0.294
	xlm-roberta-base	3	0.285
	xlm-roberta-base	1	0.277
	bert-large-cased	1	0.269
	electra-base-generator	1	0.253
	albert-base-v2	1	0.236
	xlnet-large-cased	1	0.209
Darmstadt_unis	xlm-roberta-large	3	0.329
	xlm-roberta-large	1	0.309
	electra-base-discriminator	1	0.306
	xlm-roberta-base	3	0.306
	xlm-roberta-large	2	0.301
	roberta-base	1	0.301
	xlm-roberta-base	1	0.276
	xlnet-base-cased	1	0.276
	xlnet-large-cased	1	0.269
	roberta-large	1	0.268
	electra-large-generator	1	0.267
	electra-large-discriminator	1	0.264
	xlm-roberta-base	2	0.262
	bert-large-uncased	1	0.257
	bert-base-uncased	1	0.251
	bert-large-cased	1	0.237
	electra-base-generator	1	0.229
	albert-base-v2	1	0.217
	bert-base-cased	1	0.212

Table 3: Scores on the development set for the English treebanks and the monolingual setup. Models trained on the training data before its updated version.

exception is the Norwegian dataset, where we obtained the best results with a BERT architecture.

Yet, a more thoughtful discussion would be needed to determine if some architectures truly behave better than others. Note that all these LMs are usually pre-trained using different and heterogeneous text sources, and specially for the less-resourced languages, some constraints are usually imposed during training. For instance, it is hard to conclude that berteus-base-cased (BERT) (Agerri et al., 2020) is worse than XLM-RoBERTa (Conneau et al., 2020), since the amount of resources to

Corpus	Model	Strategy	F1
NoReC_fine	nb-bert-large	1	0.479
	nb-bert-base	1	0.459
	xlm-roberta-large	1	0.450
	xlm-roberta-large	3	0.439
	xlm-roberta-large	2	0.427
	xlm-roberta-base	3	0.414
	xlm-roberta-base	2	0.411
	xlm-roberta-base	1	0.401
	electra-base-norwegian -uncased-discriminator	1	0.382
	norbert	1	0.298
MultiBooked_eu	xlm-roberta-large	3	0.662
	xlm-roberta-large	2	0.623
	xlm-roberta-base	3	0.613
	berteus-base-cased	1	0.602
	xlm-roberta-base	2	0.597
	xlm-roberta-base	1	0.571
	xlm-roberta-large	1	0.569
	RoBasquERTa	1	0.496
MultiBooked_ca	xlm-roberta-base	1	0.694
	xlm-roberta-large	1	0.683
	xlm-roberta-large	2	0.679
	xlm-roberta-large	3	0.679
	xlm-roberta-base	2	0.674
	roberta-base-ca	1	0.672
	xlm-roberta-base	3	0.653
	julibert	1	0.590
	calbert-base-uncased	1	0.579
OpeNER_es	xlm-roberta-large	3	0.666
	xlm-roberta-base	2	0.662
	xlm-roberta-base	3	0.657
	xlm-roberta-large	2	0.639
	xlm-roberta-base	1	0.635
	xlm-roberta-large	1	0.635
	bert-base-spanish-wwm-cased	1	0.630
	selectra_medium	1	0.622
	roberta-base-bne	1	0.616
	zeroshot_selectra_medium	1	0.610
	roberta-large-bne	1	0.605
	bio-bert-base-spanish-wwm-uncased	1	0,457

Table 4: Scores on the development set for the non-English treebanks and the monolingual setup. Models trained on the training data before its updated version.

train the former was more constrained.

Finally, a few days before the submission deadline, the training files of some treebanks were slightly updated by the organizers, due to minor bugs in the segmentation process that corrupted some sentences. As we did not have time to re-run all models and update the results, we chose to re-train only the model that obtained the best performance on the previous version of the treebanks. Therefore, all the outputs submitted for the test sets correspond to models trained on the updated, uncorrupted files. In Table 5 we compare the performance of the models trained on the updated and deprecated versions of the training files. Overall, we observed relatively small, but non-negligible differences, usually obtaining a better performance with the updated version of the treebank.

Official results on the test sets In Table 6 we show the performance on the test sets of our submitted models, i.e. those that achieved the highest score in the corresponding development phase. The performance is stable across different test sets, obtaining slightly better results for Iberian languages. For a detailed comparison against the rest of participants, we refer the users to Appendix 11 and the

Corpus	Model	Strategy	Old F1	New F1
NoReC_fine	nb-bert-large	1	0.479	0.492
MultiBooked_eu	xlm-roberta-large	3	0.662	0.648
MultiBooked_ca	xlm-roberta-base	1	0.694	0.699
OpeNER_es	xlm-roberta-large	3	0.666	0.709
OpeNER_en	xlm-roberta-large	3	0.714	0.716
MPQA	roberta-base	1	0.374	0.374
Darmstadt_unis	xlm-roberta-large	3	0.329	0.357

Table 5: Scores on the development set for the models trained on the corrupted and uncorrupted versions of the training files, on the monolingual setup. For each treebank, we only did the comparison for the best performing model, based on the performance on the corrupted version.

Dataset	Model	Strategy	Score
NoReC_fine	nb-bert-large	1	$0.462_{(10)}$
MultiBooked_eu	xlm-roberta-large	2	$0.680_{(7)}$
MultiBooked_ca	xlm-roberta-base	1	$0.653_{(8)}$
OpeNER_es	xlm-roberta-large	3	$0.692_{(6)}$
OpeNER_en	xlm-roberta-large	3	$0.698_{(9)}$
MPQA	roberta-base	1	$0.349_{(10)}$
Darmstadt_unis	xlm-roberta-large	3	$0.414_{(8)}$

Table 6: Scores of our models, for the monolingual subtask, on each test set. Our ranking on the shared task for each test set is indicated as a subscript.

official shared task paper (Barnes et al., 2022).

The datasets of the shared task belong to different domains: OpeNER and MultiBooked deal with hotel reviews, NoReC with professional reviews in multiple domains, Darmstadt_unis (the dataset for which we obtain the second lowest scores) contains English online university reviews, and MPQA (the dataset for which we obtain the lowest scores) is about news articles annotated with opinions and other private states (i.e., beliefs, emotions, sentiments, speculations, . . .). For the two lowest-scoring datasets, they have in common that they mostly contain single-opinion sentences, whereas the other datasets tend to have more variety in the number of opinions and their distribution. For instance, ∼85% and ∼74% of the training sentences of the Darmstadt_unis and MPQA datasets have only one opinion, while the next most 'single-opinion' dataset is multibooked_eu with only ∼53% of the sentences. However, we need to perform more detailed analysis as future work to extract more robust conclusions.

5.2 Cross-lingual setup

Table 7 shows the results for the development phase for the three target languages and their datasets. Again, XLM-RoBERTa models obtain overall the best performance, although in this case it is less surprising since cross-lingual LMs are expected to suit well this kind of challenges. Similar to the case of the monolingual setup, we decided to retrain the best-performing model with the updated versions of the training files. In Table 8, we show the comparison between the corrupted and uncorrupted versions of the datasets, which contrarily to the monolingual setup, often turned out into worse performing models.

Finally, Table 9 shows the scores for the post-evaluation baseline (model trained on the English datasets with XLM-RoBERTa) on the dev set. Very interestingly, the results show that this baseline outperformed our word-level translation approaches. We need more analysis to understand why this happens, but we hypothesize that the larger amount of English texts XLM-RoBERTa was pre-trained on could be playing an important role.

Corpus	Model	F1
Basque	xlm-roberta-base	0.434
	berteus-base-cased	0.416
	RoBasquERTa	0.323
Catalan	roberta-base-ca	0.564
	xlm-roberta-base	0.519
	julibert	0.486
	calbert-base-uncased	0.385
Spanish	xlm-roberta-base	0.605
	xlm-roberta-large	0.593
	bert-base-spanish-wwm-cased	0.583
	zeroshot_selectra_medium	0.555
	selectra_medium	0.536
	roberta-base-bne	0.515
	roberta-large-bne	0.438
	bio-bert-base-spanish-wwm-uncased	0.386

Table 7: Scores on the development set for the *translated* English treebanks and the cross-lingual setup. Models trained on the training data before its updated version.

Language	Model	Old F1	New F1
Basque	berteus-base-cased	0.416	0.424
	xlm-roberta-base	0.434	0.416
Catalan	roberta-base-ca	0.564	0.572
Spanish	xlm-roberta-large	0.593	0.570
	xlm-roberta-base	0.605	0.569

Table 8: Scores on the development set for the models trained on the corrupted and uncorrupted versions of the *translated* training files, on the cross-lingual setup. For each treebank, we only did the comparison for the best performing model, based on the performance on the corrupted version.

Official results on the test sets Finally, in Table 10 we show our results on test sets of the cross-lingual, zero-shot setup, for which we obtain again stable results. Appendix 12 contains the results for all participants.

Language	Model	Score
Basque	xlm-roberta-base	0.678
	xlm-roberta-large	0.677
Catalan	xlm-roberta-base	0.598
	xlm-roberta-large	0.625
Spanish	xlm-roberta-base	0.663
	xlm-roberta-large	0.638

Table 9: Scores on the development set of the trained English models (trained on MPQA, OpeNER_en and Darmstadt_unis corpora, *without* word-level translation) for the cross-lingual subtask.

Language	Model	Score
Models using word-level translation		
Basque	berteus-base-cased	$0.509_{(8)}$
Catalan	roberta-base-ca	$0.554_{(8)}$
Spanish	xlm-roberta-large	$0.570_{(7)}$
Combined English corpora without word-level translation		
Basque	xlm-roberta-base	$0.649_{(2)*}$
	xlm-roberta-large	$0.641_{(2)*}$
Catalan	xlm-roberta-base	$0.647_{(2)*}$
	xlm-roberta-large	$0.655_{(2)*}$
Spanish	xlm-roberta-base	$0.670_{(1)*}$
	xlm-roberta-large	$0.638_{(2)*}$

Table 10: Scores of our models, for the cross-lingual subtask, on each test set. Our ranking on the shared task for each test set is indicated as a subscript. * indicates the ranking that we would obtain in the shared task using the post-evaluation baseline models.

6 Conclusion

This paper describes our participation at the SemEval Shared Task 10 on structured sentiment analysis. We participated both in the monolingual and cross-lingual (zero-shot) setups. We applied a simple, but effective approach, relying on off-the-shelf tools, traditionally used for other purposes, and used them to predict sentiment graphs instead. More particularly, for the monolingual setup, we linked pre-trained language models with bi-affine graph parsing and training over single and multiple treebanks. In the zero-shot setup, we followed a similar approach, but relied on publicly available translation models to obtain training data, by applying a word-level translation of treebanks, to then train models similarly to the monolingual setup.

Acknowledgements

This work is supported by a 2020 Leonardo Grant for Researchers and Cultural Creators from the FBBVA,[8] as well as by the European Research Council (ERC), under the European Union's Horizon 2020 research and innovation programme (FASTPARSE, grant agreement No 714150). The work is also supported by ERDF/MICINN-AEI (SCANNER-UDC, PID2020-113230RB-C21), by Xunta de Galicia (ED431C 2020/11), and by Centro de Investigación de Galicia "CITIC" which is funded by Xunta de Galicia, Spain and the European Union (ERDF - Galicia 2014–2020 Program), by grant ED431G 2019/01.

[8]FBBVA accepts no responsibility for the opinions, statements and contents included in the project and/or the results thereof, which are entirely the responsibility of the authors.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Rodrigo Agerri, Iñaki San Vicente, Jon Ander Campos, Ander Barrena, Xabier Saralegi, Aitor Soroa, and Eneko Agirre. 2020. Give your text representation models some love: the case for Basque. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4781–4788, Marseille, France. European Language Resources Association.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Andrey Kutuzov, Oberländer, Laura Ana Maria, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? a strong baseline for aspect-based sentiment analysis with RoBERTa. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1816–1829, Online. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulić. 2021. Is supervised syntactic parsing beneficial for language understanding tasks? an empirical investigation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3090–3104, Online. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Lianzhe Huang, Xin Sun, Sujian Li, Linhao Zhang, and Houfeng Wang. 2020. Syntax-aware graph attention network for aspect-level sentiment classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 799–810, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mahesh Joshi and Carolyn Rosé. 2009. Generalizing dependency features for opinion mining. In *Proceedings of the ACL-IJCNLP 2009 conference short papers*, pages 313–316.

Hiroshi Kanayama and Ran Iwamoto. 2020. How universal are Universal Dependencies? exploiting syntax for multilingual clause-level sentiment detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4063–4073, Marseille, France. European Language Resources Association.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. 2021. Improving BERT with syntax-aware local attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 645–653, Online. Association for Computational Linguistics.

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

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Thien Hai Nguyen and Kiyoaki Shirai. 2015. PhraseRNN: Phrase recursive neural network for aspect-based sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2509–2514, Lisbon, Portugal. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926, Denver, Colorado. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In

Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pages 339–346, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Soujanya Poria, Erik Cambria, Grégoire Winterstein, and Guang-Bin Huang. 2014. Sentic patterns: Dependency-based rules for concept-level sentiment analysis. *Knowledge-Based Systems*, 69:45–63.

Devendra Sachan, Yuhao Zhang, Peng Qi, and William L. Hamilton. 2021. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661, Online. Association for Computational Linguistics.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

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

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5679–5688, Hong Kong, China. Association for Computational Linguistics.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, Osaka, Japan. The COLING 2016 Organizing Committee.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal. European Association for Machine Translation.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Peter Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

David Vilares, Miguel A. Alonso, and Carlos Gómez-Rodríguez. 2015a. A syntactic approach for opinion mining on Spanish reviews. *Natural Language Engineering*, 21(01):139–163.

David Vilares, Miguel A Alonso, and Carlos Gómez-Rodríguez. 2015b. On the usefulness of lexical and syntactic processing in polarity classification of t witter messages. *Journal of the Association for Information Science and Technology*, 66(9):1799–1816.

David Vilares, Carlos Gómez-Rodríguez, and Miguel A Alonso. 2017. Universal, unsupervised (rule-based), uncovered sentiment analysis. *Knowledge-Based Systems*, 118:45–55.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Daxin Jiang, and Nan Duan. 2021. Syntax-enhanced pre-trained model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5412–5422, Online. Association for Computational Linguistics.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523, Melbourne, Australia. Association for Computational Linguistics.

Meishan Zhang, Qiansheng Wang, and Guohong Fu. 2019. End-to-end neural opinion extraction with a transition-based model. *Information Systems*, 80:56–63.

Yuan Zhang and Yue Zhang. 2019. Tree communication models for sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3518–3527, Florence, Italy. Association for Computational Linguistics.

Z. Zhang, Y. Wu, J. Zhou, S. Duan, H. Zhao, and R. Wang. 2022. Sg-net: Syntax guided transformer for language representation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01):1–1.

A Shared Task Leaderboard

User	Team	NoReC_fine	MultiBooked_ca	MultiBooked_eu	OpeNER_en	OpeNER_es	MPQA	Darmstadt_unis	Average
zhixiaobao		0.529 (2)	0.728 (1)	0.739 (1)	0.760 (2)	0.722 (4)	0.447 (1)	0.494 (1)	0.631 (1)
Cong666		0.524 (3)	0.728 (1)	0.739 (1)	0.763 (1)	0.742 (1)	0.416 (2)	0.485 (2)	0.628 (2)
gmorio	Hitachi	0.533 (1)	0.709 (3)	0.715 (3)	0.756 (3)	0.732 (3)	0.402 (3)	0.463 (3)	0.616 (3)
colorful		0.497 (5)	0.678 (6)	0.723 (2)	0.745 (4)	0.735 (2)	0.375 (5)	0.380 (12)	0.590 (4)
whu_stone	sixsixsix	0.483 (9)	0.711 (2)	0.681 (6)	0.727 (6)	0.686 (7)	0.379 (4)	0.373 (13)	0.577 (5)
KE_AI		0.483 (9)	0.711 (2)	0.681 (6)	0.727 (6)	0.686 (7)	0.364 (7)	0.373 (13)	0.575 (6)
Fadi	SeqL	0.488 (7)	0.699 (4)	0.701 (4)	0.730 (5)	0.700 (5)	0.245 (20)	0.394 (11)	0.565 (7)
lys_acoruna	LyS_ACoruña	0.462 (10)	0.653 (8)	0.680 (7)	0.698 (9)	0.692 (6)	0.349 (10)	0.414 (7)	0.564 (8)
QiZhang	ECNU_ICA	0.496 (6)	0.684 (5)	0.686 (5)	0.676 (10)	0.623 (11)	0.351 (8)	0.409 (8)	0.561 (9)
luxinyu	ohhhmygosh	0.487 (8)	0.658 (7)	0.651 (9)	0.710 (7)	0.669 (8)	0.269 (19)	0.416 (6)	0.551 (10)
rafalposwiata	OPI	0.459 (11)	0.650 (9)	0.653 (8)	0.670 (11)	0.663 (9)	0.326 (13)	0.395 (10)	0.545 (11)
evanyfyang		0.213 (22)	0.635 (11)	0.639 (10)	0.703 (8)	0.642 (10)	0.350 (9)	0.449 (4)	0.519 (12)
robvanderg		0.366 (13)	0.648 (10)	0.605 (11)	0.632 (14)	0.614 (13)	0.296 (15)	0.344 (14)	0.501 (13)
psarangi	AMEX AI Labs	0.343 (15)	0.634 (12)	0.559 (12)	0.634 (13)	0.595 (14)	0.283 (17)	0.320 (17)	0.481 (14)
chx.dou	abondoned	0.395 (12)	0.583 (13)	0.506 (13)	0.626 (15)	0.622 (12)	0.309 (14)	0.280 (19)	0.474 (15)
zaizhep	MMAI	0.329 (16)	0.525 (14)	0.478 (17)	0.623 (16)	0.539 (16)	0.367 (6)	0.342 (15)	0.458 (16)
janpf		0.280 (19)	0.517 (15)	0.439 (19)	0.651 (12)	0.504 (17)	0.338 (11)	0.417 (5)	0.449 (17)
etms.kgp	ETMS@IITKGP	0.351 (14)	0.508 (16)	0.438 (20)	0.626 (15)	0.544 (15)	0.327 (12)	0.330 (16)	0.446 (18)
jylong		0.323 (18)	0.474 (19)	0.504 (14)	0.476 (17)	0.375 (21)	0.274 (18)	0.223 (21)	0.379 (19)
ouzh		0.323 (18)	0.474 (19)	0.504 (14)	0.476 (17)	0.375 (21)	0.274 (18)	0.223 (21)	0.378 (20)
SPDB_Innovation_Lab	Innovation Lab	0.325 (17)	0.469 (20)	0.486 (16)	0.471 (18)	0.362 (22)	0.289 (16)	0.202 (22)	0.372 (21)
lucasrafaelc		0.251 (21)	0.505 (17)	0.467 (18)	0.431 (19)	0.399 (19)	0.232 (21)	0.230 (20)	0.359 (22)
foodchup		0.265 (20)	0.493 (18)	0.491 (15)	0.415 (20)	0.480 (18)	0.149 (22)	0.139 (24)	0.347 (23)
jzh1qaz		0.186 (25)	0.431 (21)	0.385 (21)	0.381 (21)	0.393 (20)	0.094 (23)	0.092 (26)	0.280 (24)
hades_d	Mirs	0.504 (4)	0.678 (6)	0.000 (25)	0.000 (25)	0.000 (26)	0.375 (5)	0.400 (9)	0.280 (24)
huyenbui117		0.194 (23)	0.341 (22)	0.374 (22)	0.316 (23)	0.245 (25)	0.009 (26)	0.053 (27)	0.219 (25)
karun842002	SSN_MLRG1	0.191 (24)	0.323 (23)	0.331 (23)	0.306 (24)	0.257 (24)	0.015 (25)	0.104 (25)	0.218 (26)
gerarld	nlp2077	0.000 (26)	0.269 (24)	0.303 (24)	0.354 (22)	0.321 (23)	0.019 (24)	0.180 (23)	0.207 (27)
michael_wzhu91	kobe4ever	0.000 (26)	0.000 (25)	0.000 (25)	0.000 (25)	0.000 (26)	0.000 (27)	0.306 (18)	0.044 (28)
normalkim		0.000 (26)	0.000 (25)	0.000 (25)	0.000 (25)	0.000 (26)	0.000 (27)	0.000 (28)	0.000 (29)
UniParma	UniParma	0.000 (26)	0.000 (25)	0.000 (25)	0.000 (25)	0.000 (26)	0.000 (27)	0.000 (28)	0.000 (29)
whu_venti		0.000 (26)	0.000 (25)	0.000 (25)	0.000 (25)	0.000 (26)	0.000 (27)	0.000 (28)	0.000 (29)

Table 11: Leaderboard of all participants in the monolingual task

User	Team	OpeNER_es	MultiBooked_ca	MultiBooked_eu	Average
Cong666		0.644 (1)	0.643 (1)	0.632 (1)	0.640 (1)
luxinyu	ohhhmygosh	0.620 (3)	0.605 (4)	0.569 (2	0.598 (2)
gmorio	Hitachi	0.628 (2)	0.607 (3)	0.527 (4)	0.587 (3)
whu_stone	sixsixsix	0.604 (5)	0.596 (5)	0.512 (7)	0.571 (4)
QiZhang	ECNU_ICA	0.551 (10)	0.615 (2)	0.530 (3)	0.566 (5)
Fadi	SeqL	0.589 (6)	0.593 (6)	0.516 (6)	0.566 (5)
colorful		0.620 (3)	0.543 (11)	0.527 (4)	0.563 (6)
hades_d	Mirs	0.617 (4)	0.544 (10)	0.522 (5)	0.561 (7)
lys_acoruna	LyS_ACoruña	0.570 (7)	0.554 (8)	0.509 (8)	0.544 (8)
rafalposwiata	OPI	0.564 (8)	0.586 (7)	0.444 (12)	0.531 (9)
KE_AI		0.561 (9)	0.552 (9)	0.463 (11)	0.525 (10)
etms.kgp	ETMS@IITKGP	0.542 (11)	0.506 (12)	0.431 (13)	0.493 (11)
jylong		0.375 (12)	0.474 (13)	0.504 (9)	0.451 (12)
ouzh		0.375 (12)	0.474 (13)	0.504 (9)	0.451 (12)
SPDB_Innovation_Lab	SPDB Innovation Lab	0.362 (13)	0.469 (14)	0.486 (10)	0.439 (13)
gerarld	nlp2077	0.321 (14)	0.269 (15)	0.303 (14)	0.298 (14)
janpf		0.315 (15)	0.259 (16)	0.243 (15)	0.272 (15)
chx.dou	abondoned	0.013 (16)	0.009 (17)	0.004 (16)	0.009 (16)
jzh1qaz		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
zhixiaobao		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
psarangi	AMEX AI Labs	0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
normalkim		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
zaizhep	MMAI	0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
lucasrafaelc		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
evanyfyang		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
robvanderg		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
michael_wzhu91	kobe4ever	0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
UniParma	UniParma	0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
huyenbui117		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
karun842002	SSN_MLRG1	0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
whu_venti		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)
foodchup		0.000 (17)	0.000 (18)	0.000 (17)	0.000 (17)

Table 12: Leaderboard of all participants in the cross-lingual task

SPDB Innovation Lab at SemEval-2022 Task 10: A Novel End-to-End Structured Sentiment Analysis Model based on the ERNIE-M

Yalong Jia, Zhenghui Ou, Yang Yang
Shanghai Pudong Development Bank
{jiayl2, ouzh, yangy103}@spdb.com.cn

Abstract

Sentiment analysis is a classical problem of natural language processing. SemEval 2022 sets a problem on the structured sentiment analysis in task 10, which is also a study-worthy topic in research area. In this paper, we propose a method which can predict structured sentiment information on multiple languages with limited data. The ERNIE-M pretrained language model is employed as a lingual feature extractor which works well on multiple language processing, followed by a graph parser as a opinion extractor. The method can predict structured sentiment information with high interpretability. We apply data augmentation as the given datasets are so small. Furthermore, we use K-fold cross-validation and DeBERTaV3 pretrained model as extra English embedding generator to train multiple models as our ensemble strategies. Experimental results show that the proposed model has considerable performance on both monolingual and cross-lingual tasks.

1 Introduction

Sentiment analysis (Liu, 2012) is widely used in many aspects of computer science nowadays, such as human computer interaction, lingual feature extraction, etc. Affective computing techniques enable us to explore the sentiment message, which depicts the preference, emotion or even idea of people, behind the sentence itself.

Sentiment analysis task can be classified into text level, sentence level, entity level and opinion tuple level. The goal of text or sentence level sentiment analysis is to predict sentiments of given documents or sentences. On the other hand, entity level sentiment analysis needs to consider sentiments between each entity in given sentence. Furthermore, a sentence may contain multiple entities with different sentiments (positive or negative). Therefore, in comparison to entity level sentiment analysis, opinion tuple level sentiment analysis requires additional extraction of relations between entities and opinions.

Structured sentiment analysis (Barnes et al., 2022) is a task to predict a sentiment graph for given sentences. It can be theoretically cast as an information extraction problem in which one attempts to find all of the opinion tuples $O = O_1, ..., O_n$ in a text. As we can see in Figure 1, each opinion O_i is a tuple (h, t, e, p) where h is a holder who expresses a polarity p towards a target t through a sentiment expression e, implicitly defining pairwise relationships between elements of the same tuple.

The structure of the paper is as follows. Section 2 briefly reviews recent works on similar tasks; Section 3 describes our model structure in detail. Section 4 shows the analysis of the given datasets; Section 5 introduces our experimental setting and results. And finally in Section 6, we make a conclusion and give the ideas about future works.

2 Background

Dividing structured sentiment analysis into multiple subtasks is a traditional approach, by first identifying holders, targets and expressions through Named Entity Recognition (NER) module, then predicting relations among the entities. (Peng et al., 2020; Li et al., 2019) are baselines with good performance. On the other hand, the end-to-end sentiment analysis is a straight-forward approach, which directly extracts target and expression without splitting them into sub-tasks. (He et al., 2019) presents an interactive multi-task learning network(IMN) implemented by a series of a multi-layer CNN modules. Recently, (Barnes et al., 2021) cast the structured sentiment problem as the dependency graph parsing problem, and proposed a method that outperforms the SOTA(state-of-the-art) baselines on

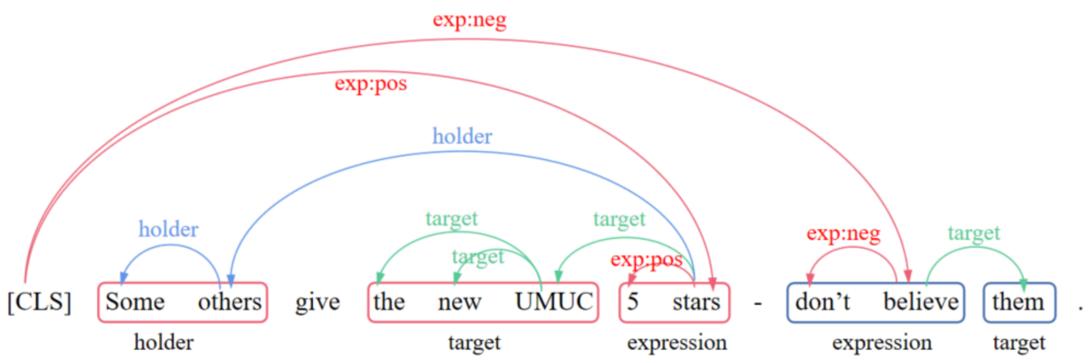


Figure 1: A structured sentiment graph is composed of a holder, target, sentiment expression, their relationships and a polarity attribute. Holders and targets can be none.

extensive experimental results.

In this task, 7 small datasets involving 5 different languages are given, which means our method needs to support multiple languages with limited data. Therefore, in this work, we use ERNIE-M pretrained model as word embedding generator, as it shows SOTA performances in various of NLP tasks in multiple languages. Because of the outstanding performance of (Barnes et al., 2021), we employ a similar network structure to extract the sentiment information.

3 Model structure

The model structure is similar to the head-final model structure in (Barnes et al., 2021), while we use a pretrained model in this work. A bert-style (ERNIE-M (Ouyang et al., 2020)) pretrained model takes sentences starting with "[CLS]" token as input. We connect "[CLS]" token with the last word of the expression, which is the root node of the tuple (h, t, e). The connection type is related to the sentiment polarity, "exp:pos" for positive polarity, "exp:neu" for neutral polarity, and "exp:neg" for negative corresponding to polarity, such that we are able to predict sentence polarity based on the connection type. As shown in Figure 1, a connection from "[CLS]" to "stars" and a connection from "[CLS]" to "believe" are established, with a connection type of "exp:pos" and "exp:neg" respectively. We describe the model structure in detail below.

For a given sentence $\overrightarrow{x} = (x_1, x_2, ..., x_n)$, where $x_i (1 \leq i \leq n)$ represents a single word. In this work, we use ERNIE-M pretrained model as a text feature extractor, which takes subword tokens as the model input. We apply subword-based tokenization on the input words.

As shown in Figure 2, we apply subword-based tokenization on the input sentence, getting $\overrightarrow{t} = (t_1, t_2, ..., t_m)$ for any $t_j (1 \leq j \leq m)$ representing a subword token. For instance, word "restful" will be split into "rest" and "##ful", where "##" indicates that the token is not the start of a word. And the process can be easily inversed by these special characters, such as we can restore ["Great", "and", "rest", "##ful", "place", "to", "stay", "."] to its original status ["Great", "and", "restful", "place", "to", "stay" "."]. After that, we input the subword tokens into the ERNIE-M model and get the embedding of the subword tokens. Then, we apply average pooling on the subword embeddings($\overrightarrow{v}$ in equation 1) which belong to a same original word to get the word representation $\overrightarrow{c} = (c_1, c_2, ..., c_n)$.

After obtaining the word representations of the sentence, we perform a position-wise feed-forward networks to obtain the representation of the heads and dependents, where heads represent head nodes, and the dependents represent follower nodes. Then we use Bilinear Attention Network (Kim et al., 2018) to calculate the pairwise correlation between each two words in the sentence.

$$\overrightarrow{v} = (v_1, ..., v_m) = \text{ERNIE-M}(t_1, ..., t_m) \quad (1)$$

$$h_i^{head} = FFN^{head}(c_i) \quad (2)$$

$$h_j^{dep} = FFN^{dep}(c_j) \quad (3)$$

$$score_{i,j} = Bilinear(h_i^{head}, h_j^{dep}) \quad (4)$$

In equation 4, we obtain a score matrix that indicates the relationship between each word in the input sentence pair-wisely, by passing head and

		ALL		Valid						Polarity		
		#	avg	#	avg	Opinion	Holder	Target	Exp	pos	neu	neg
Darmstadt_unis	train	2,253	19.99	681	21.25	806	63	806	806	340	102	364
	dev	232	18.09	82	20.21	98	9	98	98	29	15	54
MPQA	train	5,873	23.39	1,254	29.83	1,706	1,425	1,481	1,706	671	337	698
	dev	2,036	23.22	416	31.33	570	406	494	570	231	124	215
MultiBooked_ca	train	1,174	15.62	1,002	16.14	1,989	169	1,705	1,989	1,273	0	716
	dev	167	13.37	140	14.21	258	15	211	258	151	0	107
MultiBooked_eu	train	1,063	10.52	899	10.77	1,679	205	1,277	1,679	1,401	0	278
	dev	152	10.70	120	10.51	203	33	152	203	167	0	36
NoReC_fine	train	8,634	16.71	4,555	19.55	8,448	898	6,778	8,448	5,695	0	2,753
	dev	1,531	16.92	821	19.12	1,432	120	1,152	1,432	988	0	444
OpeNER_en	train	1,744	14.72	1,400	14.99	2,884	266	2679	2,884	2,101	0	783
	dev	249	14.22	198	14.98	400	49	371	400	284	0	116
OpeNER_es	train	1,438	17.13	1,252	17.58	3,042	176	2748	3,042	2,472	0	570
	dev	206	17.08	174	17.71	387	23	363	387	317	0	70

Table 1: Statistic of the given datasets, including the number of samples and average word count of all samples and valid samples(where opinions are not empty), as well as number of opinion, holder, target, expression and polarity in each dataset.

dependent embedding into a bilinear attention network, followed by a softmax layer. We apply cross-entropy loss as the loss function during model training.

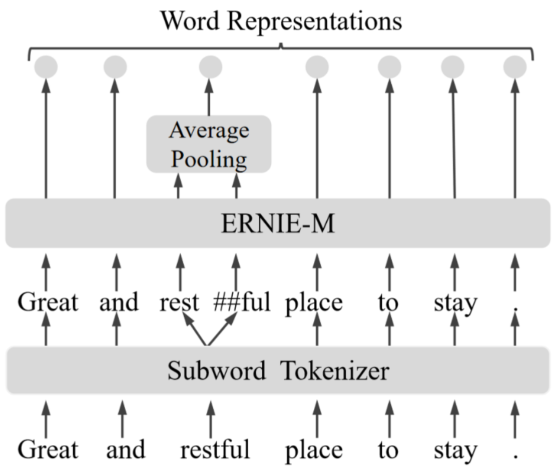

Word Representations
Average Pooling
ERNIE-M
Great and rest ##ful place to stay .
Subword Tokenizer
Great and restful place to stay .

Figure 2: The process of generating the word representations.

4 Data

SemEval-2022 task 10 provides a total of 7 datasets: NoRec_fine(Øvrelid et al., 2020), MultiBooked_eu(Barnes et al., 2018), MultiBooked_ca(Barnes et al., 2018), OpeNER_es(Agerri et al., 2013), OpeNER_en(Agerri et al., 2013), MPQA(Wiebe et al., 2005), Darmstadt_unis(Toprak et al., 2010), involving different languages (en, ca, eu, no, es) and a couple of domains. MultiBooked_ca (Catalan), MultiBooked_eu (Basque), Opener_en(English), and Opener_es(Spanish) belong to hotel reviews domain. NoRec is a Norwegian dataset in literature, movies, video games, restaurants, music and theater domains. Darmstadt_unis is an English dataset in university domain, and MPQA is an English dataset in news domain. Our data analysis on the given datasets is shown in Table 1.

We can see that there are three different types of sentiment(positive, neutral, negative) in the opinions of Darmstadt_unis and MPQA, while others only contain Positive and Negative sentiments. In addition, we find that the number of expressions is always larger than or equal to the number of holders and targets for all datasets, which means there is at least one expression in each opinion. We can draw a conclusion that in a quadruple (h, t, e, p), both h and t may be missing, but e and p are not. Therefore, the expression(more precisely, the last word of expression) is defined as the root node of a (h, t, e) triplet.

5 Experiment and result

5.1 Data preprocessing

We process the labels of the original data into the format that the model required. Taking the sentence shown in Figure 1 as an example, the preprocessed result is shown in Table 2. We truncated the sentence due to the length of the sentence.

5.2 Experiments

In model validation period, we introduce two strategies, i.e., data augmentation and focal loss, which

	[CLS]	Some	others	give	the	new	UMUC	5	stars	.
[CLS]	-	-	-	-	-	-	-	-	exp:pos	-
Some	-	-	-	-	-	-	-	-	-	-
others	-	holder	-	-	-	-	-	-	-	-
give	-	-	-	-	-	-	-	-	-	-
the	-	-	-	-	-	-	-	-	-	-
new	-	-	-	-	-	-	-	-	-	-
UMUC	-	-	-	-	target	target	-	-	-	-
5	-	-	-	-	-	-	-	-	-	-
stars	-	-	holder	-	-	-	target	exp:pos	-	-
.	-	-	-	-	-	-	-	-	-	-

Table 2: The processed label for the model, where "-" indicates that there is no relationship between two words. The words on the y-axis are heads, and the words on the x-axis are dependents.

	origin	**+aug**	**+aug+focal**
NoRec	0.4798	0.4939	-
Multib_ca	0.7182	0.7281	-
Multib_eu	0.6781	0.7070	-
OpeNER_en	0.7271	0.7321	-
OpeNER_es	0.6758	0.7202	-
MPQA	0.3375	0.3564	0.3582
Dm_unis	0.3981	0.4412	0.4073

Table 3: The results on the dev dataset, origin means that there is no extra strategy applied, +aug means a mixed training set merged by other training sets and itself, and +aug+focal means that the strategy of focal loss is applied based on +aug.

dataset	**score**
NoRec_fine	0.497
Multib_ca	0.678
Multib_eu	0.723
OpeNER_en	0.745
OpeNER_es	0.735
MPQA	0.375
Dm_unis	0.380

dataset	**score**
EN-ES	0.620
EN-CA	0.543
EN-EU	0.527

Table 4: The left table is the result of monolingual, and the right table is the result of cross-lingual.

are described in detail below.

Data augmentation As shown in Table 1, the dataset is small. The largest dataset given is the NoRec dataset with 8,634 pieces of data, and only 4,555 pieces of data are left after removing the samples with empty opinions. Therefore, we use all datasets with the same polarity types for training with the help of ERNIE-M model, which supports multiple languages. For instance, we find that the Darmstadt_unis and MPQA have three categories of sentiment polarity, while the others have two categories of sentiment polarity. Hence, we merge Darmstadt_unis and MPQA into one dataset, and merge the others into another dataset. The results are shown in Table 3, and the evaluation method[1] is based on (Barnes et al., 2021). There is an average improvement of 2.36% after training the model with merged dataset, and a most significant improvement on the Opener_es at 4.44%.

[1]https://github.com/jerbarnes/semeval22_structured_sentiment

Focal loss As shown in Table 2, we can see the imbalance of the labels. The relationship between most words are none. Therefore, we introduce focal loss strategy, which reduces the loss weight of the "none" and increases the loss weight of the other labels. Because of the baseline scores on the MPQA and Darmstadt_unis are relatively lower, we test focal loss strategy on these two datasets. However, as we can see in Table 3, the performance is not good on Darmstadt_unis. Therefore, we do not use the focal loss strategy in the Darmstadt_unis solution.

5.3 Ensemble

We apply K-fold cross-validation and ensemble on the results of different pretrained models.

K-fold cross-validation For each dataset, we merge the original training set and the validation set, and perform K-fold segmentation after shuffling. The selection of K is related to the proportion of the original dataset training set and validation set. Notice that the proportion of the training set and the validation set after the segmentation should be approximated to their original proportion. We repeat this operation K times and obtain K different models. Finally, we ensemble K different models to get the final model.

Usage of other pretrained models For English datasets (Darmstadt_unis, MPQA, Opener_en), we train additional model by replacing the ERNIE-M model with the DeBERTaV3(He et al., 2021) model as the lingual feature extractor, and keep the other parts unchanged. Then we perform K-fold cross-validation strategy on both models and ensemble them in the same way. The results are shown in Table 4.

6 Conclusion

In order to solve the issues of small dataset and cross language in SemEval-2022 task 10, we introduce a multilingual pretrained model ERNIE-M as a lingual feature extractor to the given baseline model. Furthermore, we use multiple strategies such as data augmentation, K-fold cross-validation, focal loss and ensemble to improve the model performance. In the future, we can take other techniques to do further optimization, such as different data augmentation technieques, fine-tuning the pretrained model with the in-domain dataset, and changing the label from word level to subword level to fit the subword representation of the pretrained model.

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. *arXiv preprint arXiv:2105.14504*.

Jeremy Barnes, Oberländer Laura Ana Maria Kutuzov, Andrey and, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. *arXiv preprint arXiv:1906.06906*.

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. 2018. Bilinear attention networks. *Advances in Neural Information Processing Systems*, 31.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6714–6721.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-m: enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. *arXiv preprint arXiv:2012.15674*.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

HITSZ-HLT at SemEval-2022 Task 10: A Span-Relation Extraction Framework for Structured Sentiment Analysis

Yihui Li, Yifan Yang, Yice Zhang, Ruifeng Xu
Harbin Institute of Technology (Shenzhen), China
20s051013@stu.hit.edu.cn, {evanyfyang,zhangyc_hit}@163.com,
xuruifeng@hit.edu.cn

Abstract

This paper describes our system that participated in the SemEval-2022 Task 10: Structured Sentiment Analysis, which aims to extract opinion tuples from texts. A full opinion tuple generally contains an opinion holder, an opinion target, the sentiment expression, and the corresponding polarity. The complex structure of the opinion tuple makes the task challenging. To address this task, we formalize it as a span-relation extraction problem and propose a two-stage extraction framework accordingly. In the first stage, we employ the span module to enumerate spans and then recognize the type of every span. In the second stage, we employ the relation module to determine the relation between spans. Our system achieves competitive results and ranks among the top-10 systems in almost subtasks.

1 Introduction

Sentiment analysis, also called opinion mining, aims to analysis people's attitudes and emotions towards specific targets, such as products, organizations, events, etc (Liu, 2012). It has become an important research field in natural language processing (Medhat et al., 2014; Hussein, 2018; Zhang et al., 2018).

Structured sentiment analysis. Barnes et al. (2021) formally defines a complete opinion as a quadruple (h, t, e, p) where h is a holder who expresses a polarity p towards a target t through a sentiment expression e. Figure 1 presents examples of opinion quadruples. On the basis of this definition, Barnes et al. (2022) formally establishes a benchmark for structured sentiment analysis. This benchmark consists of two tracks, the monolingual track and the crosslingual track, and we participate the monolingual track.

In this paper, we cast this task as a span-relation extraction problem (Jiang et al., 2020), which is a formalization that has been widely used in many

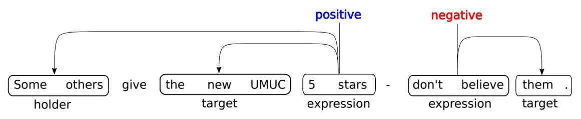


Figure 1: Examples of opinion quadruples (Barnes et al., 2021).

information extraction tasks (Eberts and Ulges, 2019; Xu et al., 2021; Lu and Ng, 2021; Li et al., 2021). With the span-relation formalization, opinion quadruple extraction is divided into two stages.

- In the first stage, we extract "meaningful" text spans and recognize their types. Specifically for this task, the type space is $\{h, t, e\}$. For those spans classified as e, we additionally detect the sentiment polarity they express.
- In the second stage, we determine the relations between spans. The relation space is set to $\{eh, et, ee, \text{none}\}$. eh and eh are used to facilitate the matching of sentiment expressions, holders, and targets during the decoding process. ee is used to deal with discontinuous sentiment expressions, which is inspired by (Li et al., 2021).

In addition, we employ span pruning (Xu et al., 2021) to reduce the computation of the second stage. Finally, opinion quadruples are decoding from the results of two stages. Our system achieves competitive results and ranks among the top-10 systems in almost subtasks.

2 Related Work

Span extraction is a fundamental method for many tasks, such as named entity recognition, aspect-level sentiment analysis, etc. This method performs element extraction by enumerating all possible spans and then determining the type of spans. Xu et al. (2017) attempts to determine the type of spans by encoding all possible spans into a representation of the same size. Sohrab and Miwa (2018) also enumerate all potential spans and then use a deep network to classify them. Luan et al. (2019) leverage the coreference and relation type confidences to enhance the representation of spans. Tan et al. (2020) added the task of span boundary detection to improve the sensitivity of the model to span boundaries. This approach was able to produce higher quality candidate spans.

Span-relation extraction for sentiment tasks focuses on extracting categories of spans and relationships between spans, such as extracting relationships between entities and extracting aspect sentiment triplet. Peng et al. (2020) try to solve the aspect sentiment triplet extraction problem using a two-stage pipeline. The first stage extracts the target as well as its polarity and opinion, using the BIOES annotation method.The second stage then couples the extracted target and opinion terms to determine their paired sentiment relation. However, This method may suffer from the problem of error propagation. End-to-end methods(Wu et al., 2020; Xu et al., 2020) can extract both span and their relationships. However, previous work has usually used word-to-word interactions to predict sentiment relationships. The disadvantage of this approach is that it ignores the sentiment consistency of the entire span. The method proposed by Xu et al. (2021) can accurately enumerate all the span representations with high likelihood and then predict the sentiment relationship between them. This approach can mitigate the impact of errors in the span extraction step on subsequent relationship prediction, while it also preserves the sentiment consistency of the entire span when predicting relationships

3 Our System

Given the input text, we first obtain its contextualized representation through a pre-trained language model, BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019). Then we input the contextualized representation into the span module and the relation module in turn to extract spans and detect relations.

3.1 Span Module

Span module roughly follows the idea of Tan et al. (2020). First we employ two binary classifiers to detect the start and end position of the “meaningful” spans respectively. Then another classifier is adopted to match the start and end positions and determine the category.

3.1.1 Start and End Prediction

Suppose $H \in \mathbb{R}^{n\times d}$ is the contextualized representation output by the language model, where n is the length of the input text. Then We calculate the probability of each position being the start or end position:

$$P_{start} = \text{sigmoid}(H \cdot W_{start}) \in \mathbb{R}^{n\times 1}, \quad (1)$$
$$P_{end} = \text{sigmoid}(H \cdot W_{end}) \in \mathbb{R}^{n\times 1}. \quad (2)$$

where $W_{start}, W_{end} \in \mathbb{R}^{d\times 1}$ are learnable parameters. Afterwards, we can decode the candidate start and end positions:

$$I_{start(>t)} = \{i \mid P^{(i)}_{start} > t, i = 1, \cdots, n\}, \quad (3)$$
$$I_{end(>t)} = \{i \mid P^{(i)}_{end} > t, i = 1, \cdots, n\}, \quad (4)$$

where threshold $t \in (0, 0.5]$ is hyper-parameter.

3.1.2 Start-End Matching and Classification

We adopt a classifier to match the start and end positions and determine the category. If the start position $i \in I_{start(>t)}$ and the end position $j \in I_{end(>t)}$ satisfy $i \leq j$, then we predict the category of span (i, j):

$$r_{ij} = [h_i; h_j; f_{width}(i, j)] \in \mathbb{R}^{3d}, \quad (5)$$
$$P^{(i,j)}_{span} = \text{softmax}(\text{FFNN}_s(r_{ij})) \in \mathbb{R}^4, \quad (6)$$

where $f_{width}(i, j) \in \mathbb{R}^d$ denotes a learnable embedding based on width $j - i$, and FFNN denotes a feed-forward neural network with non-linear activation. The span category space is $\{h, t, e, \text{invalid}\}$, where h denotes the opinion holder, t denotes the opinion target, and e denotes the sentiment expression.

For those spans classified as e, we predict its sentiment polarity additionally:

$$P^{(i,j)}_{polarity} = \text{softmax}(\text{FFNN}_p(r_{ij})) \in \mathbb{R}^3, \quad (7)$$

where the polarity space is $\{\text{POS}, \text{NEG}, \text{NEU}\}$.

3.2 Relation Module

Relation module aims to determine the relations between spans. For a span pair, we first construct a relation representation based on the span representations and then feed it into a relation classifier. Notice that we employ span pruning (Xu et al., 2021) to reduce the computation.

Language	Pretrained Model
English	roberta-large (Liu et al., 2019)
Spanish	BSC-TeMU/roberta-base-bne (Gutiérrez-Fandiño et al., 2021)
Norwegian	pere/norwegian-roberta-base
Basque	ixa-ehu/berteus-base-cased (Agerri et al., 2020)
Catalan	BSC-TeMU/roberta-base-ca (Armengol-Estapé et al., 2021)

Table 1: Pretrained language model for 5 different languages.

3.2.1 Span Pruning

Considering the large number of the predicted spans, it is not computationally practical to consider all possible pairwise relations. Following Xu et al. (2021), we prune spans in the relation classification stage. The holder, target, and sentiment expression candidates are selected based on the scores of the mention types for each span:

$$\Phi^{(i,j)}_{holder} = P^{(i,j)}_{span}(m = h), \tag{8}$$

$$\Phi^{(i,j)}_{target} = P^{(i,j)}_{span}(m = t), \tag{9}$$

$$\Phi^{(i,j)}_{expression} = P^{(i,j)}_{span}(m = e). \tag{10}$$

We use the mention scores Φ_{source}, Φ_{target} and $\Phi_{expression}$to select the top k candidates and obtain the holder candidate pool S^h, the target candidate pool S^t, and the sentiment expression candidate pool S^e, respectively. The value of k is related to the length of the sentence n:

$$k = \max(n \cdot z, k_{min}), \tag{11}$$

where z, k_{min} are hyper-parameters.

3.3 Datasets

	Language	Domain	Train	Dev	Test
MPQA	English	news	5873	2063	2112
DS$_{\text{Unis}}$	English	e-commerce	2252	232	318
OpenNER$_{\text{EN}}$	English	hotel	1745	250	499
OpenNER$_{\text{ES}}$	Spanish	hotel	1439	206	410
NoReC$_{\text{Fine}}$	Norwegian	multi-domain	8634	1531	1272
MultiB$_{\text{EU}}$	Basque	hotel	1064	152	305
MultiB$_{\text{CA}}$	Catalan	hotel	1174	168	335

Table 2: Data statistics.

3.3.1 Relation Classification

For most datasets, we only detect two relations, `expression-holder` and `expression-holder`. For datasets with discontinuous sentiment expressions, we detect `expression-expression` relation additionally. We obtain the candidate pair representation by coupling each expression candidate representation $s^e_{a,b}$ with the other candidate representation. For an expression candidate $(a, b) \in S^e$ and a holder candidate $(c, d) \in S^h$, their pair representation is:

$$g^{e,h}_{(a,b),(c,d)} = [r_{a,b}; r_{c,d}; f_{distance}(a, b, c, d) \\ f_{context}(a, b, c, d); f_{type}(e); f_{type}(h)]$$

where $f_{distance} \in R^d$ denotes a learnable embedding based distance $min(|b-c|, |a-d|)$, $f_{context} \in R^d$ is obtained by performing max-pooling operation on the context between the two spans, and f_{type} is a learnable embedding for indicating the span type. We construct $g^{e,t}$, $g^{e,e}$ in a similar way.

Then we input the pair representation to a feed-forward neural network to determine the sentiment relation:

$$P^{((a,b),(c,d))}_{relation} = \text{softmax}(\text{FFNN}_r(g^{e,h}_{(a,b),(c,d)})),$$

where the relation space is $\{eh, et, ee, \text{none}\}$.

3.4 Training

During training, we utilize the cross-entropy function to calculate the loss of start & end prediction, span classification(SC), polarity classification(PC), and relation classification(RC). The overall optimization objective is to minimize the summation of these losses:

$$\mathcal{L} = \mathcal{L}_S + \mathcal{L}_E + \mathcal{L}_{SC} + \mathcal{L}_{PC} + \mathcal{L}_{RC}. \tag{12}$$

3.5 Sentiment Structure Decoding

We first decode the sentiment expressions and their sentiment polarities from the results of the span module. Then we obtain the holder candidate pool and the target candidate pool by span pruning. For each sentiment expression, we determine whether it has a relation with each holder candidate and target candidate. Finally, the opinion quadruplets are produced based on the result of the relation classification. In addition, for discontinuous sentiment expressions, sentiment expressions are merged according to the relation between sentiment expressions.

Model	**MPQA**	**DS$_{\text{Unis}}$**	**OpenNER$_{\text{EN}}$**	**OpenNER$_{\text{ES}}$**	**NoReC$_{\text{Fine}}$**	**MultiB$_{\text{EU}}$**	**MultiB$_{\text{CA}}$**
head first	17.40	25.00	-	-	29.50	56.80	54.70
head final	18.80	26.50	-	-	**31.20**	53.70	54.70
Span-Relation	**35.00**(9)	**44.90**(4)	**70.30**(8)	**64.20**(10)	21.30(21)	**63.90**(10)	**63.50**(12)

Table 3: Results on the test dataset (Sentiment Graph F_1, %).

4 Experiments

The monolingual track (Barnes et al., 2022) provides 7 structured sentiment datasets in five languages. Their statistics are listed on Table 2.

It is worth noting that there are discontinuous spans in the NoReC$_{\text{Fine}}$ and DS$_{\text{Unis}}$ datasets. For example, in "*It looks again like UMUC will do anything for money*", "*looks again*" and "*do anything*" are annotated as the same sentiment expression.

4.1 Experiment Settings

We use BERT or RoBERTa as the text encoders. Since this task has datasets in different languages, different pre-training models are used for different language, which is detailed in Table 1.

We used Adam as our optimizer. The maximum number of epochs is set to 15, z is set to 0.3, and k_{min} is set to 5. We train our model on the training set and keep the model that performs best on the validation set. We evaluate our model on Sentiment Graph F_1 (Barnes et al., 2021) and compare our model with sentiment graph approaches (Head-first/Head-final) (Barnes et al., 2021).

4.2 Main Results

The comparison results of opinion quadruple extraction are listed in Table 3. According to these results, our approach achieves better performance on most datasets than baselines, especially on MPQA exceeding baseline by 16.2%. This demonstrates the effectiveness of our approach for opinion quadruple extraction.

4.3 Ablation Study

Model	**MPQA**	**DS$_{\text{Unis}}$**	**OpenNER$_{\text{EN}}$**
Full Model	40.67	40.04	72.38
w/o f_{width}	37.50	37.40	71.14
w/o $f_{distance}$	38.83	36.42	71.46
w/o $f_{context}$	37.54	39.47	69.39

Table 4: Ablation results on the dev dataset.

We conduct an ablation study to examine the impact of some components in the proposed model and list the results in Table 4. It can be observed that the removal of width embedding, position embedding, and context all degrade the performance, indicating their necessity.

Model	**OpenNER$_{\text{ES}}$**	**NoReC$_{\text{Fine}}$**	**MultiB$_{\text{EU}}$**	**MultiB$_{\text{CA}}$**
	62.40	23.26	61.53	54.26
w mBERT	61.62	36.22	57.17	63.35

Table 5: Effect of mBERT representations.

In addition, we also compare the performance of the multilingual pre-trained model mBERT(bert-base-multilingual-cased)(Devlin et al., 2019) for this task. To this end, we compare the experimental performance of monolingual pre-trained models with mBERT on minor language datasets and list the results in Table 5. It can be observed that mBERT achieves similar performance to the monolingual pre-trained model for most minor languages. In addition, for the Norwegian and Catalan datasets, the performance of the models with mBERT improves considerably, which may be due to the lack of corpus in these two languages when training the monolingual pre-trained models.

5 Conclusions

This paper describes our system for structured sentiment analysis. We formalize the task as a span-relation extraction problem and propose a two-stage extraction approach, which consists of a span module and a relation module. Experimental results demonstrate the effectiveness of our approach.

References

Rodrigo Agerri, Iñaki San Vicente, Jon Ander Campos, Ander Barrena, Xabier Saralegi, Aitor Soroa, and Eneko Agirre. 2020. Give your text representation models some love: the case for basque. In *Proceedings of the 12th International Conference on Language Resources and Evaluation.*

Jordi Armengol-Estapé, Casimiro Pio Carrino, Carlos Rodriguez-Penagos, Ona de Gibert Bonet, Carme Armentano-Oller, Aitor Gonzalez-Agirre, Maite

Melero, and Marta Villegas. 2021. Are multilingual models the best choice for moderately under-resourced languages? A comprehensive assessment for Catalan. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4933–4946, Online. Association for Computational Linguistics.

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Oberländer-Laura Ana Maria Kutuzov, Andrey and, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Markus Eberts and Adrian Ulges. 2019. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*.

Asier Gutiérrez-Fandiño, Jordi Armengol-Estapé, Marc Pàmies, Joan Llop-Palao, Joaquín Silveira-Ocampo, Casimiro Pio Carrino, Aitor Gonzalez-Agirre, Carme Armentano-Oller, Carlos Rodriguez-Penagos, and Marta Villegas. 2021. Spanish language models.

Doaa Mohey El-Din Mohamed Hussein. 2018. A survey on sentiment analysis challenges. *Journal of King Saud University-Engineering Sciences*, 30(4):330–338.

Zhengbao Jiang, Wei Xu, Jun Araki, and Graham Neubig. 2020. Generalizing natural language analysis through span-relation representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2120–2133, Online. Association for Computational Linguistics.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021. A span-based model for joint overlapped and discontinuous named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4814–4828, Online. Association for Computational Linguistics.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

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

Jing Lu and Vincent Ng. 2021. Span-based event coreference resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13489–13497.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *NAACL-HLT (1)*.

Walaa Medhat, Ahmed Hassan, and Hoda Korashy. 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849.

Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary enhanced neural span classification for nested named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9016–9023.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2576–2585.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4755–4766.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349.

Mingbin Xu, Hui Jiang, and Sedtawut Watcharawittayakul. 2017. A local detection approach for named entity recognition and mention detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1237–1247.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253.

SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER)

Shervin Malmasi, Anjie Fang*, Besnik Fetahu*, Sudipta Kar*, Oleg Rokhlenko
Amazon.com, Inc.
Seattle, WA, USA
{malmasi,njfn,besnikf,sudipkar,olegro}@amazon.com

Abstract

We present the findings of SemEval-2022 Task 11 on Multilingual Complex Named Entity Recognition MULTICONER.[1] Divided into 13 tracks, the task focused on methods to identify complex named entities (like media titles, products, and groups) in 11 languages in both monolingual and multi-lingual scenarios. Eleven tracks were for building monolingual NER models for individual languages, one track focused on multilingual models able to work on all languages, and the last track featured code-mixed texts within any of these languages. The task used the MULTICONER dataset, composed of 2.3 million instances in Bangla, Chinese, Dutch, English, Farsi, German, Hindi, Korean, Russian, Spanish, and Turkish. Results showed that methods fusing external knowledge into transformer models achieved the best performance. The largest gains were on the Creative Work and Group entity classes, which are still challenging even with external knowledge. MULTICONER was one of the most popular tasks in SemEval-2022 and it attracted 377 participants during the practice phase. The final test phase had 236 participants, and 55 teams submitted their systems.

1 Introduction

Processing complex and ambiguous Named Entities (NEs) is a challenging NLP task in practical and open-domain settings but has not received sufficient attention from the research community. Complex NEs, like the titles of creative works (movie/book/song/software names) are not simple nouns and are harder to recognize (Ashwini and Choi, 2014). They can take the form of any linguistic constituent, like an imperative clause ("Dial M for Murder"), and do not look like traditional NEs (Person names, locations, organizations). This ambiguity makes it challenging to recognize them based on their context. Such titles can also be semantically ambiguous, e.g. "On the Beach" can be a preposition or refer to a movie.[2] Finally, such entities usually grow at a faster rate than traditional categories, and emerging entities pose yet another challenge.

Neural models (e.g. Transformers) have produced high scores on benchmark datasets like CoNLL03/OntoNotes (Devlin et al., 2018). However, as noted by Augenstein et al. (2017), these scores are driven by the use of well-formed news text, the presence of "easy" entities (e.g. person names), and memorization due to entity overlap between train/test sets; these models perform significantly worse on complex/unseen entities (Meng et al., 2021; Fetahu et al., 2021). Researchers using NER on downstream tasks have also noted that a significant proportion of their errors are due to NER systems failing to recognize complex entities (Luken et al., 2018; Hanselowski et al., 2018). Examples of such challenges are highlighted in Table 1.

For this task, we created the MULTICONER dataset (Malmasi et al., 2022) to address the aforementioned challenges. MULTICONER provides data from three domains (Wikipedia sentences, questions, and search queries) across 11 different languages, which are used to define 11 monolingual subsets of the shared task. Additionally, the dataset has multilingual and code-mixed subsets.

We received 1,884 submissions from 55 teams during the test phase and 34 system description papers were submitted. Results showed that usage of external data and ensemble strategies played a crucial role in the strong performance on in-domain data and also contributed to domain adaptation. External knowledge brought large improvements on classes containing names of creative works and groups, allowing these systems to achieve the best overall performance.

*These authors contributed equally to this work.
[1]https://multiconer.github.io/
[2]https://www.imdb.com/title/tt0053137

Challenge	Description
Complex Entities Relevant to all domains	Not all entities are proper names: some types (e.g. creative works) can be linguistically complex. They can be complex noun phrases (`Eternal Sunshine of the Spotless Mind`), gerunds (`Saving Private Ryan`), infinitives (`To Kill a Mockingbird`), or full clauses (`Mr.Smith Goes to Washington`). Syntactic parsing of such nouns is hard, and most current parsers/NER systems fail to recognize them. The top system from WNUT 2017 achieved 8% recall for creative work entities (Aguilar et al., 2017). Effective evaluation requires corpora with many such entities.
Ambiguous Entities and Contexts Particularly for voice and search domains	Some NEs are ambiguous: they are not always entities, e.g. "`Inside Out`", "`Among Us`", and "`Bonanza`" may refer to NEs (a movie, video game, and TV show) in some contexts, but not in others. Such NEs often resemble regular syntactic constituents. News texts have long sentences discussing many entities, but other use cases (search queries, questions) have shorter inputs. Data with minimal context is needed to assess performance of such use cases. Capitalization/punctuation features are large drivers of success in NER (Mayhew et al., 2019), but short inputs (ASR, queries) often lack such surface features. An uncased evaluation is needed to assess model performance.
Emerging Entities For domains with growing entities	All entity types are open classes (new ones are added), but some groups have a faster growth rate, e.g. new books/songs/movies are released weekly resulting in a long-tail distribution. Assessing true generalization requires test sets with many unseen entities, to mimic an open-world setting.

Table 1: Challenges not tackled by current work/datasets, but addressed by the MULTICONER task and data.

2 MultiCoNER Dataset

The MULTICONER dataset was designed to address the NER challenges described in §1. It represents three domains (wiki sentences, questions, and search queries) and includes 11 languages, plus multilingual and code-mixed subsets. For a detailed description of the MultiCoNER dataset, we refer the reader to the dataset paper (Malmasi et al., 2022). The dataset is publicly available.[3]

2.1 NER Taxonomy

MULTICONER leverages the WNUT 2017 (Derczynski et al., 2017a) taxonomy entity types, which defines the following NER tag-set with six classes:

1. PER: Names of people
2. LOC: Location or physical facilities
3. CORP: Corporations and businesses
4. GRP: All other groups
5. PROD: Consumer products
6. CW: Titles of creative works like movie, song, and book titles

This taxonomy allows us to capture a wide array of entities, including those with more complex entity structures, such as creative works.

2.2 Languages and Subsets

Eleven languages are included in MULTICONER:

1. Bangla (`BN`)
2. Chinese (`ZH`)
3. Dutch (`NL`)
4. English (`EN`)
5. Farsi (`FA`)
6. German (`DE`)
7. Hindi (`HI`)
8. Korean (`KO`)
9. Russian (`RU`)
10. Spanish (`ES`)
11. Turkish (`TR`)

[3]`https://registry.opendata.aws/multiconer`

These languages were chosen to include a diverse typology of languages and writing systems, and range from well-resourced (`EN`) to low-resourced ones (`FA`).

MULTICONER contains 13 different subsets: 11 monolingual subsets for the above languages, a multilingual subset (denoted as `MULTI`), and a code-mixed one (`MIX`).

Monolingual Subsets Each of the 11 languages has its own subset, which includes data from all three domains.

Multilingual Subset This contains randomly sampled data from all the languages mixed into a single subset. This subset is designed for evaluating multilingual models, and should ideally be used under the assumption that the language for each sentence is unknown.

Code-mixed Subset This subset contains code-mixed instances, where the entity is from one language and the rest of the text is written in another language. Like the multilingual subset, this subset should also be used under the assumption that the languages present in an instance are unknown.

Class	Split	EN	DE	ES	RU	NL	KO	FA	ZH	HI	TR	BN	MULTI	MIX
PER	Train	5,397	5,288	4,706	3,683	4,408	4,536	4,270	2,225	2,418	4,414	2,606	43,951	296
	Dev	290	296	247	192	212	267	201	129	133	231	144	2,342	96
	Test	55,682	55,757	51,497	44,687	49,042	39,237	35,140	26,382	25,351	26,876	24,601	111,346	19,313
LOC	Train	4,799	4,778	4,968	4,219	5,529	6,299	5,683	6,986	2,614	5,804	2,351	54,030	325
	Dev	234	296	274	221	299	323	324	378	131	351	101	2,932	108
	Test	59,082	59,231	58,742	54,945	63,317	52,573	45,043	43,289	31,546	34,609	29,628	141,013	23,111
GRP	Train	3,571	3,509	3,226	2,976	3,306	3,530	3,199	713	2,843	3,568	2,405	32,846	248
	Dev	190	160	168	151	163	183	164	26	148	167	118	1,638	75
	Test	41,156	40,689	38,395	37,621	39,255	31,423	27,487	18,983	22,136	21,951	19,177	77,328	16,357
CORP	Train	3,111	3,083	2,898	2,817	2,813	3,313	2,991	3,805	2,700	2,761	2,598	32,890	294
	Dev	193	165	141	159	163	156	160	192	134	148	127	1,738	112
	Test	37,435	37,686	36,769	35,725	35,998	30,417	27,091	25,758	21,713	21,137	20,066	75,764	18,478
CW	Train	3,752	3,507	3,690	3,224	3,340	3,883	3,693	5,248	2,304	3,574	2,157	38,372	298
	Dev	176	189	192	168	182	196	207	282	113	190	120	2,015	102
	Test	42,781	42,133	43,563	39,947	41,366	33,880	30,822	30,713	21,781	23,408	21,280	89,273	20,313
PROD	Train	2,923	2,961	3,040	2,921	2,935	3,082	2,955	4,854	3,077	3,184	3,188	35,120	316
	Dev	147	133	154	151	138	177	157	274	169	158	190	1,848	117
	Test	36,786	36,483	36,782	36,533	36,964	29,751	26,590	28,058	22,393	21,388	20,878	75,871	20,255
#sentences	Train	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	168,300	1,500
	Dev	800	800	800	800	800	800	800	800	800	800	800	8,800	500
	Test	217,818	217,824	217,887	217,501	217,337	178,249	165,702	151,661	141,565	136,935	133,119	471,911	100,000

Table 2: MultiCoNER dataset statistics for the different languages for the Train/Dev/Test splits. For each NER class we show the total number of entity instances per class on the different data splits. The bottom three rows show the total number of sentences for each language.

2.3 Dataset Creation

The MultiCoNER dataset consists of 11 languages, and three domains (encyclopedia sentences, questions from QA, and Web queries). A detailed overview of the MultiCoNER dataset is provided in the dataset paper (Malmasi et al., 2022).

Lowner: represents the encyclopedic sentences extracted from the different localized versions of Wikipedia. We select low-context sentences and the *interlinked* entities are resolved to the *entity types* using Wikidata as a reference, according to the NER class taxonomy from (Derczynski et al., 2017b). Manual inspection of 400 sampled English sentences shows that the NER gold labels are 94% accurate.

Msq-Ner: from the MS-MARCO Q&A corpus (Bajaj et al., 2016) *question templates* are extracted by replacing the *entities* with their NER *type* (from the MultiCoNER NER taxonomy). Entities in a question are identified using spaCy.[4] The templates are translated from English into the rest of the languages.

Orcas-Ner: similar ot Msq-Ner, templates from Web user queries are extracted from the ORCAS dataset (Craswell et al., 2020). The templates are translated into the respective languages, and finally, multiple instances are constructed from each template by simply replacing the template slots with actual named entities in the target languages.

[4]https://spacy.io/

2.4 Dataset Statistics

Table 2 shows some statistics of the dataset. For all the tracks, we have released 15,300 training and 800 development instances. In the training splits, the absolute majority of instances are from the Wikipedia domain (i.e. Lowner), whereas a small number of 100 instances are domain-adaptation data, with 50 instances coming from the Web Questions (i.e. Msq-Ner) and Web Query (i.e. Orcas-Ner) domains, respectively.

The test splits on the other hand are much larger. This is done for mainly two reasons: (1) to be able to assess the generalizability of NER models on unseen and complex entities, and (2) to assess the cross-domain adaptation performance of NER models. For practical reasons, we cap the number of test instances to be at a maximum of 200k per subset, with the exception of the Code-Mixed and Multilingual subsets. The Multilingual test split was generated from the language-specific test splits and was downsampled to contain only 471k instances. On the other hand, for the Code-Mixed subset, we sample test sentences from the language-specific test split, and replace the original entity surface forms with the surface form of the entity in another language, picked at random.

More details on the dataset construction process are available in Malmasi et al. (2022).

3 Task Description and Evaluation

The shared task is composed of 11 monolingual and 2 multilingual tracks. The monolingual tracks invited participants to build monolingual models for 11 languages addressed by the shared task. The multilingual track invited multilingual models capable of identifying entities from monolingual texts from any of the 11 languages. The code-mixed track called for models to identify entities in mixed-language texts (any language pair from the 11 languages). That means the multilingual models for multilingual and code mixed tracks should be able to process texts from any language and show competitive performance for all the languages.

We used the macro-averaged F1 scores to evaluate and rank systems. Additionally, we report precision, recall, and per-domain performance.

4 Baseline System

We train and evaluate a baseline NER system using on XLM-RoBERTa (XLM-R) (Conneau et al., 2020), a multilingual Transformer model. The XLM-R model computes a representation for each token, which is then used to predict the token tag using a CRF classification layer (Sutton et al., 2012).

The XLM-R baseline is highly suited for multilingual application scenarios, such as our. It supports up to 100 languages and provides a solid baseline upon which the participants can build. The baseline was trained with a learning rate of $2e-5$ and a maximum number of 50 epochs, with an early stopping criterion of a non-decreasing validation loss for 5 epochs. The code and scripts for the baseline system were provided to the participants to use its functionalities and further extend it with their approaches.[5]

5 Participating Systems and Results

We have received submissions from 55 different teams. Among the monolingual tracks, we have observed the highest participation of 30 teams in the `English` track. Ordered by the number of participating teams, the other monolingual tracks are `Chinese` (21), `Bangla` (18), `Spanish` (18), `Hindi` (17), `Korean` (17), `German` (16), `Dutch` (15), `Farsi` (15), `Turkish` (15), and `Russian` (14). The number of participating teams for the Multilingual and Code-mixed tracks are 25 and 21, respectively. Table 3 shows the final rankings for all tracks. Detailed performance breakdown is available in Appendix A.

Most of the top-performing teams aimed at building their system targeting the multilingual track, and then retrained it for the other tracks separately and made submissions to all the 13 tracks. Therefore, in the rest of this section, we will first discuss the approaches by focusing on the multilingual track. Then, we will discuss teams that built their systems for one or more monolingual tracks. Finally, we will summarize the methods (e.g. language models, toolkits) and resources used.

5.1 Top Multilingual Systems

DAMO-NLP (Wang et al., 2022) ranked 1st in the multilingual (`MULTI`) track and all the monolingual tracks except `BN` (2nd) and `ZH` (4th). Given a text, they used a knowledge retrieval module to retrieve K most relevant paragraphs from a knowledge base (i.e. Wikipedia). Paragraphs were concatenated together with the input, and token representations were passed through a CRF to predict the labels. They employed multiple such XLM-RoBERTa models with random seeds and then used a voting strategy to make the final prediction.

USTC-NELSLIP (Chen et al., 2022a) ranked 1st in three tracks (`MIX`, `ZH`, `BN`), and 2nd for all the other tracks. The average performance gap between USTC-NELSLIP and DAMO-NLP is $\approx$3% for all the 13 tracks. USTC-NELSLIP aimed at fine-tuning a Gazetteer enhanced BiLSTM network in such a way that the representation produced for an entity has similarity with the representation produced by a pre-trained language model (LM). They developed a two-step process with two parallel networks, where a Gazetteer-BiLSTM uses a Gazetteer search to produce one-hot labels for each token in a given text and a BiLSTM produces a dense vector representation for each token. Another network uses a frozen XLM-RoBERTa to produce an embedding vector for each token. A KL divergence loss is applied to make the Gazetteer network's output similar to the LM. These two networks are jointly trained together again and their outputs are fused together for the final prediction.

QTrade AI (Gan et al., 2022) ranked 3rd in `MULTI`, 4th in `MIX`, and 8th in `ZH`. They used an XLM-RoBERTa encoder and applied sample mixing for data augmentation, along with adversarial training through data noising. For the multilingual

[5] https://github.com/amzn/multiconer-baseline

English (EN)

	Team	F1
1	DAMO-NLP	0.912
2	USTC-NELSLIP	0.855
3	PAI	0.784
4	ML-HUB	0.781
5	RACAI	0.758
6	Infrrd.ai	0.747
7	EURECOM	0.746
8	Sliced	0.745
9	MaChAmp	0.745
10	Raccoons	0.742
11	YNUNLP	0.732
12	LMN	0.725
13	brotherhood	0.724
14	L3i	0.72
15	Multilinguals	0.717
16	KDDIE	0.717
17	MarSan	0.715
18	Cardiff NLP	0.709
19	Lone Wolf	0.698
20	MIDAS	0.696
21	UC3M-PUCPR	0.692
22	CSECU-DSG	0.692
23	Sartipi-Sedighin	0.675
24	Enigma	0.672
25	DANGNT-SGU	0.669
26	AaltoNLP	0.668
27	SPDB I.L.	0.651
28	silpa_nlp	0.634
29	B.E.P.	0.632
30	**BASELINE**	0.614
31	AutoNER	0.557

Spanish (ES)

	Team	F1
1	DAMO-NLP	0.899
2	USTC-NELSLIP	0.854
3	RACAI	0.756
4	Infrrd.ai	0.753
5	MaChAmp	0.752
6	Sliced	0.751
7	YNUNLP	0.732
8	brotherhood	0.707
9	L3i	0.689
10	PA Ph&Tech	0.689
11	MarSan	0.683
12	SPDB I.L.	0.673
13	CSECU-DSG	0.656
14	EURECOM	0.628
15	Multilinguals	0.612
16	Sartipi-Sedighin	0.607
17	B.E.P.	0.601
18	**BASELINE**	0.578
19	UC3M-PUCPR	0.568

Dutch (NL)

	Team	F1
1	DAMO-NLP	0.905
2	USTC-NELSLIP	0.877
3	RACAI	0.784
4	Sliced	0.777
5	MaChAmp	0.77
6	Infrrd.ai	0.764
7	YNUNLP	0.758
8	brotherhood	0.73
9	PA Ph&Tech	0.721
10	MarSan	0.711
11	L3i	0.71
12	CSECU-DSG	0.679
13	EURECOM	0.667
14	B.E.P.	0.632
15	**BASELINE**	0.62
16	Sartipi-Sedighin	0.584

Russian (RU)

	Team	F1
1	DAMO-NLP	0.915
2	USTC-NELSLIP	0.838
3	RACAI	0.746
4	Sliced	0.737
5	YNUNLP	0.73
6	MaChAmp	0.724
7	brotherhood	0.703
8	NetEase.AI	0.698
9	EURECOM	0.682
10	MarSan	0.675
11	L3i	0.667
12	CSECU-DSG	0.631
13	B.E.P.	0.6
14	**BASELINE**	0.596
15	AutoNER	0.527

Turkish (TR)

	Team	F1
1	DAMO-NLP	0.887
2	USTC-NELSLIP	0.855
3	SU-NLP	0.72
4	RACAI	0.704
5	Sliced	0.688
6	MaChAmp	0.676
7	YNUNLP	0.668
8	ML-HUB	0.658
9	L3i	0.643
10	MarSan	0.611
11	brotherhood	0.597
12	EURECOM	0.566
13	CSECU-DSG	0.553
14	Sartipi-Sedighin	0.527
15	**BASELINE**	0.463
16	B.E.P.	0.45

Korean (KO)

	Team	F1
1	DAMO-NLP	0.886
2	USTC-NELSLIP	0.864
3	RACAI	0.717
4	CMB AI Lab	0.707
5	Sliced	0.707
6	YNUNLP	0.703
7	C-3PO	0.675
8	UA-KO	0.675
9	brotherhood	0.674
10	Infrrd.ai	0.673
11	MaChAmp	0.654
12	EURECOM	0.65
13	L3i	0.627
14	MarSan	0.623
15	CSECU-DSG	0.621
16	AaltoNLP	0.618
17	B.E.P.	0.59
18	**BASELINE**	0.552

Farsi (FA)

	Team	F1
1	DAMO-NLP	0.897
2	USTC-NELSLIP	0.871
3	RACAI	0.704
4	Sliced	0.687
5	YNUNLP	0.672
6	brotherhood	0.657
7	C-3PO	0.655
8	L3i	0.651
9	MarSan	0.621
10	MaChAmp	0.607
11	AaltoNLP	0.589
12	Sartipi-Sedighin	0.577
13	EURECOM	0.559
14	CSECU-DSG	0.558
15	**BASELINE**	0.522
16	B.E.P.	0.513

German (DE)

	Team	F1
1	DAMO-NLP	0.906
2	USTC-NELSLIP	0.89
3	RACAI	0.794
4	Sliced	0.789
5	MaChAmp	0.784
6	YNUNLP	0.773
7	L3i	0.772
8	ML-HUB	0.761
9	brotherhood	0.759
10	Infrrd.ai	0.759
11	EURECOM	0.744
12	MarSan	0.731
13	CSECU-DSG	0.725
14	AaltoNLP	0.714
15	PA Ph&Tech	0.667
16	B.E.P.	0.666
17	**BASELINE**	0.637

Chinese (ZH)

	Team	F1
1	USTC-NELSLIP	0.817
2	CASIA	0.797
3	OPDAI	0.795
4	DAMO-NLP	0.781
5	NetEase.AI	0.778
6	CMB AI Lab	0.764
7	NCUEE-NLP	0.742
8	QTrade AI	0.74
9	CSECU-DSG	0.672
10	Multilinguals	0.669
11	L3i	0.669
12	Sliced	0.652
13	Infrrd.ai	0.647
14	MaChAmp	0.638
15	EURECOM	0.634
16	RACAI	0.627
17	YNUNLP	0.614
18	brotherhood	0.609
19	MarSan	0.566
20	SPDB I.L.	0.557
21	B.E.P.	0.528
22	**BASELINE**	0.513

Hindi (HI)

	Team	F1
1	DAMO-NLP	0.862
2	USTC-NELSLIP	0.846
3	RACAI	0.681
4	Sliced	0.67
5	NetEase.AI	0.666
6	Infrrd.ai	0.657
7	brotherhood	0.642
8	YNUNLP	0.634
9	OPDAI	0.629
10	MaChAmp	0.617
11	CSECU-DSG	0.577
12	MarSan	0.563
13	EURECOM	0.528
14	silpa_nlp	0.515
15	B.E.P.	0.499
16	L3i	0.497
17	Enigma	0.486
18	**BASELINE**	0.482

Bangla (BN)

	Team	F1
1	USTC-NELSLIP	0.842
2	DAMO-NLP	0.835
3	NetEase.AI	0.709
4	RACAI	0.663
5	Infrrd.ai	0.64
6	YNUNLP	0.638
7	Sliced	0.63
8	Team Atreides	0.598
9	brotherhood	0.586
10	MaChAmp	0.565
11	MarSan	0.542
12	EURECOM	0.526
13	AaltoNLP	0.518
14	silpa_nlp	0.514
15	CSECU-DSG	0.505
16	B.E.P.	0.451
17	L3i	0.448
18	Enigma	0.427
19	**BASELINE**	0.394

Multilingual (MULTI)

	Team	F1
1	DAMO-NLP	0.853
2	USTC-NELSLIP	0.853
3	QTrade AI	0.777
4	SeqL	0.755
5	CMB AI Lab	0.737
6	UM6P-CS	0.725
7	RACAI	0.721
8	Cardiff NLP	0.717
9	Sliced	0.711
10	IIE_KDSEC	0.709
11	B.E.P.	0.707
12	OPDAI	0.695
13	brotherhood	0.694
14	MarSan	0.693
15	Infrrd.ai	0.692
16	HaveNoIdea	0.688
17	EURECOM	0.681
18	MaChAmp	0.677
19	YNUNLP	0.668
20	DS4DH	0.652
21	UPB	0.647
22	CSECU-DSG	0.644
23	NSU-AI	0.642
24	SPDB I.L.	0.632
25	L3i	0.612
26	**BASELINE**	0.478

Code-Mixed (MIX)

	Team	F1
1	USTC-NELSLIP	0.929
2	DAMO-NLP	0.918
3	CMB AI Lab	0.846
4	QTrade AI	0.844
5	SeqL	0.803
6	IIE_KDSEC	0.796
7	RACAI	0.794
8	UM6P-CS	0.792
9	EURECOM	0.776
10	OPDAI	0.775
11	YNUNLP	0.768
12	UC3M-PUCPR	0.764
13	brotherhood	0.759
14	MaChAmp	0.745
15	Sliced	0.727
16	CMNEROne	0.704
17	L3i	0.687
18	Cardiff NLP	0.681
19	B.E.P.	0.68
20	SPDB I.L.	0.673
21	MarSan	0.67
22	CSECU-DSG	0.64
23	**BASELINE**	0.581

Table 3: Ranking for all of the tracks based on Macro F1. Full forms of the team names “B.E.P.” and “SPDB I.L.” are BaselineExtendinPokemons and SPDB Innovation Lab, respectively.

track, they leveraged an architecture with shared and per-language representations. Finally, they created an ensemble of models trained with different approaches.

SeqL (Hassan et al., 2022) ranked 4^{th} in MULTI 5^{th} in MIX. They train seven XLM-RoBERTa-large and Infoxlm-large models and then used an ensemble approach with voting and score fusion to predict the final labels. They found that the ensemble approach is slightly better than the best single model, and score fusion worked better than simple voting.

CMB AI Lab (PU et al., 2022) ranked 5^{th} in MULTI, 3^{rd} in MIX, 4^{th} in KO, and 6^{th} in ZH. They first utilized a biaffine layer to identify potential entity spans in a sentence, and the extracted spans are then processed with another classifier to obtain their class label. Finally, an ensemble is created by combining different pre-trained encoders and data augmentation techniques based on translations of the original training data. In terms of pre-trained LMs, they used XLM-RoBERTa and mT5.

5.2 Other Noteworthy Systems

RACAI (Pais, 2022) (3^{rd} in ES, NL, RU, KO, FA, DE, HI; 4^{th} in BN, TR; 5^{th} in EN; 7^{th} in MULTI, MIX; 16^{th} in ZH) used XLM-RoBERTa as pre-trained LM and a lateral inhibition layer inspired by the biological mechanism of lateral inhibition. They achieved strong performance in most of the tracks without using any external data.

Sliced (Plank, 2022) (4^{th} in NL, RU, FA, DE, HI; 5^{th} in KO, TR; 6^{th} in ES; 7^{th} in BN; 8^{th} in EN; 9^{th} in MULTI; 12^{th} in ZH; 15^{th} in MIX) used the MaChAmp toolkit (van der Goot et al., 2021) that enables easy exchange of pre-trained LMs for fine-tuning as well as multi-task learning. Within this framework, they have experimented with four different pre-trained LMs and found that XLM-RoBERTa is more efficient for training their system and provides stronger performance.

MaChAmp (van der Goot, 2022) (5^{th} in DE, ES, NL; 6^{th} in RU, TR; 9^{th} in EN; 10^{th} in FA, BN, HI; 11^{th} in KO; 14^{th} in ZH, MIX; 18^{th} in MULTI) first trained a multi-task model on 7 SemEval tasks and then fine-tuned for each task individually. They report that such a multi-tasking and fine-tuning approach is beneficial for a subset of the tasks.

OPDAI (Chen et al., 2022b) (3^{rd} in ZH; 9^{th} in HI; 10^{th} in MIX; 12^{th} in MULTI) used a hybrid technique with multiple stages involving model ensemble using neural model, soft templates, and Wikipedia lexicons. Their strong performance in ZH is powered by RoBERTa-wwm (Cui et al., 2021) pre-trained on Chinese data and Chinese word embeddings (Song et al., 2018).

CASIA (Fu et al., 2022) only participated in ZH and ranked 2^{nd}. They built a hybrid system based on RoBERTa-wwm and used three training mechanisms (adversarial training, child-Tuning training, and continued pre-training). Additionally, they performed a series of data augmentation steps.

PAI (Ma et al., 2022) (3^{rd} in EN) used string matching to retrieve entities with types from the LUKE entity dictionary (Yamada et al., 2020) for a given text. Then they concatenated the entity information with the input text and fed it to a pre-trained BERT model to build the NER system.

SU-NLP (Çarık et al., 2022) only participated in TR and ranked 3^{rd}. Given an input text, they query an information retrieval (IR) system that indexes Wikipedia articles. Retrieved documents are used as context and a Turkish BERT variant (BERTurk) is used to encode the context and candidate mentions, with classifier heads for NER.

Infrrd.ai (He et al., 2022) participated in nine tracks (EN, ES, NL, KO, DE, ZH, HI, BN, MULTI) and their best rank is 4^{th} for ES. They trained a multilingual model with an XLM-RoBERTa base encoder, whose embeddings were passed into a BiLSTM encoder, which finally passed the encoded tokens to a CRF layer for classification. They also used an ensemble strategy with majority voting.

UM6P-CS (Mekki et al., 2022) ranked 6^{th} in MULTI and 8^{th} in MIX. They introduced several self-training and auxiliary tasks that aim to improve NER classification performance on top of XLM-RoBERTa. The auxiliary task of span classification focused on addressing the mention detection performance of the model, which essentially ensures that the model has good coverage of all named entities, regardless of their type. In terms of self-training, the authors predicted weak labels on the unlabelled test set and concatenated both datasets into one. The impact of self-training seems to have a significant impact with 3% improvement in terms of Precision, and 2.24% in terms of F1 score.

Multilinguals (Pandey et al., 2022a,b) participated in EN, ES, and ZH and best rank is 10^{th} in ZH. They applied a BERT encoder with different classification heads: a linear layer, a CRF layer, and a BiLSTM-CRF. BERT and linear approach worked best for EN. For ES and ZH, they pre-trained BERT using the Whole Word Masking (WWM) learning objective over Wikipedia data and the CRF classification head worked best for these tracks.

L3i (Boros et al., 2022) participated in all tracks and best rank is 7th in DE. They used Sentence-BERT (Reimers and Gurevych, 2019) to retrieve the most similar sentence from the training set and used it as context by adding it to the test text. Their model consists of a BERT encoder with a Transformer layer and a CRF head for classification.

MarSan (Tavan and Najafi, 2022) participated in all tracks and best rank is 9th in FA. They used T5 (monolingual and multilingual) to create feature vector for an input text. Then they performed a subtoken check step to mark the first subword as 1 and others as 0 (Subtoken check increased 4% F1). At the final stage, a Transformer layer is followed by a token prediction layer to perform NER.

TEAM-Atreides (Tasnim et al., 2022) only participated in BN and ranked 8th. They used an ensemble of mono-lingual ELECTRA-based models with majority voting. They also used data augmentation using translation and conducted experiments with non-contextual word embeddings.

UA-KO (Song and Bethard, 2022) ranked 8th in the KO track. They used GeoNames and the Encyclopedia of Korean Culture to incorporate entity names in the training set. Their model uses an ensemble approach with a soft-voting mechanism, combining the monolingual and multilingual models' predictions.

CSECU-DSG (Aziz et al., 2022) participated in all tracks and the best rank is 9th for ZH. The authors propose two approaches: (1) a BiLSTM-CRF that leverages stacked token embeddings from different sources, and (2) a Transformer-based encoder with a feed-forward classification head.

PA Ph&Tech (Lin et al., 2022) participated in ES, DE, and NL and best rank is 9th for NL. They used ensemble embedding from multiple transformers and reinforcement learning was also applied to maximize model accuracy. In an additional setting (Hou et al., 2022), they experimented with an ensemble approach, where they leveraged multiple transformers by assigning different weights in the transformer layers. Meanwhile, data augmentation is also applied to enlarge the training data.

Raccoons (Dogra et al., 2022) ranked 10th in the EN track. They focused on improving word representations for NER through a reinforcement trainer. This was done through a task model and controller that repeatedly interact to update the embeddings.

AaltoNLP (Pietiläinen and Ji, 2022) participated in five tracks (EN, DE, FA, BN, KO) and the best rank is 11th for FA. Their approach consists of an ensemble strategy where they train two encoders jointly, allowing the models to combine the scores from the different encoders via a linear layer. Different models used different random seeds.

LMN (Lai, 2022) ranked 12th in the EN track. They applied a transfer-based encoder with a feed-forward classification head with a CRF layer. Their best variant used the ALBERT-xxlarge model. They also experimented with entity linking with Wikipedia and augmenting data with entities of the same type.

UC3M-PUCPR (Schneider et al., 2022) participated in EN, ES, MIX, and their best rank is 12th for MIX. They have used an ensemble of language-specific pre-trained LMs with soft-voting to make the final predictions.

NamedEntityRangers (Miftahova et al., 2022) ranked 16th in the MULTI track. They used RemBERT and mT5 to experiment with two approaches, where the first approach is the classical token classification method and the second method uses a template-free paradigm in which an encoder-decoder model translates the input sequence of words to a special output, encoding named entities with the predefined label.

CMNEROne (Dowlagar and Mamidi, 2022) ranked 16th in MIX. Their approach involves fine-tuning multilingual BERT on code-mixed data. To learn language-agnostic features, they pre-trained the model for a downstream task of language identification using the multilingual dataset.

KDDIE (Martin et al., 2022) only participated in the EN track and ranked 16th. They experimented by fine-tuning BERT and DeBERTa-based models and their best system is a fine-tuned DeBERTa-XLarge model.

DS4DH (Rouhizadeh and Teodoro, 2022) ranked 20th in the MULTI track. Their approach involves fine-tuning different pre-trained LMs (Multilingual-BERT, XLM-RoBERTa-base, XLM-RoBERTa-Large, Distilbert-Multilingual) with different classification heads like CRF and fully-connected layer.

NCUEE-NLP (Lee et al., 2022) ranked 7th in the ZH track. They used external data collected from MSRA, Weibo, People Daily, Boson,

Class	Baseline	DAMO-NLP	USTC-NELSLIP	QTrade AI
PER	63.88	92.07 (+28)	90.76 (+27)	87.20 (+23)
LOC	51.87	86.52 (+35)	86.81 (+35)	80.79 (+29)
CORP	49.61	84.55 (+35)	87.86 (+38)	77.23 (+28)
PROD	44.36	84.32 (+40)	81.05 (+37)	75.23 (+31)
GRP	39.28	79.90 (+41)	81.52 (+42)	71.66 (+32)
CW	37.68	84.49 (+47)	83.81 (+46)	73.85 (+36)

Table 4: F1 scores of the baseline and top three systems in the MULTI track for each class.

CLUNER, and LG, and trained a BiLSTM-CRF model with embeddings from a BERT model pre-trained on Chinese data.

DANGNT-SGU (Nguyen and Huynh, 2022) ranked 25th in the EN track by fine-tuning RoBERTa on the training data.

silpa_nlp (Singh et al., 2022) ranked 14th in HI and BN by fine-tuning XLM-R on the training set.

6 Insights from the Systems

6.1 Advancing the State of the Art

Identifying Complex Entities From the ranking in Table 3, we see that almost all the teams could outperform the official baseline system described in Section 4 in all the tracks. For most of the tracks, the top two teams DAMO-NLP and USTC-NELSLIP's performance gap is very small compared to third place. To btter understand this difference, we look at per-class performance. In Table 4, we show per-class F1 scores for the top three teams in the MULTI track. Although the systems performed better than the official baseline by a large margin, complex entities like creative works, products, and groups are still the most difficult ones to identify. This analysis shows that the largest gains by the top systems leveraging external knowledge came from classes containing complex NEs, e.g. CW and GRP.

Domain Adaptation The official baseline system performed poorly in terms of domain adaptation and achieved much lower F1 in MSQ-NER and ORCAS-NER compared to LOWNER. Intuitively, augmenting the training data with interrogative sentences could be a way to perform better in these domains. However, we observe that the participants could overcome the challenge of domain adaptation without especially including questions and queries

Multilingual
XLM-RoBERTa (**XLM-R;** Conneau et al. (2020)) **:** DAMO-NLP, USTC-NELSLIP, QTrade AI, SeqL, CMB AI Lab, RACAI, Sliced, Infrrd.ai, UM6P-CS, UA-KO, CSECU-DSG, PA Ph&Tech, Raccoons, AaltoNLP, UC3M-PUCPR, DS4DH, silpa_nlp
mT5 (Xue et al., 2021)**:** CMB AI Lab, MarSan, NamedEntityRangers
mBERT (Devlin et al., 2019)**:** Sliced, L3i, PA Ph&Tech, UC3M-PUCPR, CMNEROne, DS4DH
RemBERT (Chung et al., 2021)**:** Sliced, NamedEntityRangers
English
BERT (Devlin et al., 2019)**:** PAI, Multilinguals, CSECU-DSG, PA Ph&Tech, Raccoons, UC3M-PUCPR, KDDIE
BigBird RoBERTa (Zaheer et al., 2021)**:** L3i
T5 (Raffel et al., 2020)**:** MarSan
XLNet (Yang et al., 2019)**:** PA Ph&Tech
ALBERT (Lan et al., 2020)**:** LMN
RoBERTa (Liu et al., 2019)**:** UC3M-PUCPR, DANGNT-SGU
DistillBERT (Sanh et al., 2019), **ELECTRA** (Clark et al., 2020)**:** UC3M-PUCPR
DeBERTa (He et al., 2021)**:** KDDIE
Spanish
Spanish BERT (Canete et al., 2020)**:** Multilinguals
BERT-wwm: L3i
Beto (Cañete et al., 2020), **Spanish RoBERTa** (Gutiérrez-Fandiño et al., 2021)**:** UC3M-PUCPR
Chinese
RoBERTa-wwm (Cui et al., 2021)**:** OPDAI, CASIA, Multilinguals
BERT [6]**:** L3i, NCUEE-NLP
Korean
KoBERT [7], **Ko-ELECTRA**[8], **KR-BERT** (Lee et al., 2020), **KLUE-RoBERTa** (Park et al., 2021)**:** UA-KO
BERT [9] **:** L3i
Bangla
BanglaBERT (Bhattacharjee et al., 2022)**:** TEAM-Atreides
Dutch
BERT [10]**:** L3i
Farsi
ParsBERT (Farahani et al., 2020)**:** L3i
German
BERT [11]**:** L3i
Hindi
IndicBERT (Kakwani et al., 2020)**:** silpa_nlp
Russian
RuBERT: L3i
Turkish
BERTurk (Schweter, 2020)**:** SU-NLP, L3i

Table 5: Pre-trained Transformer language models used by the teams for different languages. BERT models for non-English languages are trained on the specific languages' data with BERT architecture by the community.

[6] https://huggingface.co/bert-base-chinese
[7] https://github.com/SKTBrain/KoBERT
[8] https://github.com/monologg/KoELECTRA
[9] https://huggingface.co/kykim/bert-kor-base
[10] https://huggingface.co/wietsedv/bert-base-dutch-cased
[11] https://huggingface.co/dbmdz/bert-base-german-uncased

in their external data. For example, DAMO-NLP found that their approach of retrieving Wikipedia paragraphs not only provided a strong performance on LOWNER, but also helped with cross-domain transferability.

Adapting to MSQ was easier compared to ORCAS-NER for all the tracks except Bangla. The top systems like DAMO-NLP and USTC-NELSLIP struggled in MSQ-NER for Bangla, while they typically had higher F1 scores for MSQ-NER than ORCAS-NER for the other tracks. This could be an interesting direction to explore in the future.

6.2 Other Insights

External Data In Section 5 we observe that such superior performance by these top systems became possible by exploiting external knowledge during learning and inference. While USTC-NELSLIP used knowledge from pre-trained language models to fine-tune Gazetteer presentations, DAMO-NLP directly used raw texts from Wikipedia to inject context and it gave them an advantage over USTC-NELSLIP in most tracks.

As the availability of external data is higher for English compared to other languages, most of the teams participating in other languages used publicly available pre-trained models for other languages, or translated data from other languages. For example, CASIA augmented data from other languages with translation, and it helped them to secure second place in the Chinese track. In general, a large portion of the participating teams showed that they can do better if they can go beyond the provided training data, and use external data or pre-trained language models for different languages to inject external knowledge in some way.

Modeling Approaches Almost all participating systems relied on publicly available Transformer (Vaswani et al., 2017) based pre-trained language models (Table 5). XLM-RoBERTa (a.k.a. XLM-R) was the most popular choice for building multilingual models. Most of the teams participating in non-English monolingual tracks preferred this particular model to the multilingual variant of BERT.

Other recent language models like T5, ELECTRA, XLNet, and ALBERT were used by some of the teams, but mostly for English. We observed that for non-English languages, many teams used community-developed pre-trained models for other languages like Chinese, Hindi, Spanish, Korean, Bangla, Turkish, Russian, Farsi, Dutch, and German. Most of such models are trained using the BERT architecture with data for the respective languages. A lot of teams relied on the strength of Conditional Random Field (CRF; Lafferty et al. 2001) for sequence labeling problems and adopted it to gain stronger performance. Very few teams used architectures like LSTMs.

Teams that simply fine-tuned pre-trained language models performed similarly to the baseline system for most of the tracks. Apart from the previously mentioned role of external data, another vital component for strong performance is using ensemble learning strategies. Almost all the strong performing teams trained multiple models and ensembled them for making the final predictions. We have also observed some teams experimenting with adversarial training and reinforcement learning.

7 Conclusion

In this paper, we have presented an overview of the SemEval shared task on identifying complex entities in multiple languages. In this shared task, we have received system submissions from 55 competing teams, and 34 system description papers. On average, the wining systems for all the tracks outperformed the baseline system by a large margin of 35% F1.

Most of the top-performing teams in MULTICONER utilized external knowledge bases like Wikipedia and Gazetteer. They also tend to use XLM-RoBERTa as the pre-trained language model. In terms of modeling approaches, ensemble strategies helped the systems to achieve strong performance. Results from the top teams indicate that identifying complex entities like creative works is still difficult among all the classes even with the usage of external data.

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153.

Sandeep Ashwini and Jinho D. Choi. 2014. Targetable named entity recognition in social media. *CoRR*, abs/1408.0782.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Abdul Aziz, Md. Akram Hossain, and Abu Nowshed Chy. 2022. CSECU-DSG at SemEval-2022 Task 11: Identifying the Multilingual Complex Named Entity in Text Using Stacked Embeddings and Transformer based Approach. In *The 16th International Workshop on Semantic Evaluation*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Abhik Bhattacharjee, Tahmid Hasan, Kazi Mubasshir, Md. Saiful Islam, Wasi Ahmad Uddin, Anindya Iqbal, M. Sohel Rahman, and Rifat Shahriyar. 2022. Banglabert: Lagnuage model pretraining and benchmarks for low-resource language understanding evaluation in bangla. In *Findings of the North American Chapter of the Association for Computational Linguistics: NAACL 2022*.

Emanuela Boros, Carlos-Emiliano González-Gallardo, Jose G Moreno, and Antoine Doucet. 2022. L3i at SemEval-2022 Task 11: Straightforward Additional Context for Multilingual Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

José Canete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. *Pml4dc at iclr*, 2020:2020.

Buse Çarık, Fatih Beyhan, and Reyyan Yeniterzi. 2022. SU-NLP at SemEval-2022 Task 11: Complex Named Entity Recognition with Entity Linking. In *The 16th International Workshop on Semantic Evaluation*.

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

Beiduo Chen, Jun-Yu Ma, Jiajun Qi, Wu Guo, Zhen-Hua Ling, and Quan Liu. 2022a. USTC-NELSLIP at SemEval-2022 Task 11: Gazetteer-Adapted Integration Network for Multilingual Complex Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Ze Chen, Kangxu Wang, Jiewen Zheng, Zijian Cai, Jiarong He, and Jin Gao. 2022b. OPDAI at SemEval-2022 Task 11: A hybrid approach for Chinese NER using outside Wikipedia knowledge. In *The 16th International Workshop on Semantic Evaluation*.

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2021. Rethinking embedding coupling in pre-trained language models. In *International Conference on Learning Representations*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 18 million clicked query-document pairs for analyzing search. *arXiv preprint arXiv:2006.05324*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 29:3504–3514.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017a. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017b. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text, NUT@EMNLP 2017, Copenhagen, Denmark, September 7, 2017*, pages 140–147. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*. Association for Computational Linguistics.

Atharvan Dogra, Prabsimran Kaur, Guneet Singh Kohli, and Jatin Bedi. 2022. Raccoons at SemEval-2022 Task 11: Leveraging Concatenated Word Embeddings for Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Suman Dowlagar and Radhika Mamidi. 2022. CMNEROne at SemEval-2022 Task 11: Code-Mixed Named Entity Recognition by leveraging multilingual data. In *The 16th International Workshop on Semantic Evaluation*.

Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, and Mohammad Manthouri. 2020. Parsbert: Transformer-based model for persian language understanding. *ArXiv*, abs/2005.12515.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Dynamic gazetteer integration in multilingual models for cross-lingual and cross-domain named entity recognition. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jia Fu, Zhen Gan, Zhucong Li, Sirui Li, Dianbo Sui, Yubo Chen, Kang Liu, and Jun Zhao. 2022. CASIA at SemEval-2022 Task 11: Chinese Named Entity Recognition for Complex and Ambiguous Entities. In *The 16th International Workshop on Semantic Evaluation*.

Weichao Gan, Yuanping Lin, Guangbo Yu, Guimin Chen, and Qian Ye. 2022. Qtrade AI at SemEval-2022 Task 11: An Unified Framework for Multilingual NER Task. In *The 16th International Workshop on Semantic Evaluation*.

Asier Gutiérrez-Fandiño, Jordi Armengol-Estapé, Marc Pàmies, Joan Llop-Palao, Joaquín Silveira-Ocampo, Casimiro Pio Carrino, Aitor Gonzalez-Agirre, Carme Armentano-Oller, Carlos Rodríguez Penagos, and Marta Villegas. 2021. Spanish language models. *CoRR*, abs/2107.07253.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

Fadi Hassan, Wondimagegnhue Tufa, Guillem Collell, Piek Vossen, Lisa Beinborn, Adrian Flanagan, and Kuan Eeik Tan. 2022. SeqL at SemEval-2022 Task 11: An Ensemble of Transformer Based Models for Complex Named Entity Recognition Task. In *The 16th International Workshop on Semantic Evaluation*.

JiangLong He, Akshay Uppal, Mamatha N, Shiv Vignesh, Deepak Kumar, and Aditya Kumar Sarda. 2022. Infrrd.ai at SemEval-2022 Task 11: A system for named entity recognition using data augmentation, transformer-based sequence labeling model, and EnsembleCRF. In *The 16th International Workshop on Semantic Evaluation*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Changyu Hou, Jun Wang, Yixuan Qiao, Peng Jiang, Peng Gao, Guotong Xie, Qizhi LIN, Xiaopeng Wang, Xiandi Jiang, Benqi Wang, and Qifeng Xiao. 2022. SFE-AI at SemEval-2022 Task 11: Low-Resource Named Entity Recognition using Large Pre-trained Language Models. In *The 16th International Workshop on Semantic Evaluation*.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ngoc Minh Lai. 2022. LMN at SemEval-2022 Task 11: A Transformer-based System for English Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Lung-Hao Lee, Chien-Huan Lu, and Tzu-Mi Lin. 2022. NCUEE-NLP at SemEval-2022 Task 11 Chinese Named Entity Recognition Using the BERT-biLSTM-CRF model. In *The 16th International Workshop on Semantic Evaluation*.

Sangah Lee, Hansol Jang, Yunmee Baik, Suzi Park, and Hyopil Shin. 2020. Kr-bert: A small-scale korean-specific language model. *ArXiv*, abs/2008.03979.

Qizhi Lin, Changyu Hou, Xiaopeng Wang, Jun Wang, Yixuan Qiao, Peng Jiang, Xiandi Jiang, Benqi Wang, and Qifeng Xiao. 2022. PA Ph&Tech at SemEval-2022 Task 11: NER Task with Ensemble Embedding from Reinforcement Learning. In *The 16th International Workshop on Semantic Evaluation*.

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

Jackson Luken, Nanjiang Jiang, and Marie-Catherine de Marneffe. 2018. QED: A fact verification system for the FEVER shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 156–160, Brussels, Belgium. Association for Computational Linguistics.

Long Ma, Xiaorong Jian, and Xuan Li. 2022. PAI at SemEval-2022 Task 11: Name Entity Recognition with Contextualized Entity Representations and Robust Loss Functions. In *The 16th International Workshop on Semantic Evaluation*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Caleb Martin, Huichen Yang, and William Hsu. 2022. Kddie at SemEval-2022 Task 11: Using DeBERTa for Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Stephen Mayhew, Tatiana Tsygankova, and Dan Roth. 2019. ner and pos when nothing is capitalized. In *EMNLP/IJCNLP (1)*, pages 6255–6260. Association for Computational Linguistics.

Abdellah El Mekki, Abdelkader El Mahdaouy, Mohammed Akallouch, Ismail Berrada, and Ahmed Khoumsi. 2022. UM6P-CS at SemEval-2022 Task 11: Enhancing Multilingual and Code-Mixed Complex Named Entity Recognition via Pseudo Labels using Multilingual Transformer. In *The 16th International Workshop on Semantic Evaluation*.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1499–1512. Association for Computational Linguistics.

Amina Miftahova, Alexander Pugachev, Artem Skiba, Katya Artemova, Tatiana Batura, Pavel Braslavski, and Vladimir V. Ivanov. 2022. Namedentityrangers at SemEval-2022 Task 11: Transformer-based Approaches for Multilingual Complex Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Dang Tuan Nguyen and Huy Khac Nguyen Huynh. 2022. DANGNT-SGU at SemEval-2022 Task 11: Using Pre-trained Language Model for Complex Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Vasile Pais. 2022. RACAI at SemEval-2022 Task 11: Complex named entity recognition using a lateral inhibition mechanism. In *The 16th International Workshop on Semantic Evaluation*.

Amit Pandey, Swayatta Daw, and Vikram Pudi. 2022a. Multilinguals at SemEval-2022 Task 11: Transformer Based Architecture for Complex NER. In *The 16th International Workshop on Semantic Evaluation*.

Amit Pandey, Swayatta Daw, Narendra Babu Unnam, and Vikram Pudi. 2022b. Multilinguals at SemEval-2022 Task 11: Complex NER in Semantically Ambiguous Settings for Low Resource Languages. In *The 16th International Workshop on Semantic Evaluation*.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Lucy Park, Alice Oh, Jungwoo Ha, and Kyunghyun Cho. 2021. Klue: Korean language understanding evaluation.

Aapo Pietiläinen and Shaoxiong Ji. 2022. AaltoNLP at SemEval-2022 Task 11: Ensembling Task-adaptive Pretrained Transformers for Multilingual Complex NER. In *The 16th International Workshop on Semantic Evaluation*.

Barbara Plank. 2022. Sliced at SemEval-2022 Task 11: Bigger, Better? Massively Multilingual Language Models for Multilingual Complex NER. In *The 16th International Workshop on Semantic Evaluation*.

KEYU PU, Hongyi LIU, Yixiao YANG, Jiangzhou JI, Wenyi LV, and Yaohan He. 2022. CMB AI lab at SemEval-2022 Task 11: A Two-Stage Approach for Complex Named Entity Recognition via Span Boundary Detection and Span Classification. In *The 16th International Workshop on Semantic Evaluation*.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Hossein Rouhizadeh and Douglas Teodoro. 2022. DS4DH at SemEval-2022 Task 11: Multilingual Named Entity Recognition Using an Ensemble of Transformer-based Language Models. In *The 16th International Workshop on Semantic Evaluation*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Elisa Terumi Rubel Schneider, Renzo Mauricio Rivera Zavala, Paloma Martinez Fernandez, Claudia Moro, and EMERSON CABRERA PARAISO. 2022. UC3M-PUCPR at SemEval-2022 Task 11: An Ensemble Method of Transformer-based Models for Complex Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Stefan Schweter. 2020. Berturk - bert models for turkish.

Sumit Singh, Pawankumar Jawale, and Uma Shanker Tiwary. 2022. silpa_nlp at SemEval-2022 Tasks 11: Transformer based NER models for Hindi and Bangla languages. In *The 16th International Workshop on Semantic Evaluation*.

Hyunju Song and Steven Bethard. 2022. UA-KO at SemEval-2022 Task 11: Data Augmentation and Ensembles for Korean Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 175–180, New Orleans, Louisiana. Association for Computational Linguistics.

Charles Sutton, Andrew McCallum, et al. 2012. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373.

Nazia Tasnim, Md. Istiak Hossain Shihab, Asif Shahriyar Sushmit, Steven Bethard, and Farig Sadeque. 2022. TEAM-atreides at SemEval-2022 Task 11: On leveraging data augmentation and ensemble to recognize complex Named Entities in Bangla. In *The 16th International Workshop on Semantic Evaluation*.

Ehsan Tavan and Maryam Najafi. 2022. Marsan at SemEval-2022 Task 11: Multilingual complex named entity recognition using T5 and transformer encoder. In *The 16th International Workshop on Semantic Evaluation*.

Rob van der Goot. 2022. Machamp at SemEval-2022 Tasks 2, 3, 4, 6, 10, 11, and 12: Multi-task Multilingual Learning for a Pre-selected Set of Semantic Datasets. In *The 16th International Workshop on Semantic Evaluation*.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multitask learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

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

Xinyu Wang, Yongliang Shen, Jiong Cai, Tao Wang, Xiaobin Wang, Pengjun Xie, Fei Huang, Weiming Lu, Yueting Zhuang, Kewei Tu, Wei Lu, and Yong Jiang. 2022. DAMO-NLP at SemEval-2022 Task 11: A Knowledge-based System for Multilingual Named Entity Recognition. In *The 16th International Workshop on Semantic Evaluation*.

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

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

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

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2021. Big bird: Transformers for longer sequences.

Appendix

In this section, we provide the domain specific performance of the teams on each track. For each team, we report precision, recall, and F1 for the three domains, i.e., LOWNER, ORCAS, and MSQ. We also highlight the baseline system's performance breakdown for each track. Each track's result is presented in its individual table as listed here:

- Table 6 Bangla (BN)
- Table 7 German (DE)
- Table 8 English (EN)
- Table 9 Spanish (ES)
- Table 10 Farsi (FA)
- Table 11 Hindi (HI)
- Table 12 Korean (KO)
- Table 13 Dutch (NL)
- Table 14 Russian (RU)
- Table 15 Turkish (TR)
- Table 16 Chinese (ZH)
- Table 17 Code-Mixed (MIX)
- Table 18 Multi-lingual (MULTI)

A Detailed Results

A.1 Bangla (BN)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	USTC-NELSLIP	87.32	85.82	86.56	84.2	81.78	82.37	77.21	73.44	75.12	85.84	83.43	84.24
2	DAMO-NLP	86.42	86.21	86.31	83.24	81.25	82.13	73.45	70.9	72.0	84.63	82.53	83.51
3	NetEase.AI	85.83	84.53	85.16	70.32	64.64	67.17	63.36	61.69	62.14	73.78	68.39	70.88
4	RACAI	83.09	83.48	83.28	63.62	60.98	61.6	63.36	58.51	60.69	68.08	65.33	66.28
5	Infrrd.ai	81.52	81.82	81.66	61.59	58.48	59.5	56.06	57.36	56.45	65.68	62.99	63.99
6	YNUNLP	81.46	81.01	81.21	60.64	58.68	59.12	58.58	56.63	57.45	65.11	63.14	63.8
7	Sliced	82.88	83.36	83.1	59.71	57.88	58.06	57.1	57.77	57.12	64.24	62.8	63.05
8	Team Atreides	84.23	82.8	83.48	56.8	52.85	54.23	55.12	58.34	55.44	62.09	58.25	59.75
9	brotherhood	81.56	80.71	81.12	55.18	52.0	53.3	50.91	54.78	51.86	60.33	57.24	58.63
10	MaChAmp	78.44	79.84	79.13	52.18	51.52	51.02	54.05	52.96	52.87	57.25	56.61	56.46
11	MarSan	79.04	79.04	78.98	51.83	48.05	48.83	42.42	50.57	43.92	56.48	53.77	54.22
12	EURECOM	75.36	73.45	74.37	49.33	47.05	48.12	45.52	50.4	45.28	53.78	51.51	52.57
13	AaltoNLP	79.09	78.46	78.74	49.19	43.34	45.78	48.42	47.42	45.84	55.09	49.27	51.79
14	silpa_nlp	76.37	75.97	76.16	47.42	44.68	45.61	44.86	48.77	45.52	53.0	50.34	51.39
15	CSECU-DSG	74.96	74.96	74.95	46.84	43.8	44.85	45.6	48.63	46.14	52.21	49.42	50.55
16	BaselineExtendingPokemons	72.49	75.55	73.96	38.86	40.68	39.2	40.13	44.07	40.97	44.48	46.3	45.07
17	L3i	73.5	72.57	73.01	40.52	38.43	39.08	39.87	42.34	39.82	46.08	43.94	44.81
18	Enigma	73.2	73.34	72.96	41.16	36.48	37.1	39.47	40.82	36.11	46.64	42.03	42.68
19	Baseline	69.27	69.88	69.54	34.12	34.67	34.16	34.03	37.57	34.56	39.29	39.81	39.41

Table 6: Detailed results for Bangla track.

A.2 German (DE)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	94.87	94.92	94.89	84.74	84.4	84.4	84.94	87.8	86.18	90.85	90.5	90.65
2	USTC-NELSLIP	95.8	95.03	95.41	80.5	78.8	79.33	87.06	86.65	86.83	89.88	88.35	89.05
3	RACAI	91.76	91.15	91.44	62.79	61.99	61.89	70.38	73.61	71.61	80.01	78.97	79.39
4	Sliced	90.94	91.04	90.99	61.53	62.81	61.53	71.32	72.85	71.9	78.84	79.18	78.9
5	MaChAmp	89.51	89.87	89.69	61.85	63.71	62.16	69.0	73.04	70.63	78.13	78.83	78.38
6	YNUNLP	90.22	89.6	89.9	59.59	60.56	59.23	70.62	70.7	70.23	77.51	77.42	77.32
7	L3i	90.71	90.72	90.71	60.0	56.23	57.46	64.26	67.57	65.32	78.58	76.18	77.23
8	ML-HUB	88.39	87.7	88.03	59.73	58.93	59.06	60.55	69.53	63.48	76.63	75.8	76.14
9	brotherhood	90.05	89.66	89.85	56.83	55.8	55.78	64.29	67.98	65.53	76.57	75.52	75.94
10	Infrrd.ai	90.65	85.64	88.06	63.26	54.19	57.6	70.52	65.87	67.84	80.05	72.47	75.9
11	EURECOM	88.89	88.68	88.77	56.16	54.01	53.87	62.18	64.02	62.58	75.61	73.84	74.43
12	MarSan	88.4	89.05	88.7	52.14	52.92	51.53	57.75	61.82	58.54	73.1	73.6	73.12
13	CSECU-DSG	86.43	84.81	85.6	56.79	50.75	53.01	61.93	60.99	61.15	74.93	70.47	72.49
14	AaltoNLP	86.49	87.0	86.73	52.31	46.07	48.37	58.33	60.85	59.04	73.16	69.92	71.37
15	PA Ph&Tech	86.08	74.14	79.5	53.42	45.67	48.58	56.6	57.15	55.79	72.65	62.35	66.75
16	BaselineExtendingPokemons	83.81	85.25	84.52	40.75	44.29	42.04	47.58	56.25	50.58	65.44	67.99	66.59
17	Baseline	80.64	81.16	80.83	39.86	41.06	39.96	46.89	55.54	49.6	63.45	64.41	63.74

Table 7: Detailed results for German track.

A.3 English (EN)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	96.68	96.87	96.78	84.27	83.51	83.72	81.76	85.69	83.5	91.54	90.95	91.22
2	USTC-NELSLIP	92.83	91.38	92.09	76.78	75.05	75.59	80.1	83.66	81.74	86.41	84.67	85.47
3	PAI	90.87	91.05	90.96	62.2	60.08	60.58	64.79	68.2	65.91	79.09	77.88	78.37
4	ML-HUB	90.27	87.61	88.9	68.47	56.88	61.6	73.17	67.47	69.46	82.24	74.68	78.14
5	RACAI	88.67	88.08	88.37	59.37	57.87	57.84	67.77	70.11	68.75	76.54	75.35	75.78
6	Infrrd.ai	88.29	87.05	87.65	58.0	55.11	56.15	64.7	68.89	65.83	75.97	73.6	74.71
7	EURECOM	88.82	89.06	88.94	54.1	55.31	54.25	62.55	65.93	63.41	74.43	74.9	74.57
8	Sliced	87.47	87.99	87.73	56.99	57.19	56.17	67.39	69.0	68.06	74.53	74.93	74.54
9	MaChAmp	86.21	87.25	86.72	57.3	58.18	57.11	64.97	69.55	66.75	74.16	74.97	74.48
10	Raccoons	87.66	89.39	88.5	53.63	55.09	54.02	63.57	68.39	65.42	73.43	75.05	74.18
11	YNUNLP	86.75	86.92	86.83	53.96	55.4	53.98	64.99	68.33	65.78	72.99	73.64	73.17
12	LMN	87.05	88.71	87.87	50.96	52.17	51.2	58.43	63.84	60.45	71.78	73.33	72.5
13	brotherhood	87.16	86.41	86.78	52.76	51.48	51.67	61.74	65.7	62.73	73.18	71.71	72.35
14	L3i	87.21	87.34	87.26	54.6	47.87	49.71	61.33	64.08	62.57	73.82	70.8	71.96
15	Multilinguals	86.47	87.43	86.94	53.03	48.86	50.16	59.4	59.55	59.11	72.71	71.09	71.74
16	KDDIE	86.65	87.7	87.17	50.36	51.15	50.4	58.29	63.63	60.57	71.34	72.26	71.73
17	MarSan	85.75	86.21	85.96	50.83	52.24	51.16	58.91	64.9	60.64	71.11	71.91	71.45
18	Cardiff NLP	85.93	87.6	86.75	47.63	51.41	49.24	56.23	64.49	58.55	69.72	72.28	70.94
19	Lone Wolf	85.08	85.96	85.51	47.36	48.76	47.75	56.33	62.44	58.47	69.35	70.31	69.77
20	MIDAS	84.68	81.79	83.19	54.75	46.84	49.73	60.63	57.45	58.34	72.95	66.95	69.62
21	UC3M-PUCPR	86.6	87.12	86.84	46.23	46.28	44.25	54.95	57.3	54.07	69.95	69.73	69.24
22	CSECU-DSG	84.76	86.08	85.41	47.0	47.79	47.22	50.45	60.14	54.01	68.72	69.81	69.24
23	Sartipi-Sedighin	82.95	84.69	83.78	44.4	47.16	45.6	46.42	58.19	49.72	66.34	68.79	67.51
24	Enigma	82.55	83.19	82.86	46.14	46.04	45.45	57.07	61.73	58.17	66.97	67.74	67.19
25	DANGNT-SGU	83.6	84.7	84.1	43.14	44.68	43.28	51.75	58.74	53.38	66.51	67.7	66.89
26	AaltoNLP	83.27	84.19	83.73	48.89	33.87	39.24	53.95	54.21	53.51	71.57	63.0	66.85
27	SPDB Innovation Lab	81.35	81.74	81.54	42.16	43.57	42.63	49.45	57.98	52.18	64.52	65.77	65.11
28	silpa_nlp	81.48	80.54	80.99	39.58	38.98	38.65	48.81	54.1	49.95	64.13	63.06	63.42
29	BaselineExtendingPokemons	80.03	82.27	81.11	38.13	42.3	39.96	43.97	57.67	48.84	61.36	65.35	63.24
30	Baseline	78.25	78.0	78.11	38.89	37.47	37.61	46.21	52.2	48.26	62.07	60.97	61.36
31	AutoNER	72.29	74.77	73.35	30.6	33.53	31.02	45.77	49.65	47.21	54.73	57.68	55.72

Table 8: Detailed results for English track.

A.4 Spanish (ES)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	96.23	96.15	96.19	82.45	80.91	81.33	82.25	84.51	83.1	90.58	89.41	89.94
2	USTC-NELSLIP	90.15	88.08	89.1	80.99	79.17	79.68	84.49	85.95	85.1	86.64	84.39	85.44
3	RACAI	85.29	85.31	85.29	63.37	62.39	61.74	71.98	72.92	72.35	76.21	75.43	75.62
4	Infrrd.ai	85.21	85.55	85.37	62.06	61.71	61.32	66.18	70.93	67.82	75.59	75.11	75.26
5	MaChAmp	84.71	85.34	85.01	61.53	62.99	61.49	67.99	72.11	69.5	74.94	75.66	75.2
6	Sliced	85.68	85.92	85.79	60.74	61.6	60.39	69.21	71.57	70.18	75.15	75.32	75.11
7	YNUNLP	84.18	85.45	84.8	57.41	58.71	56.95	68.33	68.64	68.17	72.93	73.8	73.17
8	brotherhood	85.66	84.73	85.19	51.7	51.52	51.08	59.55	62.29	60.31	71.23	70.35	70.69
9	L3i	83.73	84.32	84.01	48.98	50.03	48.94	52.25	58.52	54.73	68.71	69.34	68.93
10	PA Ph&Tech	82.95	81.48	82.21	51.11	52.8	51.49	51.07	64.25	55.15	68.89	69.23	68.93
11	MarSan	83.12	82.84	82.96	49.26	50.7	48.52	56.64	60.4	57.46	68.65	68.71	68.3
12	SPDB Innovation Lab	83.57	81.69	82.55	49.62	48.7	46.57	60.29	57.49	57.59	68.96	67.24	67.31
13	CSECU-DSG	82.87	79.64	81.2	47.04	41.64	43.17	55.04	53.7	53.72	68.94	63.13	65.62
14	EURECOM	80.25	80.44	80.31	40.24	41.26	40.25	40.59	48.14	43.16	62.49	63.26	62.77
15	Multilinguals	81.13	80.52	80.81	36.73	34.24	34.69	42.77	45.66	43.57	62.27	60.46	61.2
16	Sartipi-Sedighin	77.29	79.74	78.36	37.11	39.41	37.62	42.44	47.33	43.67	59.82	62.03	60.7
17	BaselineExtendingPokemons	77.3	80.34	78.76	34.47	38.82	36.01	40.95	50.19	44.0	58.32	62.22	60.08
18	Baseline	75.66	77.0	76.24	33.32	36.11	33.58	41.34	45.99	43.08	57.07	59.08	57.84
19	UC3M-PUCPR	73.93	72.16	72.89	37.33	36.14	35.42	40.39	41.63	40.88	58.38	56.22	56.79

Table 9: Detailed results for Spanish track.

A.5 Farsi (FA)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	95.96	97.01	96.48	84.99	84.92	84.84	86.79	88.15	87.36	89.81	89.66	89.7
2	USTC-NELSLIP	86.13	84.49	85.29	88.2	86.57	87.2	92.45	91.11	91.75	88.16	86.1	87.05
3	RACAI	80.5	82.15	81.31	63.18	61.96	62.02	70.05	72.19	70.87	70.77	70.45	70.42
4	Sliced	79.17	82.11	80.61	59.39	61.16	59.93	64.35	69.55	66.23	67.83	69.77	68.66
5	YNUNLP	79.54	80.54	80.02	58.25	58.55	57.78	67.51	68.33	67.65	67.29	67.57	67.19
6	brotherhood	81.16	81.69	81.41	56.13	54.41	54.7	61.28	64.6	62.35	66.46	65.53	65.74
7	C-3PO	78.94	81.67	80.27	55.55	55.46	55.08	59.7	65.57	61.84	65.14	66.28	65.51
8	L3i	79.18	80.65	79.89	54.78	55.18	54.63	56.26	63.86	59.08	64.91	65.59	65.11
9	MarSan	76.49	80.84	78.59	51.54	50.99	50.61	55.71	57.7	56.29	61.8	63.06	62.14
10	MaChAmp	75.36	78.68	76.98	48.89	51.69	49.72	50.36	58.05	53.13	59.4	62.43	60.71
11	AaltoNLP	76.46	79.88	78.1	48.19	46.28	46.37	48.18	56.85	51.39	59.19	59.58	58.93
12	Sartipi-Sedighin	75.79	79.59	77.63	44.49	44.77	44.41	45.79	53.09	47.63	57.08	58.64	57.73
13	EURECOM	75.0	77.28	76.06	43.2	42.14	42.28	43.79	48.69	45.16	56.07	56.08	55.91
14	CSECU-DSG	75.96	79.15	77.5	41.98	41.09	41.06	43.58	49.03	45.55	55.8	56.17	55.81
15	Baseline	69.25	74.5	71.67	41.12	39.84	39.03	45.79	48.2	46.59	52.7	53.46	52.24
16	BaselineExtendingPokemons	70.89	77.3	73.91	36.59	39.87	37.62	34.33	46.97	39.02	49.15	54.33	51.26

Table 10: Detailed results for Farsi track.

A.6 Hindi (HI)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	84.85	83.54	84.18	86.82	84.85	85.75	88.54	89.94	89.2	87.27	85.28	86.23
2	USTC-NELSLIP	86.49	83.93	85.18	84.66	82.87	83.16	91.89	90.14	90.94	86.0	83.92	84.64
3	RACAI	82.04	81.99	82.01	64.17	62.46	62.65	75.62	75.96	75.55	69.05	67.77	68.08
4	Sliced	81.48	81.81	81.64	62.82	61.48	61.32	73.49	75.32	73.75	67.93	66.98	67.0
5	NetEase.AI	85.69	82.75	84.15	63.24	57.65	59.88	72.61	73.95	72.38	69.67	64.27	66.63
6	Infrrd.ai	79.87	80.01	79.93	61.41	60.08	59.99	71.21	75.78	72.9	66.58	65.6	65.72
7	brotherhood	82.17	81.01	81.58	59.76	56.88	57.68	68.91	74.4	70.59	65.72	63.35	64.23
8	YNUNLP	79.67	79.89	79.77	58.21	57.96	57.35	69.94	72.85	70.7	63.8	63.69	63.39
9	OPDAI	74.82	75.9	75.28	57.86	59.07	57.86	67.39	70.83	68.76	63.03	63.58	62.94
10	MaChAmp	76.31	77.7	76.99	56.15	56.27	55.5	67.53	72.36	69.47	61.9	62.21	61.73
11	CSECU-DSG	75.97	71.45	73.56	55.49	48.73	51.54	66.57	65.47	65.16	61.46	54.77	57.68
12	MarSan	75.09	75.61	75.27	49.77	50.46	49.34	62.45	66.61	63.72	56.39	57.01	56.31
13	EURECOM	68.92	69.69	69.27	48.59	46.71	47.17	53.28	59.51	54.73	53.84	52.36	52.78
14	silpa_nlp	73.9	73.75	73.81	45.53	43.87	44.32	51.24	58.5	51.16	52.44	51.22	51.49
15	BaselineExtendingPokemons	70.78	72.33	71.49	42.34	43.21	42.33	55.07	61.31	55.57	49.68	50.68	49.9
16	L3i	72.38	71.34	71.8	44.0	42.07	42.26	51.0	58.13	52.89	51.01	49.24	49.73
17	Enigma	71.14	71.84	71.03	44.43	39.86	40.47	56.21	58.85	55.9	51.61	48.28	48.62
18	Baseline	65.67	66.44	65.96	41.38	42.59	41.55	54.03	56.67	53.26	48.08	48.98	48.22

Table 11: Detailed results for Hindi track.

A.7 Korean (KO)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	96.58	97.1	96.83	81.1	81.41	81.06	79.44	84.96	81.96	88.55	88.7	88.59
2	USTC-NELSLIP	90.64	90.11	90.37	83.23	81.15	81.82	88.45	87.97	88.19	87.39	85.56	86.36
3	RACAI	85.26	86.81	86.02	58.82	58.58	57.79	69.52	69.63	69.38	72.06	71.93	71.74
4	CMB AI Lab	88.93	88.23	88.57	60.73	47.02	52.7	64.9	58.35	61.09	75.92	66.33	70.7
5	Sliced	84.81	86.93	85.85	55.92	58.41	56.44	65.28	68.6	66.81	69.82	71.94	70.66
6	YNUNLP	84.74	85.42	85.05	57.39	57.11	56.36	66.48	68.39	67.34	70.69	70.51	70.33
7	C-3PO	86.24	87.42	86.8	51.02	49.85	49.69	56.08	57.72	56.27	68.15	67.4	67.49
8	UA-KO	85.91	87.78	86.83	50.59	49.63	49.67	55.92	59.24	56.92	67.72	67.52	67.49
9	brotherhood	85.83	86.67	86.24	50.8	50.69	50.23	57.33	61.72	58.98	67.61	67.5	67.41
10	Infrrd.ai	84.15	86.13	85.13	50.75	52.07	50.99	58.9	63.65	60.55	66.69	68.17	67.29
11	MaChAmp	81.48	83.99	82.71	49.31	51.21	49.6	53.61	62.55	57.03	64.68	66.55	65.45
12	EURECOM	86.4	86.63	86.5	46.68	46.14	45.87	50.13	54.66	51.57	65.25	65.14	64.96
13	L3i	83.92	84.93	84.38	42.92	45.9	43.97	48.18	55.21	50.82	61.57	64.09	62.68
14	MarSan	81.58	84.79	83.14	43.31	45.49	43.75	47.76	54.49	49.99	61.13	63.92	62.26
15	CSECU-DSG	82.99	85.12	84.04	43.2	42.04	42.11	48.96	50.41	48.72	62.27	62.14	62.05
16	AaltoNLP	82.06	83.23	82.61	44.96	43.38	42.86	48.62	53.42	50.35	62.72	61.92	61.82
17	BaselineExtendingPokemons	77.42	82.93	80.06	39.65	41.7	40.06	41.75	51.34	44.83	57.46	60.84	58.95
18	Baseline	76.2	76.86	76.46	36.64	38.75	37.0	37.71	46.29	40.03	54.77	56.38	55.25

Table 12: Detailed results for Korean track.

A.8 Dutch (NL)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	97.92	98.0	97.96	81.16	80.39	80.46	83.17	84.26	83.65	90.95	90.14	90.5
2	USTC-NELSLIP	92.14	90.74	91.43	82.63	81.09	81.64	86.3	87.87	86.95	88.56	86.86	87.67
3	RACAI	89.68	89.73	89.7	63.9	63.4	62.83	70.34	72.81	71.2	78.82	78.3	78.41
4	Sliced	89.08	89.42	89.25	61.81	63.1	61.87	69.15	72.56	70.41	77.55	77.95	77.66
5	MaChAmp	88.03	88.65	88.33	61.35	62.83	61.44	67.87	71.78	69.47	76.72	77.43	76.99
6	Infrrd.ai	91.32	85.54	88.31	64.73	56.72	59.74	70.51	68.1	69.08	80.5	73.04	76.4
7	YNUNLP	88.95	88.21	88.56	59.39	59.79	58.78	67.04	70.12	68.25	76.19	75.82	75.82
8	brotherhood	88.86	88.05	88.44	53.74	52.63	52.35	58.5	63.87	60.22	73.96	72.46	73.04
9	PA Ph&Tech	87.49	86.76	87.11	51.28	55.73	52.76	56.39	66.13	59.98	71.06	73.32	72.05
10	MarSan	86.5	87.67	87.06	52.0	52.51	50.26	56.61	61.27	58.27	71.18	71.98	71.13
11	L3i	86.65	87.73	87.15	50.22	50.15	49.39	54.56	60.34	56.74	70.96	71.32	70.96
12	CSECU-DSG	84.82	81.82	83.24	50.84	42.96	45.53	59.8	55.21	57.11	71.74	65.0	67.94
13	EURECOM	82.05	84.25	83.1	45.27	46.43	44.8	49.11	56.91	52.23	66.11	67.75	66.7
14	BaselineExtendingPokemons	81.63	84.91	83.22	37.41	39.6	38.11	40.77	52.49	44.79	61.77	65.07	63.25
15	Baseline	80.66	81.63	81.12	37.16	36.88	36.44	43.4	49.9	45.95	62.04	62.25	62.01
16	Sartipi-Sedighin	80.21	81.07	80.6	29.57	30.59	29.78	34.69	45.4	36.96	57.86	59.07	58.37

Table 13: Detailed results for Dutch track.

A.9 Russian (RU)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	96.37	96.84	96.6	85.89	84.55	85.0	86.89	87.42	87.03	91.93	91.14	91.5
2	USTC-NELSLIP	85.22	83.22	84.2	83.16	81.71	82.23	85.37	86.71	85.91	84.85	82.89	83.82
3	RACAI	82.19	82.07	82.12	66.92	63.51	63.93	76.5	72.78	74.2	75.86	73.83	74.6
4	Sliced	80.65	82.48	81.55	63.54	63.66	62.97	72.14	71.22	71.27	73.59	74.11	73.73
5	YNUNLP	81.41	80.01	80.67	64.38	62.83	62.64	71.95	69.98	70.17	74.09	72.28	72.99
6	MaChAmp	78.65	81.28	79.94	62.64	63.11	62.04	68.82	69.12	68.38	72.0	73.06	72.37
7	brotherhood	80.59	80.92	80.75	57.67	56.22	56.26	64.75	65.26	63.52	71.0	69.84	70.27
8	NetEase.AI	81.07	77.42	79.19	60.42	54.74	56.89	64.69	65.51	63.45	72.61	67.44	69.79
9	EURECOM	80.26	80.11	80.17	54.56	51.04	51.82	63.33	63.7	61.24	69.74	67.19	68.21
10	MarSan	77.99	79.42	78.68	52.33	55.01	53.1	57.5	63.79	58.09	66.83	68.44	67.49
11	L3i	78.64	78.91	78.77	52.76	49.85	50.69	56.76	60.27	57.01	67.89	65.82	66.72
12	CSECU-DSG	75.86	78.26	77.04	44.29	45.97	44.57	53.87	58.07	54.23	62.6	63.9	63.08
13	BaselineExtendingPokemons	71.28	76.78	73.92	41.76	45.55	43.03	44.28	53.55	47.22	58.04	62.4	60.0
14	Baseline	70.58	74.55	72.47	42.57	44.93	43.45	46.36	52.95	48.01	58.42	60.96	59.59
15	AutoNER	62.52	65.16	63.66	37.35	40.19	37.88	51.05	55.08	52.22	51.79	54.33	52.7

Table 14: Detailed results for Russian track.

A.10 Turkish (TR)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	DAMO-NLP	96.45	96.42	96.43	86.86	85.39	85.85	89.22	88.4	88.76	89.79	87.82	88.69
2	USTC-NELSLIP	90.32	89.8	90.05	84.22	82.7	83.17	87.64	88.1	87.83	86.62	84.7	85.52
3	SU-NLP	83.53	85.11	84.29	76.0	61.07	67.57	75.13	64.87	68.73	78.86	66.43	72.02
4	RACAI	87.45	88.83	88.13	65.72	64.73	64.04	74.65	73.1	73.59	71.81	70.25	70.42
5	Sliced	86.72	88.51	87.6	62.81	63.47	62.31	70.77	71.38	70.92	69.17	69.22	68.77
6	MaChAmp	84.63	86.93	85.75	61.48	62.81	61.37	65.26	68.63	66.63	67.55	68.3	67.58
7	YNUNLP	86.59	86.99	86.78	61.14	61.04	59.66	73.32	69.45	70.8	68.17	67.05	66.81
8	ML-HUB	84.31	86.92	85.55	61.11	59.47	59.62	56.0	68.23	59.21	66.51	65.98	65.79
9	L3i	85.6	86.99	86.28	57.3	57.15	56.65	63.23	65.47	63.89	64.85	64.26	64.28
10	MarSan	84.73	86.67	85.68	52.39	56.12	53.49	56.27	60.51	57.01	60.22	62.7	61.09
11	brotherhood	85.27	86.86	86.01	50.88	52.78	51.18	57.21	62.29	58.72	59.42	60.68	59.71
12	EURECOM	82.97	85.45	84.18	47.75	50.08	48.41	48.96	55.71	50.97	55.93	57.86	56.57
13	CSECU-DSG	81.86	79.8	80.76	56.5	40.42	46.26	58.08	48.79	52.18	64.57	49.25	55.3
14	Sartipi-Sedighin	79.69	85.74	82.52	42.23	47.34	43.71	45.77	53.34	48.6	50.77	55.81	52.69
15	Baseline	75.87	79.21	77.49	36.2	39.91	37.0	39.19	44.33	40.62	45.31	48.28	46.25
16	BaselineExtendingPokemons	77.39	82.64	79.86	33.63	38.92	35.4	35.41	43.94	38.09	42.74	48.32	44.97

Table 15: Detailed results for Turkish track.

A.11 Chinese (ZH)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	USTC-NELSLIP	92.76	89.42	91.01	77.96	74.73	75.64	85.94	84.84	85.37	83.94	80.07	81.69
2	CASIA	88.52	81.48	84.61	81.04	72.75	75.88	86.56	80.62	83.37	84.77	76.13	79.7
3	OPDAI	85.14	85.44	85.27	75.46	75.81	74.69	80.9	83.93	82.24	80.54	79.47	79.54
4	DAMO-NLP	89.75	87.87	88.77	74.41	73.09	72.24	80.57	80.97	80.1	80.64	77.45	78.06
5	NetEase.AI	89.73	85.48	87.47	75.69	70.89	72.01	77.71	78.92	77.84	81.52	75.63	77.77
6	CMB AI Lab	90.18	86.67	88.37	74.36	64.08	68.54	76.66	75.66	75.53	81.28	72.29	76.36
7	NCUEE-NLP	85.28	81.26	83.1	70.9	68.38	68.5	75.33	76.8	75.27	77.01	72.99	74.18
8	QTrade AI	88.76	85.43	86.98	69.19	66.37	66.3	76.88	77.45	76.88	76.91	72.82	74.0
9	CSECU-DSG	84.85	83.33	84.04	59.16	59.61	58.05	65.15	71.02	66.86	68.55	67.61	67.22
10	Multilinguals	84.48	83.01	83.72	59.48	59.2	57.91	61.93	71.1	64.78	68.39	67.22	66.95
11	L3i	84.5	82.14	83.25	59.63	59.46	58.07	63.71	70.65	65.08	68.62	67.07	66.91
12	Sliced	85.54	84.25	84.86	58.6	56.1	55.02	64.92	67.65	65.09	67.99	65.07	65.21
13	Infrrd.ai	82.98	77.89	80.19	59.28	56.65	56.03	62.74	70.05	64.26	67.83	64.16	64.68
14	MaChAmp	83.38	81.82	82.55	57.16	55.83	54.48	61.16	65.41	61.14	66.45	63.88	63.81
15	EURECOM	83.53	81.39	82.25	57.51	54.4	53.25	63.81	68.65	64.31	66.89	63.43	63.4
16	RACAI	86.53	83.53	84.91	58.07	52.36	51.29	65.45	65.12	63.34	68.17	62.05	62.7
17	YNUNLP	83.01	82.63	82.79	53.44	51.46	50.3	64.27	67.0	64.48	63.99	61.41	61.38
18	brotherhood	85.04	81.55	83.11	53.03	49.84	49.69	58.74	62.61	59.25	64.08	60.05	60.86
19	MarSan	81.95	80.59	81.19	48.76	46.19	44.49	57.56	61.96	58.11	60.15	57.1	56.64
20	SPDB Innovation Lab	80.88	79.4	80.06	45.56	46.47	43.97	53.92	59.42	54.94	57.25	57.09	55.74
21	BaselineExtendingPokemons	74.57	78.13	76.23	43.8	44.51	41.39	52.72	56.93	51.41	54.44	55.06	52.8
22	Baseline	73.7	73.18	73.29	43.39	42.43	40.54	45.66	53.57	46.53	53.51	52.32	51.3

Table 16: Detailed results for Chinese track.

A.12 Code-Mixed (MIX)

Rank	Team	LOWNER			ORCAS-NER			MSQ-NER			Average		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	USTC-NELSLIP	95.5	94.99	95.21	89.1	88.64	88.74	93.32	93.39	93.22	93.21	92.61	92.9
2	DAMO-NLP	93.88	93.36	93.57	88.37	87.25	87.68	91.39	90.24	90.56	92.35	91.24	91.79
3	CMB AI Lab	93.68	91.22	92.35	79.89	68.21	73.2	85.02	76.56	80.0	88.4	81.27	84.62
4	QTrade AI	93.21	90.36	91.65	78.06	71.21	73.93	82.86	77.44	79.42	87.12	82.01	84.35
5	SeqL	91.55	90.92	91.16	67.04	65.81	65.89	76.32	75.28	74.93	81.1	79.72	80.29
6	IIE_KDSEC	87.45	88.35	87.8	67.28	67.25	66.88	75.48	76.15	75.29	79.52	79.74	79.59
7	RACAI	89.43	89.6	89.42	65.78	65.48	65.29	74.53	74.51	73.87	79.58	79.23	79.37
8	UM6P-CS	87.7	88.16	87.82	67.08	66.62	66.5	75.06	74.57	74.11	79.38	79.08	79.21
9	EURECOM	86.63	87.25	86.82	63.58	62.71	62.71	73.99	74.11	73.28	77.84	77.37	77.6
10	OPDAI	87.26	85.98	86.45	61.87	61.12	61.0	72.78	71.82	71.49	77.79	77.23	77.46
11	YNUNLP	85.85	87.03	86.33	62.48	62.85	62.12	72.23	72.62	71.77	76.64	76.98	76.78
12	UC3M-PUCPR	87.8	86.74	87.15	60.94	58.86	59.48	71.92	70.74	70.57	77.15	75.64	76.36
13	brotherhood	87.84	87.51	87.57	60.83	60.13	59.87	71.28	70.51	70.0	76.62	75.45	75.91
14	MaChAmp	85.37	86.66	85.82	58.05	60.8	58.66	69.41	69.87	68.55	73.85	75.4	74.52
15	Sliced	86.96	88.05	87.41	54.91	55.8	54.4	68.51	67.59	66.63	72.67	73.23	72.74
16	CMNEROne	83.05	83.24	82.97	51.68	52.34	51.36	63.81	64.02	63.05	70.41	70.62	70.44
17	L3i	73.32	71.49	71.9	56.08	56.22	55.53	66.74	66.4	65.79	68.93	68.73	68.7
18	Cardiff NLP	72.56	73.5	72.66	57.17	58.84	57.45	67.71	69.29	67.96	67.4	69.03	68.07
19	BaselineExtendingPokemons	72.97	72.47	72.31	56.13	55.7	55.45	67.01	66.59	66.01	67.92	68.27	67.99
20	SPDB Innovation Lab	76.07	76.58	76.08	51.73	52.82	51.78	64.59	65.22	64.05	66.96	67.8	67.32
21	MarSan	73.27	70.17	70.76	54.73	56.43	54.75	65.63	66.03	64.59	67.36	67.41	67.03
22	CSECU-DSG	68.11	68.02	67.62	53.46	53.4	52.53	63.54	63.26	62.32	64.23	64.36	64.03
23	Baseline	74.03	74.23	73.72	38.85	37.13	37.15	51.45	50.92	49.86	59.1	57.66	58.14

Table 17: Detailed results for Code-Mixed track.

A.13 Multilingual (MULTI)

Team		**LOWNER**			**ORCAS-NER**			**MSQ-NER**			**Avg.**		
		P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
1. DAMO-NLP	**BN**	86.4	85.64	86.01	79.18	76.72	77.77	71.85	69.23	70.28	79.14	77.2	78.02
	DE	94.29	94.51	94.4	81.87	82.11	81.62	84.91	86.01	85.31	87.02	87.54	87.11
	EN	96.52	96.66	96.59	82.26	81.45	81.57	82.4	83.87	83.04	87.06	87.33	87.06
	ES	95.63	95.82	95.72	79.61	79.07	78.85	83.16	86.22	84.48	86.13	87.04	86.35
	FA	95.94	96.97	96.45	82.98	82.85	82.69	86.15	86.4	86.11	88.36	88.74	88.42
	HI	84.64	83.8	84.2	82.42	80.96	81.46	87.74	88.59	88.1	84.93	84.45	84.58
	KO	96.59	97.07	96.83	79.29	79.7	79.18	80.88	84.72	82.6	85.59	87.17	86.2
	NL	97.65	97.78	97.72	79.1	79.03	78.56	84.46	84.16	84.27	87.07	86.99	86.85
	RU	96.0	96.6	96.3	82.19	81.52	81.42	86.19	85.92	85.9	88.12	88.01	87.87
	TR	97.66	97.87	97.77	82.66	82.18	82.0	87.69	84.86	86.17	89.34	88.31	88.65
	ZH	88.37	87.9	88.12	67.74	66.83	65.59	78.02	79.74	78.09	78.05	78.16	77.27
	Avg.	93.61	93.69	93.65	79.94	79.31	79.16	83.04	83.61	83.12	85.53	85.54	85.31
2. USTC-NELSLIP	**BN**	87.48	85.7	86.57	83.99	81.65	82.29	76.6	73.16	74.71	82.69	80.17	81.19
	DE	95.8	95.1	95.45	80.0	79.0	79.25	87.47	85.36	86.33	87.76	86.49	87.01
	EN	93.02	91.63	92.32	77.19	76.11	76.36	80.01	84.03	81.69	83.41	83.92	83.46
	ES	90.05	88.11	89.06	80.65	79.38	79.62	84.84	87.69	86.09	85.18	85.06	84.92
	FA	86.23	84.9	85.54	87.81	86.34	86.9	93.02	91.14	92.04	89.02	87.46	88.16
	HI	86.74	83.83	85.25	84.65	82.8	83.17	91.51	90.64	91.04	87.63	85.76	86.48
	KO	90.62	90.44	90.53	83.34	81.15	81.87	87.36	87.29	87.27	87.11	86.29	86.55
	NL	91.93	90.93	91.43	82.65	81.4	81.81	86.89	86.71	86.58	87.16	86.35	86.61
	RU	84.8	83.42	84.1	82.57	81.15	81.62	86.88	86.67	86.64	84.75	83.75	84.12
	TR	90.32	89.77	90.04	83.52	82.58	82.82	86.43	87.53	86.94	86.76	86.63	86.6
	ZH	92.99	89.98	91.44	77.42	72.83	74.39	85.02	82.84	83.87	85.14	81.88	83.23
	Avg.	90.0	88.53	89.25	82.16	80.4	80.92	86.0	85.73	85.75	86.06	84.89	85.3
3. QTrade AI	**BN**	85.09	85.1	85.08	70.37	68.98	69.22	65.93	63.96	64.69	73.8	72.68	73.0
	DE	93.08	92.36	92.72	73.04	72.64	72.36	77.95	76.61	76.69	81.36	80.54	80.59
	EN	89.84	89.02	89.42	67.71	67.3	67.14	72.03	72.45	71.82	76.53	76.26	76.13
	ES	87.98	86.53	87.24	72.32	71.78	71.51	77.0	80.07	78.03	79.1	79.46	78.93
	FA	81.96	81.97	81.94	71.21	70.69	70.59	75.71	76.16	75.42	76.29	76.27	75.99
	HI	84.18	83.48	83.81	72.33	70.98	71.3	79.48	81.06	80.13	78.66	78.5	78.41
	KO	87.57	87.49	87.52	68.01	67.34	67.08	74.83	75.64	75.0	76.81	76.82	76.53
	NL	90.99	90.1	90.54	72.66	72.37	72.06	76.97	77.82	77.05	80.21	80.1	79.88
	RU	83.38	82.4	82.89	74.83	72.91	73.03	80.89	76.97	78.38	79.7	77.43	78.1
	TR	88.81	88.64	88.72	74.18	73.22	73.09	79.34	77.84	78.41	80.78	79.9	80.07
	ZH	88.06	85.55	86.74	68.26	66.02	65.88	77.85	77.3	77.34	78.06	76.29	76.65
	Avg.	87.36	86.6	86.97	71.36	70.38	70.3	76.18	75.99	75.72	78.3	77.66	77.66
4. SeqL	**BN**	86.08	84.98	85.51	67.19	64.55	65.63	60.09	61.73	60.53	71.12	70.42	70.56
	DE	92.87	91.98	92.42	69.63	68.72	68.75	72.43	76.2	73.71	78.31	78.97	78.29
	EN	90.42	88.95	89.67	64.87	63.63	63.9	69.69	72.79	70.85	74.99	75.12	74.81
	ES	88.39	86.19	87.26	69.13	68.01	68.19	72.47	78.55	74.79	76.66	77.58	76.75
	FA	83.53	81.32	82.38	69.12	67.58	68.15	70.29	73.81	70.87	74.31	74.24	73.8
	HI	84.84	82.7	83.73	69.59	67.18	68.01	76.47	80.6	78.26	76.97	76.82	76.67
	KO	87.59	87.27	87.42	65.93	64.75	64.9	70.51	75.32	72.58	74.68	75.78	74.97
	NL	91.14	89.69	90.41	68.73	67.82	67.92	73.22	76.14	74.11	77.7	77.89	77.48
	RU	84.08	81.8	82.91	72.86	69.85	70.68	78.32	76.7	76.91	78.42	76.12	76.83
	TR	89.37	88.01	88.68	70.95	69.69	69.88	73.83	76.25	74.8	78.05	77.98	77.79
	ZH	88.15	85.89	86.95	63.15	59.56	59.44	71.81	72.57	71.1	74.37	72.67	72.5
	Avg.	87.86	86.25	87.03	68.29	66.49	66.86	71.74	74.61	72.59	75.96	75.78	75.5
5. CMB AI Lab	**BN**	89.09	81.88	85.32	72.13	52.88	60.4	66.07	55.27	59.97	75.76	63.34	68.56
	DE	94.33	89.61	91.9	73.18	58.05	63.61	81.46	69.55	74.44	82.99	72.4	76.65
	EN	91.86	87.1	89.4	68.42	51.66	57.75	77.06	64.82	70.07	79.12	67.86	72.4
	ES	89.97	84.68	87.21	71.83	56.53	62.11	78.19	70.48	73.89	80.0	70.56	74.4
	FA	85.07	80.51	82.66	72.96	56.35	62.92	77.93	65.99	70.8	78.65	67.62	72.13
	HI	87.89	79.94	83.66	73.64	57.26	63.76	82.92	74.45	78.35	81.48	70.55	75.26
	KO	89.46	84.83	87.06	71.92	55.66	61.62	79.96	67.73	72.91	80.45	69.41	73.86
	NL	92.65	88.07	90.29	72.96	57.99	63.43	80.4	68.32	73.26	82.0	71.46	75.66
	RU	84.39	80.89	82.57	73.96	59.41	64.69	84.19	69.33	75.59	80.85	69.88	74.28
	TR	90.65	87.57	89.07	74.6	58.82	64.69	81.79	69.13	74.55	82.35	71.84	76.11
	ZH	91.05	83.63	87.11	70.03	50.03	57.0	76.11	64.96	69.57	79.06	66.21	71.23

	Avg.	89.67	84.43	86.93	72.33	55.88	62.0	78.73	67.28	72.13	80.25	69.19	73.69
6. UM6P-CS	**BN**	81.82	81.7	81.75	62.2	59.46	60.15	61.75	58.43	59.8	68.59	66.53	67.24
	DE	90.69	90.55	90.62	63.99	64.33	63.51	73.59	72.19	72.64	76.09	75.69	75.59
	EN	87.81	87.46	87.63	59.2	58.28	58.05	69.68	68.15	68.81	72.23	71.3	71.5
	ES	86.06	85.57	85.81	63.52	63.03	62.49	73.01	73.74	73.13	74.2	74.11	73.81
	FA	80.46	80.72	80.58	63.04	62.53	62.37	69.88	71.14	70.07	71.13	71.46	71.01
	HI	81.21	80.49	80.83	64.59	61.92	62.33	79.6	76.31	77.83	75.13	72.91	73.66
	KO	84.56	86.02	85.28	59.03	58.9	58.33	66.54	70.03	67.97	70.04	71.65	70.53
	NL	89.13	88.99	89.05	63.86	63.77	63.09	72.86	71.28	71.91	75.28	74.68	74.69
	RU	81.19	81.65	81.41	66.65	64.66	64.71	76.58	72.15	73.96	74.81	72.82	73.36
	TR	87.33	87.73	87.53	65.65	65.2	64.58	76.34	74.47	75.28	76.44	75.8	75.8
	ZH	84.55	83.4	83.97	60.37	57.44	56.97	69.06	71.16	69.74	71.32	70.67	70.22
	Avg.	84.98	84.93	84.95	62.92	61.77	61.51	71.72	70.82	71.01	73.21	72.51	72.49
7. RACAI	**BN**	82.81	82.79	82.79	62.85	60.41	61.11	59.47	60.58	59.86	68.38	67.93	67.92
	DE	91.27	91.07	91.17	63.67	63.11	62.63	70.67	72.1	71.04	75.2	75.43	74.95
	EN	87.83	88.18	88.01	59.72	58.49	58.29	67.49	69.47	68.27	71.68	72.04	71.52
	ES	83.38	83.45	83.4	61.91	61.09	60.66	67.61	73.13	69.95	70.96	72.56	71.34
	FA	79.7	82.46	81.03	63.68	62.93	62.85	68.12	70.83	68.97	70.5	72.07	70.95
	HI	82.33	81.99	82.15	65.02	63.16	63.46	75.29	77.86	76.27	74.21	74.34	73.96
	KO	85.51	87.23	86.34	59.83	59.88	58.95	67.11	70.08	68.36	70.82	72.4	71.22
	NL	89.08	89.5	89.29	63.63	63.22	62.66	69.13	70.72	69.75	73.95	74.48	73.9
	RU	80.36	82.19	81.26	66.4	63.83	63.98	75.57	73.34	74.1	74.11	73.12	73.11
	TR	86.53	88.54	87.52	65.51	64.64	64.21	72.34	73.61	72.8	74.79	75.6	74.84
	ZH	84.72	84.42	84.57	59.53	57.11	56.4	66.33	70.17	67.17	70.19	70.56	69.38
	Avg.	84.87	85.62	85.23	62.89	61.62	61.38	69.01	71.08	69.69	72.25	72.78	72.1
8. Cardiff NLP	**BN**	82.43	83.67	83.02	60.67	60.32	60.1	59.19	59.87	59.05	67.43	67.95	67.39
	DE	91.24	91.47	91.35	62.09	63.47	62.06	70.23	72.8	71.19	74.52	75.91	74.87
	EN	87.7	88.63	88.16	57.69	58.55	57.49	66.69	68.7	67.5	70.69	71.96	71.05
	ES	85.4	86.71	86.04	61.36	62.47	61.2	68.96	74.82	71.23	71.91	74.67	72.82
	FA	79.43	82.98	81.16	60.08	61.88	60.55	66.17	71.89	68.33	68.56	72.25	70.01
	HI	82.11	82.29	82.18	63.18	63.07	62.6	73.93	78.68	75.94	73.07	74.68	73.57
	KO	84.79	87.72	86.22	55.74	58.53	56.38	66.13	70.77	68.25	68.88	72.34	70.29
	NL	89.03	90.18	89.59	61.04	62.53	61.12	68.23	72.6	70.03	72.77	75.1	73.58
	RU	79.89	82.87	81.35	62.96	62.97	62.03	73.59	72.84	72.74	72.14	72.89	72.04
	TR	86.6	89.3	87.92	62.78	63.69	62.28	72.14	73.33	72.59	73.84	75.44	74.26
	ZH	85.06	85.05	85.05	55.35	55.0	53.08	64.74	69.65	66.51	68.38	69.9	68.21
	Avg.	84.88	86.44	85.64	60.27	61.13	59.9	68.18	71.45	69.4	71.11	73.01	71.64
9. Sliced	**BN**	82.89	83.11	82.97	59.22	57.37	57.58	56.15	57.69	56.51	66.09	66.06	65.68
	DE	90.83	90.88	90.85	61.76	62.8	61.72	70.31	71.86	70.87	74.3	75.18	74.48
	EN	87.58	87.91	87.74	57.15	57.24	56.36	67.62	68.84	67.97	70.78	71.33	70.69
	ES	85.59	85.81	85.69	60.51	61.49	60.28	69.32	73.36	70.77	71.81	73.55	72.25
	FA	79.75	82.35	81.03	59.67	61.46	60.2	63.45	68.85	65.38	67.62	70.89	68.87
	HI	81.16	81.29	81.21	62.7	61.39	61.23	73.22	76.61	74.29	72.36	73.1	72.24
	KO	84.49	86.53	85.5	55.3	57.96	55.91	64.05	68.25	65.92	67.95	70.91	69.11
	NL	88.7	89.21	88.95	61.48	62.92	61.64	69.04	72.58	70.42	73.07	74.9	73.67
	RU	80.34	82.38	81.34	64.09	64.0	63.41	74.2	73.51	73.43	72.88	73.29	72.73
	TR	86.6	88.45	87.5	62.71	63.5	62.27	70.09	70.85	70.33	73.13	74.27	73.36
	ZH	85.86	84.44	85.1	58.35	55.95	54.95	65.57	68.28	65.85	69.93	69.56	68.63
	Avg.	84.89	85.67	85.26	60.27	60.55	59.6	67.55	70.06	68.34	70.9	72.09	71.06
10. IIE_KDSEC	**BN**	82.17	81.64	81.86	59.9	56.97	57.87	58.43	57.74	57.53	66.84	65.45	65.75
	DE	90.17	90.43	90.3	62.06	62.48	61.57	69.19	72.44	70.21	73.8	75.12	74.02
	EN	87.3	87.48	87.39	57.62	57.44	56.75	66.92	69.35	67.66	70.61	71.42	70.6
	ES	85.51	85.07	85.28	61.75	62.09	61.25	67.08	73.81	69.43	71.45	73.66	71.98
	FA	79.36	81.72	80.5	60.97	60.08	59.99	63.57	68.11	64.71	67.96	69.97	68.4
	HI	81.06	80.14	80.59	63.67	61.61	62.21	73.87	77.78	75.2	72.86	73.18	72.66
	KO	84.54	86.11	85.3	57.64	58.01	57.34	65.12	70.12	67.09	69.1	71.41	69.91
	NL	88.41	89.01	88.7	61.86	62.45	61.45	68.59	71.4	69.5	72.95	74.28	73.22
	RU	79.81	81.62	80.7	65.4	64.32	64.15	72.79	72.06	71.48	72.67	72.66	72.11
	TR	86.37	88.14	87.24	63.1	62.88	62.25	68.35	70.43	68.9	72.61	73.82	72.8
	ZH	85.15	82.83	83.91	58.91	54.57	54.43	66.85	68.59	66.58	70.31	68.66	68.31
	Avg.	84.53	84.93	84.71	61.17	60.26	59.93	67.34	70.17	68.03	71.01	71.78	70.89

11. B.E.P	**BN**	82.24	82.26	82.23	60.92	57.93	58.85	59.92	56.44	57.4	67.7	65.54	66.16
	DE	90.68	90.47	90.57	61.17	61.23	60.51	70.51	71.47	70.36	74.12	74.39	73.82
	EN	87.72	87.39	87.55	57.43	55.95	55.79	66.24	66.78	65.91	70.46	70.04	69.75
	ES	86.07	85.34	85.7	61.03	60.65	60.09	68.97	73.12	70.36	72.02	73.04	72.05
	FA	80.28	80.42	80.34	59.67	59.43	58.96	65.17	66.57	65.2	68.37	68.81	68.17
	HI	80.81	80.26	80.5	63.6	60.79	61.29	76.33	75.39	75.51	73.58	72.15	72.43
	KO	84.97	85.84	85.4	57.15	57.65	56.65	65.8	68.49	66.7	69.31	70.66	69.58
	NL	88.97	88.79	88.87	61.39	61.5	60.85	67.05	70.13	68.06	72.47	73.47	72.59
	RU	81.67	80.97	81.32	64.11	63.15	62.83	73.87	72.41	72.28	73.22	72.18	72.14
	TR	86.98	87.56	87.26	62.79	61.74	61.23	72.42	71.26	71.45	74.06	73.52	73.31
	ZH	84.83	83.46	84.12	58.83	53.95	53.3	67.71	67.5	65.38	70.46	68.3	67.6
	Avg.	85.02	84.8	84.9	60.74	59.45	59.12	68.54	69.05	68.06	71.43	71.1	70.69
12. OPDAI	**BN**	82.79	82.06	82.42	59.68	50.37	54.03	58.27	53.76	55.55	66.92	62.06	64.0
	DE	90.97	90.33	90.65	59.6	56.78	57.75	67.82	68.07	67.45	72.8	71.73	71.95
	EN	87.98	87.95	87.97	51.85	49.03	49.94	63.56	65.24	64.06	67.8	67.4	67.32
	ES	84.12	83.18	83.64	53.54	50.86	51.5	64.2	66.7	64.98	67.29	66.91	66.71
	FA	80.9	81.79	81.33	57.93	52.49	54.77	62.88	63.56	62.62	67.24	65.95	66.24
	HI	81.59	80.68	81.13	61.16	53.24	56.5	72.25	72.98	72.45	71.67	68.97	70.02
	KO	85.44	86.11	85.77	58.73	51.67	54.56	65.73	65.7	65.58	69.96	67.83	68.63
	NL	89.39	89.03	89.21	58.69	56.36	57.07	66.34	68.08	66.93	71.47	71.16	71.07
	RU	81.44	81.61	81.53	63.88	55.46	58.98	70.79	66.93	68.21	72.04	68.0	69.57
	TR	86.58	87.47	87.02	60.69	56.24	57.92	66.58	66.02	66.1	71.28	69.91	70.35
	ZH	86.47	85.8	86.08	68.15	66.7	66.1	82.63	84.11	83.11	79.08	78.87	78.43
	Avg.	85.24	85.09	85.16	59.45	54.47	56.28	67.37	67.38	67.0	70.69	68.98	69.48
13. brotherhood	**BN**	82.62	82.8	82.7	60.63	57.71	58.47	57.14	57.33	56.89	66.79	65.95	66.02
	DE	90.46	89.99	90.22	59.96	59.81	59.29	68.52	70.3	68.86	72.98	73.37	72.79
	EN	87.53	87.65	87.59	56.32	55.8	55.43	63.92	66.72	64.83	69.26	70.06	69.28
	ES	83.11	82.3	82.7	57.19	56.52	56.11	63.23	69.44	65.8	67.85	69.42	68.2
	FA	80.22	82.03	81.11	57.32	56.51	56.39	61.97	66.71	63.37	66.5	68.42	66.95
	HI	81.19	80.59	80.88	61.79	59.02	59.66	72.26	76.23	73.69	71.74	71.95	71.41
	KO	85.03	86.45	85.74	56.45	56.91	55.95	63.5	68.34	65.33	68.33	70.57	69.01
	NL	88.79	88.94	88.86	59.28	59.62	58.89	64.91	68.78	66.29	70.99	72.44	71.35
	RU	80.39	81.61	80.99	60.8	59.46	59.18	65.77	67.56	66.18	68.99	69.54	68.79
	TR	86.41	88.05	87.22	60.66	59.96	59.59	67.52	70.86	68.66	71.53	72.95	71.82
	ZH	85.77	83.95	84.8	58.11	53.34	53.1	64.98	68.96	66.02	69.62	68.75	67.97
	Avg.	84.68	84.94	84.8	58.96	57.7	57.46	64.88	68.29	65.99	69.51	70.31	69.42
14. MarSan	**BN**	81.56	81.36	81.42	59.37	56.93	57.76	55.92	56.57	54.9	65.62	64.95	64.69
	DE	90.43	90.55	90.46	62.24	63.31	62.27	69.96	73.92	70.82	74.21	75.93	74.52
	EN	86.88	87.4	87.1	57.16	56.77	56.27	63.0	67.82	64.38	69.02	70.66	69.25
	ES	85.62	85.01	85.27	59.28	60.47	59.45	64.83	73.28	67.66	69.91	72.92	70.79
	FA	78.55	79.51	78.97	55.59	54.98	54.96	58.27	63.37	59.0	64.14	65.95	64.31
	HI	79.63	78.53	79.01	58.95	57.04	57.49	71.59	74.39	71.75	70.06	69.99	69.42
	KO	83.58	84.05	83.78	55.49	55.08	54.6	61.29	65.69	62.36	66.79	68.28	66.91
	NL	88.09	88.54	88.29	60.07	61.31	60.12	67.07	72.15	68.55	71.74	74.0	72.32
	RU	80.59	79.57	80.04	61.8	59.8	60.18	69.08	70.0	67.99	70.49	69.79	69.4
	TR	85.94	86.8	86.34	62.51	62.63	62.0	65.92	69.66	67.05	71.46	73.03	71.8
	ZH	84.14	83.3	83.68	58.8	55.43	55.24	66.48	70.16	67.09	69.81	69.63	68.67
	Avg.	84.09	84.06	84.03	59.21	58.52	58.21	64.86	68.82	65.6	69.39	70.47	69.28
15. Infrrd.ai	**BN**	79.97	80.51	80.18	59.53	55.61	56.49	58.78	55.59	56.88	66.09	63.9	64.52
	DE	88.97	88.84	88.9	60.59	59.98	59.48	67.99	68.85	67.77	72.52	72.56	72.05
	EN	86.12	86.47	86.29	57.76	55.14	55.44	65.26	67.91	66.29	69.72	69.84	69.34
	ES	84.4	84.64	84.51	60.32	59.01	58.59	64.46	69.94	66.42	69.73	71.2	69.84
	FA	78.94	81.21	80.04	58.66	56.92	57.06	63.36	65.37	63.88	66.99	67.83	66.99
	HI	78.69	78.11	78.37	61.63	58.28	59.05	72.43	76.04	73.63	70.92	70.81	70.35
	KO	82.69	84.76	83.69	57.2	56.44	55.88	64.15	67.99	65.73	68.01	69.73	68.43
	NL	87.77	87.82	87.79	61.02	60.09	59.71	67.85	70.08	68.79	72.21	72.66	72.1
	RU	78.57	81.61	80.04	63.47	61.07	61.3	72.86	71.51	71.29	71.63	71.4	70.88
	TR	84.87	87.51	86.16	61.93	60.12	59.97	67.84	68.31	67.87	71.54	71.98	71.33
	ZH	81.81	81.9	81.78	56.59	53.22	52.84	62.12	65.99	62.94	66.84	67.04	65.86
	Avg.	82.98	83.94	83.43	59.88	57.81	57.8	66.1	67.96	66.5	69.65	69.9	69.24

16. HaveNoIdea	**BN**	82.01	80.5	81.17	57.25	51.15	52.6	56.97	52.24	53.64	65.41	61.3	62.47
	DE	91.22	90.04	90.61	63.33	61.86	61.18	75.42	72.88	73.66	76.66	74.93	75.15
	EN	88.34	87.01	87.65	58.52	55.63	55.5	67.91	66.85	67.23	71.59	69.83	70.13
	ES	87.7	84.1	85.82	61.96	59.84	59.27	70.31	70.84	70.51	73.33	71.59	71.87
	FA	79.44	76.16	77.65	53.27	50.53	50.87	60.15	60.02	58.97	64.28	62.24	62.49
	HI	80.84	78.05	79.34	57.6	53.9	54.46	74.25	71.63	72.43	70.89	67.86	68.74
	KO	83.01	81.49	82.16	51.96	47.59	48.03	63.46	63.07	62.57	66.14	64.05	64.26
	NL	89.99	87.36	88.63	63.26	61.13	60.44	72.94	70.77	71.56	75.39	73.09	73.55
	RU	82.66	80.28	81.41	65.65	61.35	61.82	77.69	70.21	73.32	75.33	70.61	72.19
	TR	87.22	86.57	86.87	63.86	61.5	61.22	72.33	68.81	70.2	74.47	72.29	72.76
	ZH	83.1	71.09	76.34	56.85	48.15	49.48	65.75	63.87	63.44	68.57	61.03	63.09
	Avg.	85.05	82.06	83.42	59.41	55.69	55.9	68.83	66.47	67.05	71.1	68.07	68.79
17. EURECOM	**BN**	79.9	78.85	79.33	54.31	53.41	53.54	54.12	55.24	53.31	62.78	62.5	62.06
	DE	89.29	89.02	89.15	59.53	60.47	59.52	67.38	71.35	68.7	72.06	73.61	72.46
	EN	86.11	86.01	86.06	55.55	55.49	55.0	62.94	66.83	64.13	68.2	69.44	68.4
	ES	85.28	84.32	84.8	59.56	60.08	59.23	65.83	69.89	67.37	70.22	71.43	70.47
	FA	77.82	78.34	78.06	55.33	54.92	54.87	57.17	62.99	58.75	63.44	65.42	63.89
	HI	78.35	76.35	77.29	55.74	54.91	55.04	69.25	72.53	69.84	67.78	67.93	67.39
	KO	82.15	83.8	82.96	53.99	53.35	53.26	58.34	65.45	61.01	64.83	67.53	65.74
	NL	87.59	87.71	87.65	59.47	60.39	59.45	65.7	69.59	67.04	70.92	72.56	71.38
	RU	79.1	79.66	79.37	60.59	59.66	59.48	69.16	69.25	68.08	69.62	69.52	68.97
	TR	84.83	86.71	85.76	60.28	60.43	59.82	65.44	69.03	66.69	70.18	72.05	70.75
	ZH	83.34	82.02	82.66	57.82	54.94	54.48	64.36	68.51	65.12	68.51	68.49	67.42
	Avg.	83.07	82.98	83.01	57.47	57.1	56.7	63.61	67.33	64.55	68.05	69.13	68.08
18. MaChAmp	**BN**	78.09	79.68	78.87	52.07	51.38	50.89	53.49	53.03	52.46	61.22	61.36	60.74
	DE	89.24	89.56	89.4	62.45	64.2	62.79	70.25	72.33	71.08	73.98	75.36	74.42
	EN	85.93	86.97	86.44	58.08	58.48	57.7	64.65	69.39	66.2	69.56	71.62	70.11
	ES	84.81	85.53	85.15	61.77	63.48	61.94	67.16	72.73	69.23	71.25	73.91	72.1
	FA	75.24	78.49	76.82	49.14	51.84	49.9	49.79	58.31	52.85	58.06	62.88	59.86
	HI	76.69	78.04	77.35	56.29	56.45	55.72	67.68	73.68	70.04	66.89	69.39	67.7
	KO	81.05	83.76	82.38	49.07	51.05	49.34	53.97	62.91	57.36	61.36	65.91	63.03
	NL	87.56	88.34	87.94	61.45	63.0	61.62	68.38	72.58	70.23	72.46	74.64	73.26
	RU	78.31	81.24	79.74	62.04	62.51	61.46	68.92	68.97	68.14	69.75	70.91	69.78
	TR	84.48	87.23	85.81	61.76	63.15	61.68	64.52	67.95	65.9	70.25	72.78	71.13
	ZH	79.96	80.37	80.13	49.66	48.44	46.96	58.89	63.62	59.88	62.83	64.15	62.32
	Avg.	81.94	83.56	82.73	56.71	57.63	56.36	62.52	66.86	63.94	67.06	69.36	67.68
19. YNUNLP	**BN**	79.19	77.07	78.05	57.95	51.59	53.7	55.19	55.49	53.56	64.11	61.38	61.77
	DE	87.61	86.41	86.99	59.12	58.75	57.98	69.29	69.04	67.19	72.01	71.4	70.72
	EN	85.67	84.19	84.9	54.77	52.48	52.38	64.73	63.75	63.04	68.39	66.81	66.77
	ES	85.22	81.31	83.2	57.7	56.3	55.91	66.15	68.41	66.0	69.69	68.67	68.37
	FA	79.42	75.53	77.39	56.48	54.03	54.38	63.67	64.01	62.28	66.52	64.52	64.68
	HI	77.69	74.36	75.93	59.47	56.21	57.07	71.12	73.9	71.03	69.43	68.16	68.01
	KO	81.56	80.87	81.18	56.52	54.11	54.28	59.42	65.49	60.53	65.83	66.82	65.33
	NL	87.43	84.81	86.08	58.73	57.78	57.11	67.24	67.15	66.17	71.14	69.91	69.78
	RU	81.74	75.93	78.72	62.54	58.78	59.31	70.27	68.38	67.57	71.52	67.69	68.53
	TR	84.4	83.52	83.95	60.08	57.53	57.37	66.84	67.32	65.46	70.44	69.46	68.93
	ZH	80.19	77.99	79.0	51.79	47.17	47.22	61.37	63.07	61.05	64.45	62.75	62.42
	Avg.	82.74	80.18	81.4	57.74	54.98	55.16	65.03	66.0	63.99	68.5	67.05	66.85
20. DS4DH	**BN**	79.42	75.85	77.57	53.09	45.66	47.87	54.34	48.87	50.71	62.29	56.79	58.71
	DE	88.21	85.58	86.87	55.3	52.95	52.98	66.38	63.79	64.55	69.96	67.44	68.13
	EN	86.21	82.8	84.44	52.18	48.88	49.18	61.27	57.99	59.06	66.55	63.22	64.23
	ES	85.31	80.72	82.92	55.12	51.71	52.0	63.9	62.0	62.66	68.11	64.81	65.86
	FA	81.89	74.94	78.22	56.3	51.66	53.26	63.25	61.66	61.71	67.15	62.75	64.4
	HI	79.81	74.44	76.95	57.56	51.0	53.0	71.66	69.79	70.06	69.68	65.08	66.67
	KO	84.45	81.23	82.8	52.55	49.93	49.99	62.8	60.73	61.42	66.6	63.96	64.74
	NL	88.16	84.7	86.38	55.24	52.71	52.7	65.16	62.9	63.74	69.52	66.77	67.61
	RU	82.52	75.96	79.09	59.17	54.36	55.35	68.95	62.22	64.87	70.22	64.18	66.44
	TR	86.83	83.17	84.96	56.56	52.84	53.15	67.34	62.2	64.38	70.25	66.07	67.5
	ZH	83.24	78.17	80.52	53.77	47.15	48.15	61.73	61.21	60.78	66.25	62.18	63.15
	Avg.	84.19	79.78	81.88	55.17	50.8	51.6	64.25	61.21	62.18	67.87	63.93	65.22

21. UPB	**BN**	79.84	80.64	80.2	52.51	51.07	50.88	52.09	50.36	49.95	61.48	60.69	60.34
	DE	87.67	88.32	87.99	52.24	53.77	52.18	60.43	65.32	61.79	66.78	69.14	67.32
	EN	85.14	86.16	85.64	50.6	51.15	50.14	57.73	62.56	59.02	64.49	66.62	64.93
	ES	83.81	84.02	83.91	52.05	52.73	51.33	57.96	64.4	60.21	64.61	67.05	65.15
	FA	77.06	80.14	78.57	53.26	53.7	52.46	57.45	62.04	58.86	62.59	65.29	63.3
	HI	77.53	77.68	77.58	55.0	54.26	53.98	65.64	69.92	66.47	66.06	67.29	66.01
	KO	81.16	84.35	82.71	50.74	51.9	50.41	56.73	62.38	58.56	62.88	66.21	63.89
	NL	86.73	87.61	87.14	53.31	53.99	52.73	59.68	64.38	61.06	66.57	68.66	66.98
	RU	77.93	80.93	79.39	57.27	57.01	56.06	63.2	65.12	62.73	66.13	67.69	66.06
	TR	83.19	85.37	84.26	53.39	52.94	52.04	58.02	59.78	57.85	64.87	66.03	64.72
	ZH	82.89	82.21	82.49	52.98	50.47	48.96	58.15	63.05	58.42	64.67	65.24	63.29
	Avg.	82.09	83.4	82.72	53.03	53.0	51.92	58.83	62.66	59.54	64.65	66.36	64.73
22. CSECU-DSG	**BN**	77.5	76.82	77.12	51.42	46.24	47.93	52.23	52.55	51.63	60.38	58.54	58.89
	DE	87.12	86.34	86.72	52.5	53.92	52.52	62.89	65.57	63.82	67.5	68.61	67.69
	EN	84.06	83.34	83.69	49.84	49.8	48.91	58.88	60.99	59.47	64.26	64.71	64.02
	ES	82.93	81.62	82.26	51.62	52.18	51.02	60.08	64.29	61.72	64.88	66.03	65.0
	FA	77.28	77.87	77.55	52.12	51.76	51.45	58.85	63.0	59.66	62.75	64.21	62.88
	HI	76.99	75.54	76.23	56.3	54.03	54.51	67.76	70.11	67.08	67.01	66.56	65.94
	KO	80.51	82.81	81.62	49.44	50.17	49.04	58.58	61.56	59.6	62.85	64.85	63.42
	NL	86.53	85.57	86.05	53.53	54.14	53.05	62.14	65.41	63.34	67.4	68.37	67.48
	RU	77.98	78.23	78.09	54.22	53.26	52.63	66.57	63.92	64.18	66.26	65.14	64.97
	TR	83.99	84.09	84.03	54.47	54.08	53.26	61.25	63.19	61.46	66.57	67.12	66.25
	ZH	81.91	80.29	81.03	50.09	48.27	47.08	56.59	60.81	57.6	62.86	63.12	61.9
	Avg.	81.53	81.14	81.31	52.32	51.62	51.04	60.53	62.85	60.87	64.79	65.21	64.4
23. NSU-AI	**BN**	77.78	77.27	77.49	52.46	49.81	50.37	49.13	48.34	48.07	59.79	58.47	58.64
	DE	87.9	87.79	87.84	53.35	53.64	52.74	58.78	63.51	60.52	66.68	68.31	67.03
	EN	84.65	85.02	84.83	50.6	50.22	49.68	54.44	59.15	56.25	63.23	64.8	63.59
	ES	83.59	82.96	83.27	52.78	52.49	51.58	55.14	61.31	57.59	63.84	65.59	64.15
	FA	75.33	77.73	76.47	50.9	51.12	50.45	54.1	60.42	56.47	60.11	63.09	61.13
	HI	77.1	76.02	76.53	55.3	52.65	53.21	65.03	67.59	65.28	65.81	65.42	65.01
	KO	81.62	83.14	82.35	53.53	52.63	52.3	57.73	61.64	59.15	64.29	65.8	64.6
	NL	87.03	86.79	86.91	54.37	54.41	53.51	59.23	64.16	61.11	66.88	68.45	67.17
	RU	78.68	79.38	79.01	58.83	56.94	57.01	64.38	66.72	64.85	67.29	67.68	66.96
	TR	83.23	84.38	83.79	54.85	53.59	53.2	57.77	59.41	58.08	65.28	65.79	65.02
	ZH	81.64	80.89	81.19	52.25	49.82	48.45	60.37	62.88	59.94	64.76	64.53	63.19
	Avg.	81.69	81.94	81.79	53.57	52.48	52.05	57.83	61.38	58.85	64.36	65.27	64.23
24. SPDB Innovation Lab	**BN**	75.59	75.54	75.54	45.87	41.92	42.27	53.4	47.64	49.55	58.29	55.03	55.79
	DE	85.92	84.87	85.38	50.94	51.48	50.04	65.47	64.83	64.63	67.45	67.06	66.68
	EN	83.15	82.06	82.59	47.89	46.77	46.15	59.29	58.03	58.41	63.44	62.29	62.38
	ES	82.49	80.59	81.5	49.9	49.15	48.04	61.47	61.4	61.13	64.62	63.71	63.56
	FA	76.55	76.23	76.35	50.3	48.04	48.11	58.45	57.38	56.84	61.76	60.55	60.43
	HI	75.35	73.81	74.51	55.39	51.67	52.38	72.27	68.99	70.21	67.67	64.83	65.7
	KO	79.77	80.56	80.14	47.68	47.32	46.18	58.55	59.16	58.25	62.0	62.34	61.52
	NL	85.65	84.98	85.31	51.67	52.05	50.66	66.56	63.74	64.68	67.96	66.92	66.88
	RU	78.46	76.82	77.63	56.74	54.1	53.82	68.18	60.85	63.58	67.8	63.92	65.01
	TR	83.54	83.65	83.58	53.9	53.08	52.11	64.62	61.86	62.75	67.35	66.2	66.15
	ZH	81.39	78.83	80.02	51.15	46.6	46.01	60.06	59.38	57.95	64.2	61.61	61.33
	Avg.	80.71	79.81	80.23	51.04	49.29	48.71	62.57	60.3	60.73	64.78	63.13	63.22
25. L3i	**BN**	73.57	72.55	72.84	47.0	42.43	43.79	38.16	39.75	38.04	52.91	51.58	51.56
	DE	89.55	88.49	89.01	57.48	56.04	55.68	62.63	65.6	63.51	69.89	70.04	69.4
	EN	85.81	85.55	85.67	53.37	50.62	50.86	54.89	55.68	54.55	64.69	63.95	63.69
	ES	84.27	83.74	83.98	55.13	53.47	53.37	57.22	62.82	59.23	65.54	66.68	65.53
	FA	75.71	75.4	75.51	47.17	43.21	44.43	48.19	53.31	49.09	57.02	57.31	56.34
	HI	70.72	68.68	69.54	46.78	42.45	43.33	49.58	56.2	50.6	55.69	55.78	54.49
	KO	80.02	78.75	79.37	45.4	41.19	42.21	47.43	50.83	47.26	57.62	56.92	56.28
	NL	87.12	87.18	87.13	57.05	55.68	55.37	60.83	64.2	61.96	68.33	69.02	68.15
	RU	77.55	77.69	77.59	54.56	50.71	51.51	58.71	60.39	57.88	63.6	62.93	62.33
	TR	83.21	85.44	84.3	54.03	50.87	51.34	53.43	56.44	53.88	63.56	64.25	63.18
	ZH	82.39	80.96	81.59	54.62	50.17	49.54	57.08	61.1	56.76	64.7	64.08	62.63
	Avg.	80.9	80.4	80.59	52.05	48.8	49.22	53.47	56.94	53.89	62.14	62.05	61.23

26. Baseline	**BN**	73.32	74.14	73.54	39.28	36.45	37.49	39.87	41.36	40.03	50.83	50.65	50.35
	DE	83.69	82.35	82.93	41.91	40.47	40.63	49.96	53.5	50.94	58.52	58.77	58.17
	EN	79.78	78.97	79.29	41.19	38.93	39.36	44.36	49.88	45.8	55.11	55.93	54.82
	ES	67.52	64.25	65.53	36.1	33.13	33.33	36.75	39.54	37.08	46.79	45.64	45.32
	FA	47.01	45.37	45.46	32.2	26.58	28.23	34.42	35.25	34.46	37.88	35.74	36.05
	HI	72.06	71.1	71.38	44.95	41.59	42.63	52.53	55.79	53.4	56.51	56.16	55.8
	KO	42.84	37.41	39.8	39.07	34.38	35.76	35.81	41.62	36.13	39.24	37.8	37.23
	NL	68.94	65.02	66.73	40.6	38.48	38.59	45.62	49.21	46.64	51.72	50.9	50.65
	RU	51.22	53.0	51.17	35.76	30.7	32.65	32.34	34.37	32.27	39.77	39.36	38.7
	TR	57.51	53.81	55.17	36.67	33.55	34.09	37.01	39.82	37.62	43.73	42.39	42.29
	ZH	77.29	77.5	77.36	42.83	42.68	41.32	46.75	54.27	49.89	55.62	58.15	56.19
	Avg.	65.56	63.9	64.4	39.14	36.09	36.73	41.4	44.96	42.21	48.7	48.32	47.78

Table 18: Detailed results for the Multi-lingual track. Full form of B.E.P. is BaselineExtendingPokemons.

LMN at SemEval-2022 Task 11: A Transformer-based System for English Named Entity Recognition

Ngoc Minh Lai
Van Ho Middle School
`nl834771@gmail.com`

Abstract

Processing complex and ambiguous named entities is a challenging research problem, but it has not received sufficient attention from the natural language processing community. In this short paper, we present our participation in the English track of SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition. Inspired by the recent advances in pretrained Transformer language models, we propose a simple yet effective Transformer-based baseline for the task. Despite its simplicity, our proposed approach shows competitive results in the leaderboard as we ranked 12 over 30 teams. Our system achieved a macro F1 score of 72.50% on the held-out test set. We have also explored a data augmentation approach using entity linking. While the approach does not improve the final performance, we also discuss it in this paper.

1 Introduction

Recognizing complex named entities (NEs) is a challenging research problem, but it has not received sufficient attention from the natural language processing community (Meng et al., 2021a; Fetahu et al., 2021). Complex NEs can be complex noun phrases (e.g., *National Baseball Hall of Fame and Museum*), gerunds (e.g., *Saving Private Ryan*), infinitives (e.g., *To Build a Fire*), or even full clauses (e.g., *I Capture The Castle*). This ambiguity makes it difficult to recognize them based on their context (Aguilar et al., 2017; Luken et al., 2018; Hanselowski et al., 2018).

In this paper, we describe our participation in the English track of SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (Malmasi et al., 2022a,b). Inspired by the recent success of Transformer-based pre-trained language models in many NLP tasks (Devlin et al., 2019; Joshi et al., 2019; Lai et al., 2019; Joshi et al., 2020; Tran et al., 2020; Yu et al., 2020; Wen et al., 2021; Lai et al., 2021; Monaikul et al., 2021), we propose a simple but effective Transformer-based baseline for the task. Despite its simplicity, our proposed approach shows promising results: the official ranking indicated that our system achieved a macro F_1 score of 72.50% on the test set and ranked 12th out of 30 teams. We have also explored a data augmentation approach using entity linking. While the approach does not improve the final performance, we also discuss it in this paper.

In the following sections, we first describe the related work in Section 2 and the proposed method in Section 3. We then describe the experiments and their results in Section 4. Finally, Section 5 concludes this work and discusses potential future research directions.

2 Related Work

Many previous named entity recognition (NER) methods are based on the sequence labeling approach (Collobert et al., 2011; Ma and Hovy, 2016; Lample et al., 2016; Chiu and Nichols, 2016; Lee et al., 2019; Yang et al., 2018; Yang and Zhang, 2018; Lai et al., 2020a; Li et al., 2020). For example, Collobert et al. (2011) introduced a neural architecture that uses convolutional neural networks (CNNs) to encode tokens combined with a CRF layer for the classification. Many other studies used recurrent neural networks (RNNs) instead of CNNs to encode the input and a CRF for the prediction (Ma and Hovy, 2016; Lample et al., 2016). With the recent rise of pre-trained language models, recent NER models typically make use of context-dependent embeddings such as ELMo (Peters et al., 2018) or BERT (Devlin et al., 2019).

While neural-based models have achieved impressive results on popular benchmark datasets like CoNLL03 and OntoNotes (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003; Pradhan et al., 2012), these models typically do not perform well on complex/unseen entities (Augenstein et al., 2017). Complex named entities (e.g., titles

Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 1438 - 1443
July 14-15, 2022 ©2022 Association for Computational Linguistics

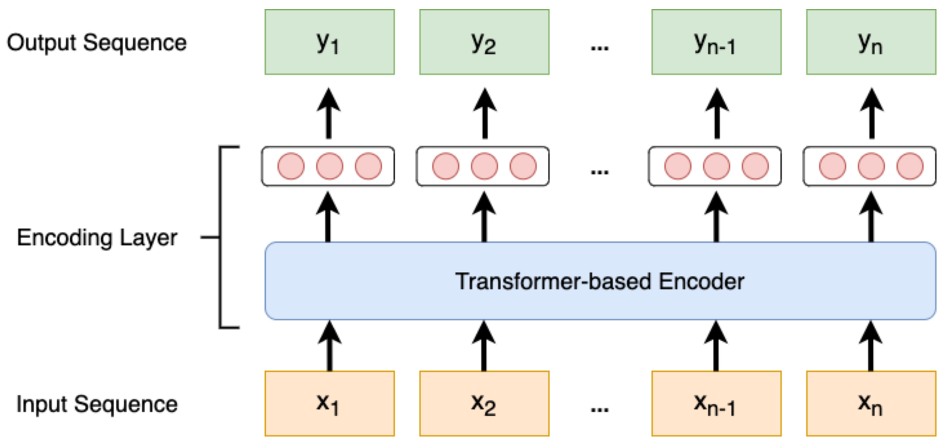


Figure 1: Overview of our Transformer-based model.

Tag	Description
`{B,I}-PER`	A named entity of a *person*
`{B,I}-LOC`	A named entity of a *location*
`{B,I}-GRP`	A named entity of a *group*
`{B,I}-CORP`	A named entity of a *corporation*
`{B,I}-PROD`	A named entity of a *product*
`{B,I}-CW`	A named entity of a *creative work*
`O`	Not a named entity

Table 1: The label set.

of creative works) are typically not simple nouns and are harder to recognize. The challenges of NER for recognizing complex entities and in low-context situations was recently outlined by Meng et al. (2021b). Other work has extended this to multilingual and code-mixed settings (Fetahu et al., 2021).

3 Method

3.1 Baseline model

Similar to many previous studies (Lample et al., 2016; Chiu and Nichols, 2016), we formulate the task as a sequence labeling problem. Given an input sequence consisting of n tokens $(x_1, ..., x_n)$, the goal is to predict a sequence of labels $(y_1, ..., y_n)$, where y_i is the label corresponding to token x_i. Table 1 describes the label set. We follow the BIO format: `B` denotes the beginning of a named entity, `I` denotes the continuation of a named entity, and `O` corresponds to tokens that are not part of any named entity.

Figure 1 shows a high-level overview of our Transformer-based model. Our model first forms a contextualized representation for each input token using a Transformer encoder (Devlin et al., 2019). Let $\mathbf{H} = (\mathbf{h}_1, ..., \mathbf{h}_n)$ be the output of the encoder where $\mathbf{h}_i \in \mathbb{R}^d$. After that, we can predict the final label y_i for each input token x_i:

$$\begin{aligned} \mathbf{y}_i &= \text{softmax}(\text{FFNN}_\theta(\mathbf{h}_i)) \\ y_i &= \arg\max_j \mathbf{y}_{ij} \end{aligned} \tag{1}$$

where FFNN_θ is a trainable feedforward network. $\mathbf{y}_i$ is the predicted probability distribution over the label set for the token x_i. The model is fine-tuned end-to-end via minimizing the typical cross-entropy loss.

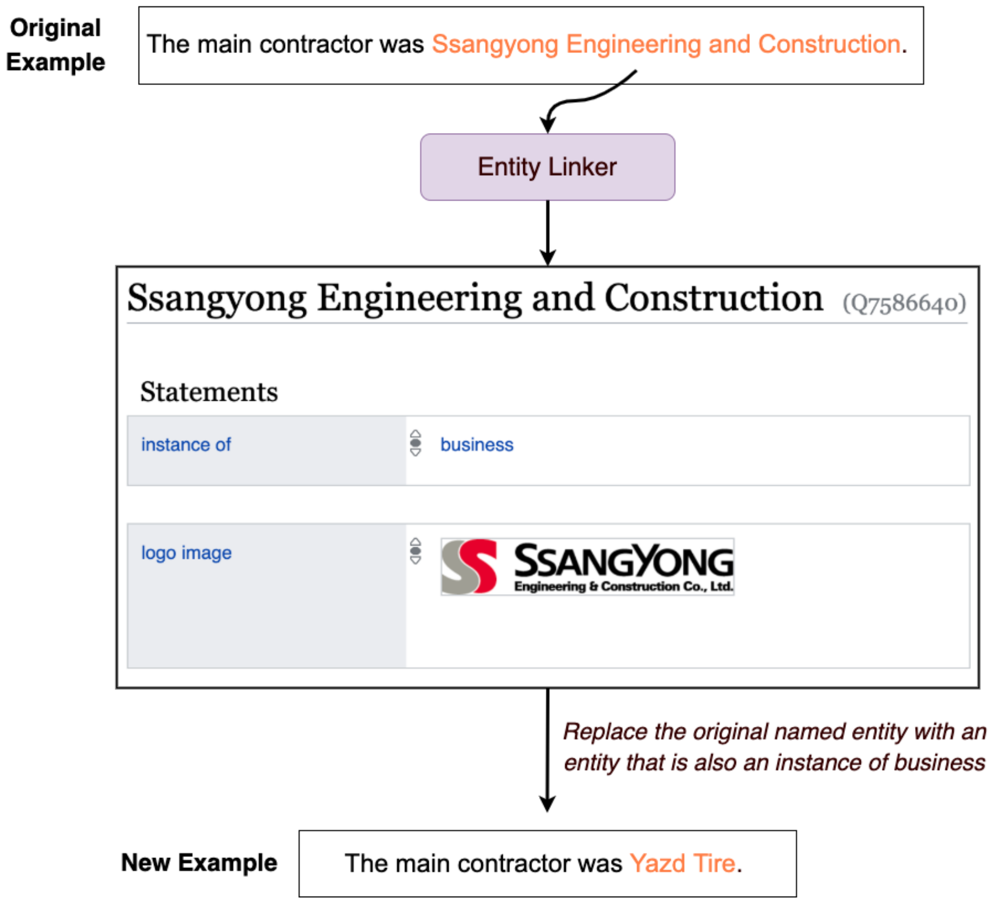
Unlike many previous studies (Lample et al., 2016; Chiu and Nichols, 2016), our model does not have a CRF layer (Lafferty et al., 2001). A recent paper suggested that when using a pretrained Transformer language model for sequence labeling, adding a CRF layer may not improve the performance substantially (Chen et al., 2019).

3.2 Data Augmentation

To increase the size of the training set, we have also experimented with a simple data augmentation approach (Figure 2). For example, consider the sentence "*The main contractor was Ssangyong Engineering and Construction.*", which is an example in the training set of the English track of MultiCoNER. In this case, "*Ssangyong Engineering and Construction*" is a named mention referring to a Korean corporation. To create a new training example, we can replace the named mention with a different entity that is also a corporation.

More specifically, in this example, we first use an entity linker[1] to link the named mention to its corresponding entity in Wikidata, a large-scale knowledge graph. From the found Wikidata page, we

[1]`https://github.com/laituan245/EL-Dockers`



Original Example
The main contractor was Ssangyong Engineering and Construction.
Entity Linker
Ssangyong Engineering and Construction (Q7586640)
Statements
instance of
business
logo image
SSANGYONG Engineering & Construction Co., Ltd.
Replace the original named entity with an entity that is also an instance of business
New Example
The main contractor was Yazd Tire.

Figure 2: Our data augmentation approach.

can extract all types of information about the entity. We can utilize these types of information to find a new entity that is different but highly similar to the original entity. For simplicity, in this work, we simply try to find a new entity of the same Wikidata type as the original entity. At the end, we will have a new example (e.g., "*The main contractor was Yazd Tire.*").

4 Results

4.1 Data and Experimental Setup

The learning rate is set to be 2e-5, and the batch size is 32. We experimented with different numbers of training epochs, 10 and 20. We use Huggingface's Transformer library (Wolf et al., 2020) to experiment with various Transformer language models:

- **BERT**. Devlin et al. (2019) introduced a language representation model named BERT, which is pre-trained using two tasks: masked language modeling (MLM) and next sentence prediction (NSP). We used the large version of BERT (i.e., bert-large-uncased) in this work.

- **RoBERTa**. Liu et al. (2019) proposed an improved recipe for training BERT models. The modifications include: (1) training the model longer, with bigger batches, over more data; (2) removing the NSP objective; (3) training on longer sequences; and (4) dynamically changing the masking pattern applied to the training data. We used the large version of RoBERTa (i.e., roberta-large) in this work.

- **ALBERT**. Lan et al. (2020) introduced ALBERT, a BERT-based model with two parameter reduction techniques: factorized embedding parameterization and cross-layer parameter sharing. We used the xxlarge version of ALBERT (i.e., albert-xxlarge-v2) in this work.

4.2 Results on the Development Set

Table 3 shows the overall results on the development set of the English track of MultiCoNER. We see that ALBERT-xxlarge trained with 20 epochs outperforms all other baseline models on the development set. As such, we use this model to generate predictions for the test set. The model achieved a

Original Example	Generated Example
the guardian described the album 's release as one of the 50 key events ...	metro described the album 's release as one of the 50 key events ...
the game uses a battery packed random-access memory in order to save progress .	the game uses a battery packed delay line memory in order to save progress .
in the end the best placed rider was wilfried cretskens who finished 61st .	in the end the best placed rider was harald andersson who finished 61st .
it was broadcast on the channel animal planet , with episodes having aired between 2001 and 2003 .	it was broadcast on the channel true4u , with episodes having aired between 2001 and 2003 .

Table 2: Some of the newly generated examples.

	Prec.	Recall	F1
RoBERTa-large (10 epochs)	85.63	87.82	86.68
BERT-large (10 epochs)	86.02	88.34	87.14
ALBERT-xxlarge (10 epochs)	86.81	88.7	87.7
ALBERT-xxlarge (20 epochs)	86.47	89.49	87.91

Table 3: Overall results on the development set. Macro scores (%) are shown.

macro F1 score of 72.50% on the held-out test set. Note that the baseline models shown in Table 3 are trained using only the original training set (without any data augmentation).

4.3 Analysis of the Data Augmentation Approach

For each example in the training set, we used the data augmentation approach (Section 3.2) to generate a new example. Table 2 shows some of the newly generated examples.

We used all of the original and newly generated examples to train a new RoBERTa-large model (the number of epochs is 10). The model performs worst than the RoBERTa-large model trained with only the original examples. Nevertheless, we still believe the approach has a lot of potential, and we leave further exploration to future work.

5 Conclusion

In future work, we plan to conduct a thorough error analysis and apply visualization techniques to understand our models better (Murugesan et al., 2019). In addition, as pretrained Transformer models are typically computationally expensive and have many parameters, we are also interested in reducing the computational complexity of our baseline models using compression techniques (Sanh et al., 2019; Lai et al., 2020b; Sun et al., 2020).

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153, Copenhagen, Denmark. Association for Computational Linguistics.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Comput. Speech Lang.*, 44:61–83.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *ArXiv*, abs/1902.10909.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

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

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Tuan Lai, Heng Ji, ChengXiang Zhai, and Quan Hung Tran. 2021. Joint biomedical entity and relation extraction with knowledge-enhanced collective inference. *arXiv preprint arXiv:2105.13456*.

Tuan Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. 2019. A gated self-attention memory network for answer selection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5953–5959, Hong Kong, China. Association for Computational Linguistics.

Tuan Manh Lai, Trung Bui, Doo Soon Kim, and Quan Hung Tran. 2020a. A joint learning approach based on self-distillation for keyphrase extraction from scientific documents. *arXiv preprint arXiv:2010.11980*.

Tuan Manh Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. 2020b. A simple but effective bert model for dialog state tracking on resource-limited systems. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8034–8038. IEEE.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *ArXiv*, abs/1909.11942.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Yangming Li, Han Li, Kaisheng Yao, and Xiaolong Li. 2020. Handling rare entities for neural sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6441–6451, Online. Association for Computational Linguistics.

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

Jackson Luken, Nanjiang Jiang, and Marie-Catherine de Marneffe. 2018. QED: A fact verification system for the FEVER shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 156–160, Brussels, Belgium. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021a. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512, Online. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021b. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of* *the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Natawut Monaikul, Giuseppe Castellucci, Simone Filice, and Oleg Rokhlenko. 2021. Continual learning for named entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13570–13577.

Sugeerth Murugesan, Sana Malik, Fan Du, Eunyee Koh, and Tuan Manh Lai. 2019. Deepcompare: Visual and interactive comparison of deep learning model performance. *IEEE computer graphics and applications*, 39(5):47–59.

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

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

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

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

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

Quan Hung Tran, Nhan Dam, Tuan Lai, Franck Dernoncourt, Trung Le, Nham Le, and Dinh Phung. 2020. Explain by evidence: An explainable memory-based neural network for question answering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5205–5210, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Haoyang Wen, Ying Lin, Tuan Lai, Xiaoman Pan, Sha Li, Xudong Lin, Ben Zhou, Manling Li, Haoyu Wang, Hongming Zhang, et al. 2021. Resin: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 133–143.

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

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jie Yang and Yue Zhang. 2018. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, System Demonstrations*, pages 74–79, Melbourne, Australia. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

PA Ph&Tech at SemEval-2022 Task 11: NER Task with Ensemble Embedding from Reinforcement Learning

Qizhi LIN, Xiaopeng WANG, Xiandi JIANG, Benqi WANG, Qifeng XIAO
Ping An Puhui Enterprise Management Co. Ltd, Shanghai, China
linqizhi090@lu.com, wangxiaopeng069@lu.com
Changyong HOU, Yixuan QIAO, Jun WANG, Peng JIANG
Ping An Healthcare Technology, Beijing, China
wangjun916@pingan.com.cn

Abstract

From pretrained contextual embedding to document-level embedding, the selection and construction of embedding have drawn more and more attention in the NER domain in recent research. This paper aims to discuss the performance of ensemble embeddings on complex NER tasks. Enlightened by Wang's methodology, we try to replicate the dominating power of ensemble models with reinforcement learning optimizor on plain NER tasks to complex ones. Based on the composition of semeval dataset, the performance of the applied model is tested on lower-context, QA, and search query scenarios together with its zero-shot learning ability. Results show that with abundant training data, the model can achieve similar performance on lower-context cases compared to plain NER cases, but can barely transfer the performance to other scenarios in the test phase.

1 Introduction

Named Entity Recognition (NER) as a typical topic in the NLP field, has displayed numerous outstanding outcomes in recent years, especially benefiting from development of pretrained models. However, there are still challenging aspects that remain to be tackled, such as short texts (low-context), emerging entities, and complex entities. In this task, corpus from low-context, QA and search query scenarios are collected to represent different complex NER tasks (Meng et al., 2021). Wang et al. (2021) proposed an automated concatenation model on plain NER tasks with benchmark dataset like CONLL03, which automatically generates optimized concatenation of stack embeddings with reinforcement learning strategies for structure prediction tasks like NER. A similar methodology is applied on the given dataset to test if the challenges mentioned above can be properly addressed.

2 Related Work

The development of sequence tagging models can be concluded mainly into two separate parts: encoder and decoder. Ever since the BiLSTM-CRF model (Ma and Hovy, 2016) was brought up, it has been widely used as the decoder end of NER models, while researchers shifted increasing attention to the structure of encoder. Various categories of embeddings have been raised, including character embeddings, non-contextual embedding like word2vec (Mikolov et al., 2013) and pretrained contextualized embeddings like ELMo (Peters et al., 2018) and Flair (Akbik et al., 2018). With the emergence of Transformers, the performance of large pretrained contextualized models (Devlin et al., 2019)have sweeped the leaderboard in lots of NLP tasks. In terms of language, Multilingual BERT(M-BERT) (Pires et al., 2019) demonstrates excellent representation of multilingual embeddings, which was later surpassed by XLM-R (Conneau et al., 2020), a more powerful and comprehensive multilingual model. To better allocate these embedding methods, many recent researchers have tried different methods like ensemble, weighting and concatenation. Automated concatenation (Wang et al., 2021) is a superior method that automatically generates optimized concentration of stack embeddings with reinforcement learning strategies. It uses result accuracy as reward for the controller to decide which embeddings to drop.

3 Data

As described by Fetahu et al. (2021), the dataset (Malmasi et al., 2022a) of this semeval task (Malmasi et al., 2022b) mainly consists of three sources: Low-Context Wikipedia, MS-MARCO Question (Bajaj et al., 2018) and ORCAS Search Query (Craswell et al., 2020). Sentences from Wikipedia are parsed and linked pages are resolved to their respective Wikidata entities, to create a corpus of 1.4

million low-context sentences with annotated entities. Part of them form the train and dev dataset. By templating questions in MS-MARCO QnA dataset and 10 million Bing user queries from the ORCAS dataset, and slotting with random entities based on frequency, 17, 868 questions and 471, 746 queries are generated to form the test set together with the rest of Low-context sentences.

Several data processing and augmenting is conducted before training to cater for the applied model. Since document-level embeddings have been proved to be effective for performance improvement in NLP tasks, we adopt Yamada et al. (2020)'s method and extract features by sending the adjacent sentences in corpus together as a document to pretrained models. The number of sentences that each document contains is defined by grid search and eventually set to 30. Although the adjacent sentences are not sentimental related as a document, the document level representation still enriches the context and greatly resolves the low-context situation.

Another novelty of the semeval dataset is the unbalance between train and test set, in terms of quantity and content. To solve the quantity unbalancing, we manage to augment the train set by slotting the entities in sentences and altering with another one from the same type. In case of overfitting, we eventually augment the train set to 3 times of its original size.

4 Methodology

After dealing with data, a common approach for better results is to fine-tune the transformer-based embeddings first, where sentences are sent to the model which is connected to a linear layer for tag prediction. Different language embeddings are selected for different tracks of the task. For English models, we fine-tune BERT-base, BERT-large, M-BERT, XLNET (Yang et al., 2019), Roberta (Liu et al., 2019), XLM-R separately and concatenate these embeddings with basic settings of other embeddings like ELMo, Flair and fastText (Bojanowski et al., 2017), for final training. For Dutch, Spanish and German models, we fine-tine M-BERT, XLM-R and BERT for each language respectively. The parameters for the fine-tuning process are 10 max epochs with batch size of 1 and learning rate of 5.0×10^{-06}.

The concatenated embeddings are used as inputs for a sequence tagging model with BiLSTM layers and CRF layer. The accuracy of the model is used as the reward for reinforcement learning to train the controller, which uses the policy gradient method to maximize the expected reward. We refer to Wang's search space algorithm for the design of reward function and gradient update. The parameters for the training process are 25 maximum episodes, 70 maximum epochs with batch size of 32 and learning rate of 5.0×10^{-06}.

	Dutch	Spanish	German
XLM-R+Fine-tune	0.8985	0.8502	0.8983
Final+Fine-tune	0.9030	0.8612	0.9106

Table 1: Dev Results

5 Results

Due to limitations in time and calculation resources, we only present the results on non-English language models. Table 1 shows the comparison between fine-tuning marco-F1 scores and final marco-F1 scores on dev set for each language in our experiment. It clearly shows that the automatic concatenation method is effective in improving the performance of fine-tuned embeddings. We can also find that XLM-R model plays a vital important role in the concatenated embeddings for non-English models.

Table 2 shows detailed final results on the test set for each language. All scores are calculated as F1 scores. As mentioned in the data section, apart from the performance on low-context NER tasks, the dataset also focuses on the zero-shot learning ability of models on QA and search query scenarios. It can be seen that with fine-tune and reinforcement learning training on low-context corpus, the model achieves high performance on the responsive part in test set, but fails to maintain the performance when making predictions on corpus from other sources.

6 Conclusion

We focus on the performance of automatic concatenated embedding model on semeval complex NER task, and draw the conclusion that with proper data processing methods, the model can learn excellent sequence tagging ability from low-context corpus and achieve outstanding performance on corresponsive part in test set, but cannot transfer such ability to QA and search query domains in test set. Meanwhile, document-level feature extraction and data augmenting by slotting and altering entities are

		Dutch			Spanish			German	
	LOWNER	MSQ-NER	ORCAS-NER	LOWNER	MSQ-NER	ORCAS-NER	LOWNER	MSQ-NER	ORCAS-NER
LOC	0.9222	0.7061	0.4618	0.8519	0.6700	0.4608	0.8230	0.6797	0.4051
PER	0.9432	0.8429	0.6993	0.9315	0.8210	0.6195	0.8837	0.8000	0.6663
PROD	0.7930	0.5456	0.6178	0.7170	0.4327	0.5721	0.7311	0.4307	0.5223
GRP	0.8716	0.3515	0.4148	0.8175	0.3771	0.4662	0.8085	0.4308	0.4462
CORP	0.8791	0.5062	0.5121	0.8567	0.4811	0.5008	0.7631	0.4937	0.4592
CW	0.8176	0.6466	0.4601	0.7578	0.5273	0.4084	0.7603	0.5122	0.4155
macro-F1	0.8711	0.5998	0.5276	0.8221	0.5515	0.5149	0.7950	0.5579	0.4858
overall macro-F1		0.7205			0.6966			0.6675	

Table 2: Cross-language Test Results

proved to be reproductably effective for common NER tasks.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. Ms marco: A human generated machine reading comprehension dataset.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 18 million clicked query-document pairs for analyzing search.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

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

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

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

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

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Automated Concatenation of Embeddings for Structured Prediction. In *the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (**ACL-IJCNLP 2021**)*. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

UC3M-PUCPR at SemEval-2022 Task 11: An Ensemble Method of Transformer-based Models for Complex Named Entity Recognition

Elisa Terumi Rubel Schneider[1], **Renzo M. Rivera-Zavala**[2], **Paloma Martinez**[2], **Claudia Moro**[1] and **Emerson Cabrera Paraiso**[1]

[1]Graduate Program on Informatics, Pontifícia Universidade Católica do Paraná, Brazil
paraiso@ppgia.pucpr.br

[2]Computer Science and Engineering Department, Universidad Carlos III de Madrid, Spain
pmf@inf.uc3m.es

Abstract

This study introduces the system submitted to the SemEval 2022 Task 11: MultiCoNER (Multilingual Complex Named Entity Recognition) by the UC3M-PUCPR team. We proposed an ensemble of transformer-based models for entity recognition in cross-domain texts. Our deep learning method benefits from the transformer architecture, which adopts the attention mechanism to handle the long-range dependencies of the input text. Also, the ensemble approach for named entity recognition (NER) improved the results over baselines based on individual models on two of the three tracks we participated in. The ensemble model for the code-mixed task achieves an overall performance of 76.36% F1-score, a 2.85 percentage point increase upon our individually best model for this task, XLM-RoBERTa-large (73.51%), outperforming the baseline provided for the shared task by 18.26 points. Our preliminary results suggest that contextualized language models ensembles can, even if modestly, improve the results in extracting information from unstructured data.

1 Introduction

Named Entity Recognition (NER) is a Natural Language Processing (NLP) technique to extract relevant information from unstructured natural language data, identifying and categorizing entities in texts and thereby supporting other NLP tasks.

However, processing complex and ambiguous entities is a challenging task that has not received sufficient attention from the research community (Meng et al., 2021). These complex entities can be formed by any linguistic constituent (long noun phrases and sometimes complete sentences), as titles of creative works such as books and movies, different from traditional entities like person names and locations. As the creative work titles can be semantically ambiguous, the challenge is recognizing entities based mainly on their context.

The SemEval 2022 Task 11: MultiCoNER (Multilingual Complex Named Entity Recognition) (Malmasi et al., 2022b) targets the recognition of lowercase complex entities in multilingual and multi-domain texts, encouraging researchers to develop new approaches to extract diversified entity types. The challenge involves the use of human language modeling in the NER task, on cross-domain texts. The semantic structure has six types of entities: person, location, group, corporation, product and creative work. The dataset was provided for 11 languages, and, in addition, this task also provided multi-language and code-mixed [1] features, encouraging the development of more generic and adaptive systems.

As the contextualized pre-trained language models based on the transformer architecture have reached the state-of-the-art in several NLP tasks (Vaswani et al., 2017), in our method, we explore transformer-based models and combined the results into an ensemble, which can make better predictions and have a superior performance than any single contributing model (as demonstrated by Copara et al. (2020), Knafou et al. (2020) and Hernandez et al. (2021)). The models were fine-tuned to NER task on the English, Spanish and code-mixed datasets.

The paper is organised as follows: Section 2 provides a brief explanation of related works, Section 3 provides details about the data used for training our models, Section 4 describes the proposed method with implementation details, Section 5 presents our results with some discussions, and in Section 6 we present the conclusions obtained from the observed results.

2 Related Works

Context-dependent representations models, pre-trained on large-scale unstructured data, particu-

[1]With entities in a different language than the rest of the query, as explained in (Fetahu et al., 2021).

larly those supported by the transformer architecture (Vaswani et al., 2017), have been reached the state-of-the-art performance in NLP problems, including NER task.

These contextual word embedding models use the learned representations over the large data and, in a process called fine-tuning, have their last layers updated to adapt for a downstream task, using task-specific training data.

The Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) is a language representation model designed to pre-train deep bidirectional representations from unlabeled text conditioning on both left and right context in all layers. A Transformer is a encoder-decoder architecture, but in BERT-based models only the encoder is used, with the attention mechanism that learns contextual relationships between words in a text. The BERT-based models are pre-trained using two language generic tasks: masked language modeling (MLM) and next-sentence-prediction (NSP) tasks.

RoBERTa was built on BERT's language masking strategy, where the system learns to predict hidden sections of text within otherwise unannotated language examples. RoBERTa modifies key hyperparameters in BERT, as removing next-sentence pretraining objective. Also, it was trained with much larger mini-batches and learning rates, improving the masked language modeling objective and leading to better downstream task performance (Conneau et al., 2020).

DistilBERT learns a distilled version of BERT, maintaining almost all performance but using only half the number of parameters with a technique called distillation, which approximates the large neural network by a smaller one (Sanh et al., 2019). We also chose DistilBERT on our ensemble because, although it has 40% less parameters than BERT-base and runs 60% faster, can preserve over 95% of BERT's performances (Sanh et al., 2019).

XLM-RoBERTa is a transformer-based masked language model trained on one hundred languages, which outperformed multilingual BERT on a variety of cross-lingual benchmarks (Conneau et al., 2020).

Some researches have focused on the combination of transformer-based models into an ensemble, for NLP tasks. In the work of Copara et al. (2020), the ensemble of contextualized language models resulted in an effective approach for NER in chemical patent documents, outperforming individual transformer models. Knafou et al. (2020) achieved high results with ensemble approach for NER and their study indicates that the more models were used in the ensemble, the more the performances tend to be high and stable. The research of Hernandez et al. (2021) shows that the combination of three BERT-based models obtained superior results than the individual models, in the classification of texts from social media.

Motivated by the success of transformer-based models along with the ensemble approach, we trained several models and combined the results into an ensemble.

3 Data

This shared task encourages NLP researchers to develop complex NER systems for 11 languages, with semantically ambiguous and complex entities in short and low-context settings. Complex entities as creative works are really challenging as they are harder to recognize. An example of entities for English and Spanish can be seen in Figure 1.

According to the organizers (Malmasi et al., 2022a), 15,300 sentences were made available for training and 800 sentences for evaluation in the mono-lingual tracks. The training file has 15,000 sentences for code-mixed and the evaluation dataset has 500 sentences. Test files have varying sizes, with 219,652 sentences for English, 272,887 sentences for Spanish, and 100,000 for code-mixed tracks.

Data enrichment: To improve the performance of our models, we enriched our datasets with other NER annotated corpus containing similar entities. For English, we concatenated to the original MultiCoNER dataset the corpus CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), emerging entities (Derczynski et al., 2017), and In Media Res (Braşoveanu et al., 2020). CoNLL-2003 is a NER dataset released in CoNLL-2003 shared task for

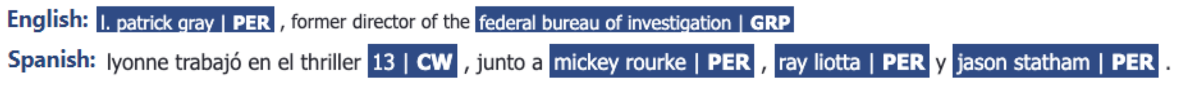


Figure 1: Example of data for English and Spanish (Malmasi et al., 2022a).

language-independent NER. We used the English data (Tjong Kim Sang and De Meulder, 2003), provided from Reuters Corpus containing entities of location, person, organization, and miscellaneous. We converted this data into MultiCoNER format, transforming organization into corporation. The emerging entities (Derczynski et al., 2017) is a corpus released by WNUT2017 Shared Task with focus on rare and emerging entity recognition from noisy user-generated data. This corpus contains the entities person, location, corporation, product, creative-work, and group, the same entity types we have on MultiCoNER. In Media Res is a corpus for evaluating named entity linking with creative works (Braşoveanu et al., 2020), annotated from Wikipedia articles with entities like person, organization, location, (creative) work, event and other. We converted it to MultiCoNER format, excluding events and others and converting work to creative work.

For Spanish, we enriched the dataset with CoNLL-2002 (Tjong Kim Sang, 2002) and Wikiner (Nothman et al., 2012). CoNLL-2002 contains entities of type persons, organizations, locations, times and quantities. We use the Spanish texts, converting organization to corporation, and eliminating entities we do not use. Wikiner (Nothman et al., 2012) is an annotated dataset of Wikipedia texts for 9 languages. It has entities of person, location, organization, and other (misc). We used the texts in Spanish, and we did the conversion from organization to corporation and an analysis of entities of type misc, where could be categorized as products, creative works and others (ignored). Because of this analysis of misc-type entities, we only processed 10% of the corpus.

All the datasets were converted to lowercase format, and concatenated to the original MultiCoNER train dataset.

4 System Description

Our method consists on an ensemble of transformer-based models, trained for NER task and joined by the soft voting method (Figure 2). From the 11 languages, we decided to participate in the English and Spanish tracks, since we already have some related research on them (Rivera and Martinez (2021) and Akhtyamova et al. (2020)), and also in the multilingual and code-mixed tracks to assess our strategy on multilingual texts.

In this section, we detail the development of NER models with its parameters and training setup, the ensemble method, and metrics used in this shared task.

4.1 NER models

We use the transformer-based pre-trained models as checkpoint for fine-tuning on the NER task, with the enriched MultiCoNER datasets. In the fine-tuning step, the existing models were specialized for NER task, where a fully connected layer was added on top of the hidden states of each token to classify tokens according to the named-entities classes. For each track we

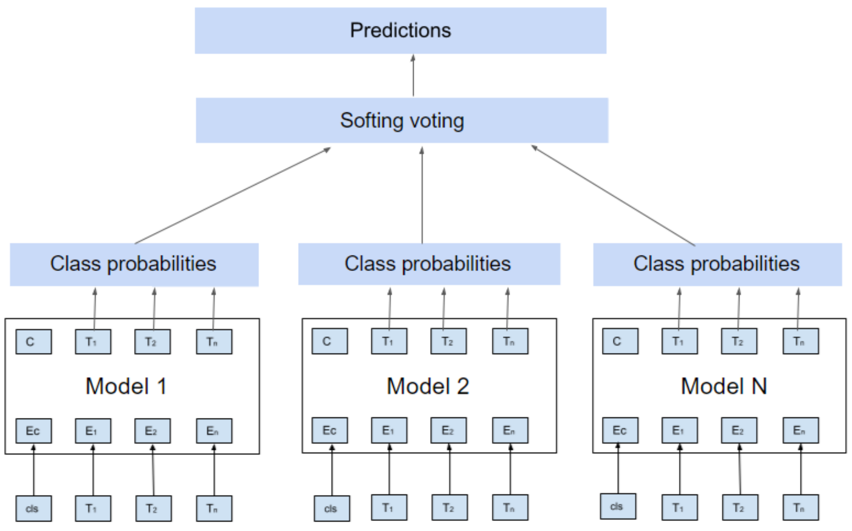


Figure 2: Architecture of our ensemble approach.

Pretrained Models	**#Params**	**Track**
Beto-uncased (Cañete et al., 2020)	110M	Spanish
BERT-base (Devlin et al., 2019)	110M	English
BERT-large (Devlin et al., 2019)	340M	English
BERT-base-multilingual(Devlin et al., 2019)	110M	English, Spanish, code-mixed
DistilBERT-base (Sanh et al., 2019)	66M	English, Spanish, code-mixed
Electra-discriminator (Clark et al., 2020)	110M	English
RoBERTa-base (Liu et al., 2019)	110M	English
RoBERTa-large (Liu et al., 2019)	340M	English
RoBERTa-base-bne (Gutiérrez-Fandiño et al., 2022)	125M	Spanish
RoBERTa-large-bne (Gutiérrez-Fandiño et al., 2022)	355M	Spanish
XLM-RoBERTa-base (Conneau et al., 2020)	270M	English, Spanish, code-mixed

Table 1: Pretrained models features.

participated, we performed the fine-tuning for NER on the pre-trained models. Table 1 shows all the models used in the ensembles and their number of parameters.

English track: The following pre-trained models, trained on a large corpus of English text, were fine-tuned on English train dataset: BERT-base-uncased (Devlin et al., 2019), BERT-large-uncased (Devlin et al., 2019), DistilBERT-base-uncased (Sanh et al., 2019), RoBERTa-base (Liu et al., 2019), RoBERTa-large (Liu et al., 2019), Electra-discriminator (Clark et al., 2020), in addition to the multilingual BERT-base-multilingual-uncased (Devlin et al., 2019) and XLM-RoBERTa-base (Conneau et al., 2020).

Spanish track: We fine-tuned on Spanish train dataset the following pre-trained models, trained on a large corpus of Spanish text: Beto uncased (Cañete et al., 2020), RoBERTa-base-bne (Gutiérrez-Fandiño et al., 2022), RoBERTa-large-bne (Gutiérrez-Fandiño et al., 2022), in addition to the multilingual BERT-base-multilingual-uncased (Devlin et al., 2019), XLM-RoBERTa-base (Conneau et al., 2020) and DistilBERT (Sanh et al., 2019).

For English and Spanish, we also pre-trained and fine-tuned a Megatron model (Shoeybi et al., 2019), however, it was not possible to incorporate it in the ensemble on time, and therefore it was separately evaluated as an unofficial result.

Multilingual and code-mixed track: We first fine-tuned models on the multilingual dataset provided by the shared task and then fine-tuned again on the code-mixed dataset, as the code-mixed training dataset is too small and training just with it could have lower performance. We fine-tuned these multilingual models[2]: BERT-base-multilingual-uncased (Devlin et al., 2019), XLM-RoBERTa (Conneau et al., 2020), and DistilBERT (Sanh et al., 2019). Due to time constraints, although we had trained models for the multilingual track, we could not run the predictions on the test dataset on time to submit to the multilingual track (since it was large files), so our results on multilingual track are unofficial.

All language models were fine-tuned on the training dataset explained in the previous section. We fine-tuned for 10 epochs, with a sequence length of a maximum of 128 tokens, a learning rate of 4e-5, warmup proportion of 0.06, and Adam optimizer for the deep learning training. We used the Pytorch implementation of Hugging Face library (Wolf et al., 2020).

Evaluation metrics: The metric returned by the shared task submission system is F1 macro average performance. For a better analysis and discussion, the performance is also examined per class label.

4.2 Ensemble method

Our ensemble method is based on a soft voting strategy (Schwenker, 2013), where each model returns its predicted probabilities and the class label is obtained by an argmax of the sum of all probabilities (in contrast to hard voting, where the system uses predicted class labels for majority rule voting).

In other words, for a given document, all models infer their predictions independently for each entity and return the probabilities of each label, i.e., their logits[3]. For each token, we sum the logits of all models of the ensemble for this given token and

[2]Which attends all languages of the multilingual and code-mixed tracks.

[3]Logits are the model outputs before application of an activation function (e.g. softmax).

then apply an argmax function, to assign a label to this passage.

The final label output for a token *n* is given by:

$$y_n(x_n) = argmax(\sum_{i=1}^{T} w_i h_i(x_n)) \qquad (1)$$

where *T* is the number of models participating in the ensemble, $\mathrm{w_i}$ is the weight of a model *i*, and $\mathrm{h_i(x_n)}$ is a list with the probabilities for all classes of the model *i* for the token *n*.

For each track, we performed three ensemble experiments.

Experiment 1: As explained above, the token label is an argmax of the sum of predicted probabilities of all models. In this experiment, all models have the same weight (w).

Experiment 2: Each model receives a score ranging from 0.1 to 1, which indicates its efficiency in the evaluation dataset (best overall F1-score). To identify the label of each token, we sum the output of all models, i.e., the probabilities of the token belonging to each class, in a weighted way. In other words, the predicted class probabilities for each classifier are multiplied by the classifier weight, summed across all models, and averaged. Thus, the final class label is then derived from the higher average probability (argmax).

Experiment 3: As in experiment 2, we also calculate weights for each model according to its effectiveness in the evaluation set, but on this experiment, we also calculate weights for each class as well (entity type). The model that performed better for that specific class has the highest weight, also values between 0.1 and 1.

5 Results and Discussion

Official results: The share-task proceedings reports the results (macro F1-score) by the participating systems in the SemEval 2022 Task 11 (Malmasi et al., 2022b), where our results are shown as UC3M-PUCPR team. These results, as well as the analysis performed in the discussion section, are exclusively computed on the test set. Our results refer to experiment 2 (where we used weights for each model), which performed better for all tracks. For the English track, our approach was ranked in 21th place from 30 participants, with a F-Score of 69.24, outperforming the baseline provided for

Named entity	English	Spanish	(*) Multilingual	Code-mixed
Location	0.6821	0.5807	0.7312	0.7925
Person	0.8470	0.7522	0.8017	0.8631
Product	0.6695	0.5102	0.6266	0.7692
Group	0.6953	0.5287	0.5945	0.7051
Creative work	0.6171	0.4350	0.6020	0.6937
Corporation	0.6437	0.6003	0.6741	0.7570
All	0.6924	0.5679	0.6641	0.7636

Table 2: Our F1 results for each class. (*) Multilingual results are unofficial.

the shared task by 8.04 points. On the other hand, for the Spanish track, our system had a significant drop, ranking in the last position (18th from 18 teams), with a F1-Score of 56.79, i.e., 33.15 points behind first place and 0.6 points behind baseline. Our Megatron-trained model performed better than the ensemble in this track, with 60.45 of F1 (unofficial result). In the code-mixed track, our approach ranked in 12th place from 22 participants, with an F1-Score of 76.36, outperforming the baseline by 18.26. For the multilingual track, we achieved 66.41 of F1, 12.31 points higher than baseline, but this result are not official since it was obtained after the competition deadline. Table 2 shows the official F1-scores of our submissions separated by classes, which allows the performance analysis of each entity individually, for all tracks. In table 3, we can assess the performance of each model separately, per class, for English track.

Discussion: The ensemble method modestly improved the F1 performance in relation to the use of individual models for English (1.92 of ensemble improve in relation to the best individual model, BERT-large) and code-mixed (2.85 of ensemble improve in relation to the best individual model, XLM-RoBERTa-large). However, for Spanish track, no improvement was seen in the ensemble compared to Beto individually.

For English track, where we work with eight different pre-trained models, the ensemble model outperforms the other models for all classes, as can be seen in table 3, where the F1 seem relatively stable across all the pre-trained models.

To help understand the reason for the low results in Spanish, we present a confusion matrix, in figure 3. In this track, an analysis of the errors indicates that the model had difficulty recognizing the entities products, groups and creative works. By labeling them with "O" instead of the correct class, lowers the recall value for this entities.

When analyzing the performance of the entities

Language Models	Location	Person	Product	Group	Creative work	Corporation	F1
BERT-base	0.6703	0.8268	0.6283	0.6602	0.5777	0.6018	0.6608
BERT-large	0.6706	0.8309	0.6412	0.6766	0.5971	0.6228	0.6732
BERT-base-multilingual	0.6703	0.8268	0.6283	0.6602	0.5777	0.6018	0.6608
DistilBERT-base	0.6531	0.8189	0.5996	0.6445	0.5476	0.5869	0.6418
Electra-discriminator	0.6550	0.8323	0.6435	0.6570	0.5958	0.6130	0.6661
(*) Megatron	0.5860	0.7302	0.5242	0.5535	0.4374	0.5424	0.5623
RoBERTa-base	0.6203	0.7669	0.5565	0.6047	0.5256	0.5334	0.6012
XLM-RoBERTa-base	0.6623	0.7868	0.5930	0.6072	0.5355	0.5697	0.6258
Ensemble	**0.6821**	**0.847**	**0.6695**	**0.6953**	**0.6171**	**0.6437**	**0.6924**

Table 3: F1-score by model on the English track test set. (*) Megatron model was not part of the ensemble.

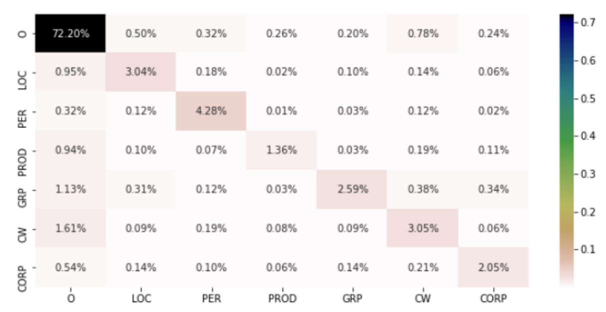


Figure 3: Confusion matrix on Spanish test data.

separately, we noticed that the creative work class had the worst F1 value for English, Spanish and code-mixed tracks, while the person class had the best performance. This is a sign that even in lower case, the trained models could still identify a person's name, i.e., they improved their ability to classify the token based on its context, which is a plus point for transformer-based contextualized models. The creative work entity was already expected to be the most challenging, since movie and book names are not simple noun phrases and are harder to recognize. As noted by Meng et al. (2021), the use of well-formed text with "easy" entities perform better than unseen entities or noisy text. Figure 4 shows how different models detected person and creative work entities, from English evaluation dataset, where we can see correct and incorrect detection in relation to creative work entity. Although some individual models incorrectly predicted the creative work entity, the ensemble achieved the correct answer.

Furthermore, we believe that the prediction of group, corporation, and product entities caused overlap between them because they can be semantically similar, which led to lower than expected results. The datasets we used to enrich the original MultiCoNER datasets do not have these entities so clearly.

For English and code-mixed tracks, we noticed that the large models performed better than the base models, which was already expected, agreeing with the results of previous research. For English, the best-isolated model was BERT-large-uncased, and for code-mixed, XLM-RoBERTa-large, both for evaluation and testing phases. On the other hand, on Spanish track, the uncased version of Beto model performed better than the new RoBERTa models generated from more than 201 million documents (570 Gb) from the Spanish National Library.

Despite improving performance, only the ensemble strategy may not be enough to have competitive results, when analyzing the results of the other teams, especially for the Spanish task, which had poor results.

The enrichment of the train databases may have helped for some entities, but it was not enough to significantly improve performance either, since our best results were precisely in code-mixed track, in which we did not use this technique. The entity types between the external databases may have minor semantic differences, for example, there is a fine line between group and organization. Maybe a closely analyze and even manual re-annotation on these entities could have improved performance.

Also, given such a large amount of test data (files with more than 200,000 sentences), the use of ensemble with transformer-based models may not be the most suitable for processing on regular computers, without much processing GPU power, due to the need to process this large amount of data with several models. In addition to the prediction process, the weighted sum of probabilities also consumes much memory. Our team could not process the test files for the multilingual track before the deadline, and we could not process the Megatron model on time to put into the ensembles (which could improve the results).

We realize that complex entities still represent a great challenge to the NLP community. In addition, lowercase texts, sentences without punc-

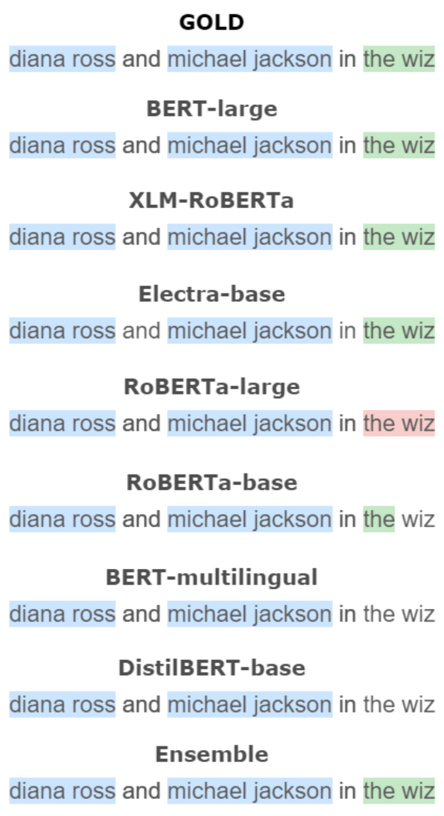

GOLD
diana ross and michael jackson in the wiz
BERT-large
diana ross and michael jackson in the wiz
XLM-RoBERTa
diana ross and michael jackson in the wiz
Electra-base
diana ross and michael jackson in the wiz
RoBERTa-large
diana ross and michael jackson in the wiz
RoBERTa-base
diana ross and michael jackson in the wiz
BERT-multilingual
diana ross and michael jackson in the wiz
DistilBERT-base
diana ross and michael jackson in the wiz
Ensemble
diana ross and michael jackson in the wiz

Figure 4: An example of predictions by different models from English evaluation dataset, where blue represents person entity type, green represents creative works and pink, group type.

tuation or grammatically incomplete, in a cross-domain dataset without any standardization of sentence length, make it even more challenging to extract information from unstructured data, as was the case in this task.

6 Conclusions

We presented our method for recognizing complex entities, which consists in a transformer-based models ensemble. Although our team's results are not among the first, the ensemble method worked for both English and code-mixed tracks, in which we obtained an improvement in the F1 value compared to individual transformer models. Our method outperformed in macro F1 the baseline provided by the organizers in 8.04 points on English track, 18.26 on code-mixed track and 12.31 on multilingual track (unofficial).

As our method is based only on machine learning, without fixed rules, this indicates that the transformer models were able to take advantage of natural language contexts to capture the most relevant features, on lowercase cross-domain texts.

Since extraction of complex entities represents a challenge and these entities are increasingly present, we would like to continue improving our method. Future work intends to train models using a different split (hold out) of the data, training models with more data and for more epochs. Also, since creative works had the worst performance on the three tracks, a hybrid approach that includes a dictionary with creative works can contribute to better results. Furthermore, an analysis of which models should participate in the ensemble can lead to better results and lower processing costs. For example, less robust models like DistilBERT that are faster and lighter, but do not improve performance, possibly do not contribute to the improvement of the results.

Despite having the lowest task evaluation score for the Spanish track, this method exhibited competitive performance at English and code-mixed tracks.

Acknowledgements

The authors thank the shared task organizers for the organization of this event. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Partially supported by Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with UC3M in the line of Excellence of University Professors (EPUC3M17) and PID2020-116527RB-I00 project (ACCESS2MEET).

References

Liliya Akhtyamova, Paloma Martínez, Karin Verspoor, and John Cardiff. 2020. Testing contextualized word embeddings to improve ner in spanish clinical case narratives. *IEEE Access*, 8:164717–164726.

Adrian M. P. Braşoveanu, Albert Weichselbraun, and Lyndon J. B. Nixon. 2020. In media res: A corpus for evaluating named entity linking with creative works. In *CONLL*.

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

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Pre-training transformers as energy-based cloze models. In *EMNLP*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jenny Copara, Nona Naderi, Julien Knafou, Patrick Ruch, and Douglas Teodoro. 2020. Named entity recognition in chemical patents using ensemble of contextual language models. In *Working notes of the CLEF 2020, 22-25 September 2020*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Asier Gutiérrez-Fandiño, Jordi Armengol-Estapé, Marc Pàmies, Joan Llop-Palao, Joaquin Silveira-Ocampo, Casimiro Pio Carrino, Carme Armentano-Oller, Carlos Rodriguez-Penagos, Aitor Gonzalez-Agirre, and Marta Villegas. 2022. MarIA: Spanish language models. *Procesamiento del Lenguaje Natural*, 68(0):39–60.

Luis Alberto Robles Hernandez, Rajath Chikkatur Srinivasa, and Juan M Banda. 2021. A pharmacovigilance application of social media mining: An ensemble approach for automated classification and extraction of drug mentions in tweets. In *NeurIPS 2021 Workshop LatinX in AI*.

Julien Knafou, Nona Naderi, Jenny Copara, Douglas Teodoro, and Patrick Ruch. 2020. BiTeM at WNUT 2020 shared task-1: Named entity recognition over wet lab protocols using an ensemble of contextual language models. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 305–313, Online. Association for Computational Linguistics.

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

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. 2012. Learning multilingual named entity recognition from Wikipedia. In *Artificial Intelligence*, volume 194, pages 151–175. Elsevier.

Renzo Rivera and Paloma Martinez. 2021. Analyzing transfer learning impact in biomedical cross-lingual named entity recognition and normalization. *BMC Bioinformatics*, 22.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC*$^2Workshop$.

Friedhelm Schwenker. 2013. Ensemble methods: Foundations and algorithms [book review]. *Computational Intelligence Magazine, IEEE*, 8:77–79.

Mohammad Shoeybi, Mostofa Ali Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *ArXiv*, abs/1909.08053.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

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

DAMO-NLP at SemEval-2022 Task 11: A Knowledge-based System for Multilingual Named Entity Recognition

Xinyu Wang[◇⋆], Yongliang Shen[♠⋆], Jiong Cai[◇⋆], Tao Wang, Xiaobin Wang[†], Pengjun Xie[†]
Fei Huang[†], Weiming Lu[♠], Yueting Zhuang[♠], Kewei Tu[◇], Wei Lu[‡], Yong Jiang[†*]
[†]DAMO Academy, Alibaba Group
[◇]School of Information Science and Technology, ShanghaiTech University
[♠]College of Computer Science and Technology, Zhejiang University
[‡]StatNLP Research Group, Singapore University of Technology and Design
{wangxy1,caijiong,tukw}@shanghaitech.edu.cn
{syl,luwm}@zju.edu.cn, luwei@sutd.edu.sg
yongjiang.jy@alibaba-inc.com

Abstract

The MultiCoNER shared task aims at detecting semantically ambiguous and complex named entities in short and low-context settings for multiple languages. The lack of contexts makes the recognition of ambiguous named entities challenging. To alleviate this issue, our team **DAMO-NLP** proposes a knowledge-based system, where we build a multilingual knowledge base based on Wikipedia to provide related context information to the named entity recognition (NER) model. Given an input sentence, our system effectively retrieves related contexts from the knowledge base. The original input sentences are then augmented with such context information, allowing significantly better contextualized token representations to be captured. Our system wins 10 out of 13 tracks in the MultiCoNER shared task.[1]

1 Introduction

The MultiCoNER shared task (Malmasi et al., 2022b) aims at building Named Entity Recognition (NER) systems for 11 languages, including English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla. The task has three kinds of tracks including one multilingual track, 11 monolingual tracks and one code-mixed track. The multilingual track requires training multilingual NER models that are able to handle all languages. The monolingual tracks require training individual monolingual models where each model works for only one language. The code-mixed track requires handling code-mixed samples (sentences that may involve multiple languages). The datasets mainly contain sentences from three domains: Wikipedia, web questions and user queries, which are usually short and low-context sentences. Moreover, these short sentences usually contain semantically ambiguous and complex entities, which makes the problem more difficult. In practice, professional annotators usually use their domain knowledge to disambiguate such kinds of entities. They may retrieve the related documents from a knowledge base (KB) or from a search engine to better guide them the annotation of ambiguous named entities (Wang et al., 2019). Therefore, we believe retrieving related knowledge can help the NER model to disambiguate hard samples in the shared task as well. A motivating example is shown in Figure 1, which shows how the retrieval results could help to improve the prediction in practice.

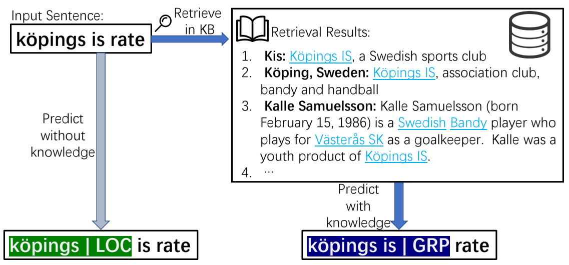

Input Sentence:
köpings is rate
Retrieve in KB
Retrieval Results:
1. **Kis:** Köpings IS, a Swedish sports club
2. **Köping, Sweden:** Köpings IS, association club, bandy and handball
3. **Kalle Samuelsson:** Kalle Samuelsson (born February 15, 1986) is a Swedish Bandy player who plays for Västerås SK as a goalkeeper. Kalle was a youth product of Köpings IS.
4. ...
Predict without knowledge
Predict with knowledge
köpings | LOC is rate
köpings is | GRP rate

Figure 1: A motivating example from the English test set. In the retrieval results, the bold phrases are the title of the retrieved page and the underlined phrases contain the hyperlinks to other pages. LOC and GRP are entity labels representing location and group respectively.

In this paper, we propose a general knowledge-based system for the MultiCoNER shared task. We propose to retrieve the related documents of the input sentence so that the recognition of difficult entities can be significantly eased. Based on Wikipedia of the 11 languages, we build a multilingual KB to search for the related documents of the input sentence. We then feed the input sentence and the related documents into the NER model. Moreover, we propose an iterative retrieval approach to im-

[*]: project lead. [⋆]: equal contributions.
[1]Our code is publicly available at https://github.com/Alibaba-NLP/KB-NER.

prove the retrieval quality. During training, we propose multi-stage fine-tuning. We first train a multilingual model so that the NER model can learn from all annotations. Next, we train the monolingual models (one for each language) and a code-mixed model by using the fine-tuned XLM-RoBERTa (XLM-R) (Conneau et al., 2020) embeddings in the multilingual model as initialization to further boost model performance on monolingual and code-mixed tracks. For each track, we train multiple models with different random seeds and use majority voting to form the final predictions.

Besides the system description, we make the following observations based on our experiments:

1. Knowledge-based systems can significantly improve both in- and out-of-domain performance compared with system without knowledge inputs.
2. Our multi-stage fine-tuning approach can help improve model performance in all the monolingual and code-mixed tracks. The approach can also reduce the training time to speed up our system building at different stages.
3. Our iterative retrieval strategy can further improve the retrieval quality and result in significant improvement on the performance of code-mixed track.
4. Searching over Wikipedia KB performs better than using online search engines on the MultiCoNER datasets.
5. Comparing with other model variants we have tried, our NER model enjoys a good balance between model performance and speed.

2 Related Work

NER (Sundheim, 1995) is a fundamental task in natural language processing. The task has a lot of applications in various domains such as social media (Derczynski et al., 2017), news (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003), E-commerce (Fetahu et al., 2021; Wang et al., 2021b), and medical domains (Doğan et al., 2014; Li et al., 2016). Recently, pretrained contextual embeddings such as BERT (Devlin et al., 2019), XLM-R and LUKE (Yamada et al., 2020) have significantly improved the NER performance. The embeddings are trained on large-scale unlabeled data such as Wikipedia, which can significantly improve the contextual representations of named entities. Recent efforts (Peters et al., 2018; Akbik et al., 2018; Straková et al., 2019) concatenate different kinds of pretrained embeddings to form stronger token representations. Moreover, the embeddings are trained over long documents, which allows the model to easily model long-range dependencies to disambiguate complex named entities in the sentence. Recently, a lot of work shows that utilizing the document-level contexts in the CoNLL NER datasets can significantly improve token representations and achieves state-of-the-art performance (Yu et al., 2020; Luoma and Pyysalo, 2020; Yamada et al., 2020; Wang et al., 2021a). However, the lack of context in the MultiCoNER datasets means the embeddings cannot take advantage of long-range dependencies for entity disambiguation. Recently, Wang et al. (2021b) use Google search to retrieve external contexts of the input sentence and successfully achieve state-of-the-art performance across multiple domains. We adopt this idea so that the embeddings can utilize the related knowledge by taking the advantage of long-range dependencies to form stronger token representations. Comparing with Wang et al. (2021b), we build the local KB based on Wikipedia because the KB matches the in-domain data of the shared task and is fast enough to meet the time requirement in the test phase[2].

Fine-tuning pretrained contextual embeddings is a useful and effective approach to many NLP tasks. Recently, some of the research efforts propose to further train the fine-tuned embeddings with specific training data or in a larger model architecture to improve model performance. Shi and Lee (2021) proposed two-stage fine-tuning, which first trains a general multilingual Enhanced Universal Dependency (Bouma et al., 2021) parser and then fine-tunes on each specific language separately. Wang et al. (2021a) proposed to train models through concatenating fine-tuned embeddings. We extend these ideas as multi-stage fine-tuning, which improves the accuracy of monolingual models that use fine-tuned multilingual embeddings as initialization in training. Moreover, multi-stage fine-tuning can accelerate the training process in system building.

3 Our System

We introduce how our knowledge-based NER system works in this section. Given a sentence of n tokens $\boldsymbol{x} = \{x_1, \cdots, x_n\}$, the sentence is fed into our knowledge retrieval module. The knowledge retrieval module takes the sentence as the query and retrieves top-k related paragraphs in KB.

[2]There are only 7 days for the test phase.

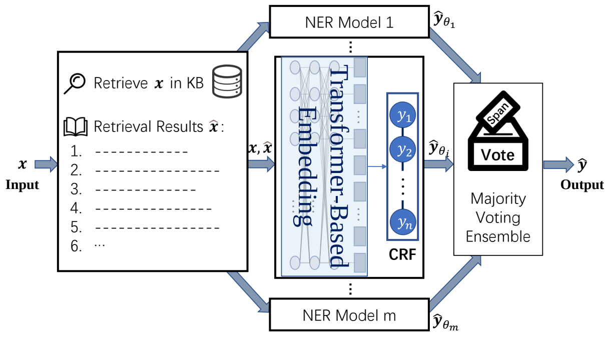


Figure 2: The architecture of our knowledge-based NER system.

The system then concatenates the input sentence and the related paragraphs together and feeds the concatenated sequence into the embeddings. The output token representations of the input sentence are fed into a linear-chain conditional random field (CRF) (Lafferty et al., 2001) layer and the CRF layer produces the label predictions. Given the label predictions of multiple NER models with different random seeds, the ensemble module uses a voting strategy to decide the final predictions $\hat{\boldsymbol{y}} = \{\hat{y}_1, \cdots, \hat{y}_n\}$ of the sentence. The architecture of our framework is shown in Figure 2.

3.1 Knowledge Retrieval Module

Retrieval-augmented context is effective for named entity recognition tasks (Wang et al., 2021b), as external relevant contexts can provide auxiliary information for disambiguating complex named entities. We construct multilingual KBs based on Wikipedia pages of the 11 languages, and then retrieve relevant documents by using the input sentence as a query. These retrieved documents act as contexts and are fed into the NER module. To enhance the retrieval quality, we further designed an iterative retrieval approach, which incorporates predicted entities of NER models into the search query.

Knowledge Base Building Wikipedia is an evolving source of knowledge that can facilitate many NLP tasks (Chen et al., 2017; Verlinden et al., 2021). Wikipedia provides a rich collection of mention hyperlinks (referred to as wiki anchors). For example, in the sentence "*Steve Jobs founded Apple*", entities "*Steve Jobs*" and "*Apple*" are linked to the wiki entries `Steve_Jobs` and `Apple_Inc` respectively. For the NER task, these anchors provide useful clues on where the entities are to the model. Based on Wikipedia we can build local Wikipedia search engines to retrieve the relevant context of the input sentences for each language.

We download the latest (2021-12-20) version of the Wikipedia dump from Wikimedia[3] and convert it to plain texts. Then we use ElasticSearch (ES)[4] to index them. ElasticSearch is document-oriented, and the document is the least searchable unit. We define the document in our local Wikipedia search engines with three fields: sentence, paragraph and title. We create inverted indexes on both the sentence field and the title field. The former is used as a sentence-level full-text retrieval field, while the latter indicates the core entity described by the wiki page and can be used as an entity-level retrieval field. The paragraph field stores the contexts of the sentence. To take advantage of the rich wiki anchors in Wikipedia paragraphs, we marked them with special markers. For example, to incorporate the hyperlinks [*Apple* $\rightarrow$ `Apple Inc`] and [*Steve Jobs* $\rightarrow$ `Steve Jobs`] to the paragraph, we transformed "*Steve Jobs founded Apple*" into "*<e:Steve Jobs>Steve Jobs</e> founded <e:Apple_inc>Apple</e>*"[5].

Sentence Retrieval Retrieval at the sentence level takes the input sentence as a query and retrieves the top-k documents on the sentence field. Given an input sentence, we select the corresponding search engine according to the language of the sentence.

Iterative Entity Retrieval The core of the NER task lies in the entities, while retrieval at the sentence level overlooks the key entities in the sentences. For this reason, we consider the relevance of the entities in the sentence to the title field in the documents during retrieval. We concatenate the entities in the sentences with "|" and then retrieve them on the title field. On the training and development sets, we utilize the ground-truth entities directly. On the test set, we first perform the sentence retrieval and then use the entity mentions[6] predicted by the model for entity retrieval. This bootstrapping manner can be applied for T turns.

Context Processing After top-k results from the KB are retrieved, the system post-processes the retrieved documents into the contexts of the input sentence. There are three options of utilizing

[3]https://dumps.wikimedia.org/
[4]https://www.elastic.co/
[5]<e:XXX>YYY</e>: where XXX is the title of the linked page and YYY is the phrase with hyperlink in the sentence.
[6]Here we define mentions as the named entities ignoring the entity types.

the texts in the documents, which are: 1) use the matched paragraph; 2) use the matched sentence; 3) use the matched sentence but remove the wiki anchors. We compare the performance of each option in section 5.4. In each retrieved document, we concatenate the title and texts together to form the context $\hat{\boldsymbol{x}}_i$. The results are then concatenated into $\{\hat{\boldsymbol{x}}_1, \cdots, \hat{\boldsymbol{x}}_k\}$ based on the retrieval ranking.

3.2 Named Entity Recognition Module

In our system, we use XLM-R large as the embedding for all the tracks. It is a multilingual model and is applicable to all tracks. Given the input sentence $\boldsymbol{x}$ and the retrieved contexts $\{\hat{\boldsymbol{x}}_1, \cdots, \hat{\boldsymbol{x}}_k\}$, we add the separator token (i.e., "</s>" in XLM-R) between them and concatenated them together to form the input $\tilde{\boldsymbol{x}}$ of the NER module. We chunk retrieved texts to avoid the amount of subtoken in the sequence exceeding the maximum subtoken length in XLM-R (i.e., 512 in XLM-R).

Our system regards the NER task as a sequence labeling problem. The embedding layer in the NER module encode the concatenated sequence $\tilde{\boldsymbol{x}}$ and output the corresponding token representations $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n, \cdots\}$. The module then feeds the token representations $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$ of the input sentence into a linear-chain CRF layer to obtain the conditional probability $p_\theta(\boldsymbol{y}|\tilde{\boldsymbol{x}})$:

$$\psi(y', y, \boldsymbol{v}_i) = \exp(\mathbf{W}_y^T \boldsymbol{v}_i + \mathbf{b}_{y',y}) \quad (1)$$

$$p_\theta(\boldsymbol{y}|\tilde{\boldsymbol{x}}) = \frac{\prod_{i=1}^{n} \psi(y_{i-1}, y_i, \boldsymbol{v}_i)}{\sum_{\boldsymbol{y}' \in \mathcal{Y}(\boldsymbol{x})} \prod_{i=1}^{n} \psi(y'_{i-1}, y'_i, \boldsymbol{v}_i)}$$

where θ represents the model parameters and $\mathcal{Y}(\boldsymbol{x})$ denotes the set of all possible label sequences given $\boldsymbol{x}$. In the potential function $\psi(y', y, \boldsymbol{v}_i)$, $\mathbf{W}_y^T \boldsymbol{v}_i$ is the emission score and $\mathbf{b}_{y',y}$ is the transition score, where $\mathbf{W}^T \in \mathbb{R}^{t \times d}$ and $\mathbf{b} \in \mathbb{R}^{t \times t}$ are parameters and the subscripts y' and y are the indices of the matrices. During training, the negative log-likelihood loss $\mathcal{L}_{\text{NLL}}(\theta) = -\log p_\theta(\boldsymbol{y}^*|\tilde{\boldsymbol{x}})$ for the concatenated input sequence with gold labels $\boldsymbol{y}^*$ is used. During inference, the model prediction $\hat{\boldsymbol{y}}_\theta$ is given by Viterbi decoding.

3.3 Ensemble Module

Given predictions $\{\hat{\boldsymbol{y}}_{\theta_1}, \cdots, \hat{\boldsymbol{y}}_{\theta_m}\}$ from m models with different random seeds, we use majority voting to generate the final prediction $\hat{\boldsymbol{y}}$. We convert the label sequences into entity spans to perform majority voting. Following Yamada et al. (2020), the module ranks all spans in the predictions by the number of votes in descending order and selects the spans with more than 50% votes into the final prediction. The spans with more votes are kept if the selected spans have overlaps and the longer spans are kept if the spans have the same votes.

4 Experimental Setup

4.1 Data and Evaluation Methodology

We use the official MultiCoNER dataset (Malmasi et al., 2022a) in all tracks to train our NER models. There are mainly three domains in the dataset: LOWNER (Low-Context Wikipedia NER) contains low-context sentences from Wikipedia; MSQ (MS-MARCO Question NER) is based on MS-MARCO web question corpus (Nguyen et al., 2016) containing a lot of natural language questions; ORCAS (Search Query NER) contains user queries from Microsoft Bing (Craswell et al., 2020). The MSQ and ORCAS samples are taken as out-of-domain data in the shared task. The training and development sets only contain a small collection of samples of these two domains and mainly contain data from the LOWNER domain. The test set, however, contains much more MSQ and ORCAS samples to assess the out-of-domain performance.

The results of the shared task are evaluated with the entity-level macro F1 scores, which treat all the labels equally. In comparison, most of the publicly available NER datasets (e.g., CoNLL 2002, 2003 datasets) are evaluated with the entity-level micro F1 scores, which emphasize common labels.

4.2 Training

NER Model Training Before building the final system, we compare a lot of variants of the system. We train these variant models on the training set for 3 times each with different random seeds and compare the averaged performance of the models. According to the dataset sizes, we train the models for 5 epochs, 10 epochs and 100 epochs for multilingual, monolingual and code-mixed models respectively. Our final NER models are trained on the combined dataset including both the training and development sets on each track to fully utilize the labeled data. For models trained on the training set, we use the best macro F1 on the development set during training to select the best model checkpoint. For models trained on the combined dataset,

System	EN	ES	NL	RU	TR	KO	FA	DE	ZH	HI	BN	MULTI	MIX	AVG.
Ours: Baseline	77.81	76.80	80.51	74.65	72.83	70.81	72.68	81.92	65.56	67.80	65.27	74.19	77.75	73.74
Sliced	74.54	75.11	77.66	73.73	68.77	70.66	68.66	78.90	65.21	67.00	63.05	71.07	72.74	71.32
RACAI	75.78	75.62	78.41	74.60	70.42	71.74	70.42	79.39	62.70	68.08	66.28	72.10	79.37	72.69
USTC-NELSLIP	85.47	85.44	87.67	83.82	85.52	86.36	87.05	89.05	**81.69**	84.64	**84.24**	85.30	**92.90**	86.09
Ours	**91.22**	**89.94**	**90.50**	**91.50**	**88.69**	**88.59**	**89.70**	**90.65**	78.06	**86.23**	83.51	**85.31**	91.79	**88.13**

Table 1: Part of the official results and the post-evaluation results of our baseline system.

we use the final model checkpoint after training[7].

Multi-stage Fine-tuning Besides our final settings, we have a lot of stages of KB settings during our system building. Multi-stage fine-tuning aims at transferring the parameters of fine-tuned embeddings in a model at an early stage into other models in the next stage. The approach stores the checkpoint of fine-tuned XLM-R embeddings at the early stage and uses it as the initialization of XLM-R embeddings for model training at the next stage. One benefit of multi-stage fine-tuning is the monolingual and code-mixed models, can utilized the annotations of all the tracks to further improve the model performance. XLM-R embedding is a multilingual embedding with strong cross-lingual transferability over all 11 languages. Therefore, we use the checkpoint of fine-tuned multilingual model for continue fine-tuning on the monolingual and code-mixed models. Another benefit of multi-stage fine-tuning is that it accelerates the training speed. As the size of the multilingual dataset is relatively large, it is quite time-consuming to train a multilingual model. When we try different types of KB, we can utilize the checkpoints of multilingual models at the previous stage to train the monolingual and code-mixed models with new types of contexts without training new multilingual models. Moreover, we can reduce the training epochs for faster speed since the XLM-R checkpoints have already learned from all the datasets.

Continue Pretraining To make XLM-R learn the data distribution of the shared task, we combine the training and development sets on the monolingual tracks to build a corpus to continue pretrain XLM-R. Specifically, we collocate all sentences according to their languages, then cut the text into chunks of fixed length, and train the model on these text chunks using the Masked Language Modeling objective. We continue pretrain XLM-R for 5 epochs. We use the continue pretrained XLM-R model as the initialization of the multilingual models during training.

[7]Please refer to Appendix A for detailed settings.

5 Results and Analysis

In this section, we use language codes[8] to represent languages, and use MULTI and MIX to represent multilingual and code-mixed tracks respectively[9].

5.1 Main Results

There are 55 teams that participated in the shared task. Due to limited space, we only compare our system with the systems from teams USTC-NELSLIP, RACAI and Sliced[10]. In the post-evaluation phase, we evaluate a baseline system without using the knowledge retrieval module to further show the effectiveness of our knowledge-based system. The official results and the results of our baseline system are shown in Table 1. Our system performs the best on 10 out of 13 tracks and is competitive on the other 3 tracks. Moreover, our system outperforms our baseline by 14.39 F1 on average, which shows the knowledge retrieval module is extremely helpful for disambiguating complex entities leading to significant improvement on model performance.

5.2 How Significant is the Role of Knowledge-based System on Each Domain?

To further show the effectiveness of our knowledge-based system, we show the relative improvements of our system over our baseline system on each domain in Table 2. We observe that in most of the cases, the two out-of-domain test sets have more relative improvements than the in-domain test set. This observation shows that the knowledge from Wikipedia can not only improve the performance of the LOWNER domain which is the same domain as the KB, but also has very strong cross-domain

[8]`https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes`

[9]Please refer to Appendix B for more analysis.

[10]Please refer to `https://multiconer.github.io/results` for more details about the results.

			EN	**ES**	**NL**	**RU**	**TR**	**KO**	**FA**	**DE**	**ZH**	**HI**	**BN**	**AVG.**
In-domain	LOWNER	Baseline	88.70	86.54	89.92	81.52	88.52	86.25	81.85	91.71	85.43	83.13	82.69	86.02
		Ours	96.78	96.19	97.96	96.60	96.43	96.83	96.48	94.89	88.66	84.18	86.31	93.76
		Δ	+8.08	+9.65	+8.04	+15.08	+7.91	+10.58	+14.63	+3.18	+3.23	+1.05	+3.62	+7.74
Out-of-domain	MSQ	Baseline	70.49	71.86	72.63	72.31	75.49	68.57	71.54	74.63	67.38	73.57	58.66	70.65
		Ours	83.50	83.10	83.34	87.03	88.76	81.96	87.36	86.18	79.80	89.20	72.00	83.84
		Δ	+13.01	+11.24	+10.71	+14.72	+13.27	+13.39	+15.82	+11.55	+12.42	+15.63	+13.34	+13.19
	ORCAS	Baseline	62.07	62.71	67.39	64.83	66.92	56.08	65.52	67.52	55.34	62.03	60.68	62.83
		Ours	83.72	81.33	80.29	85.00	85.85	81.06	84.84	84.40	72.11	85.75	82.13	82.41
		Δ	+21.65	+18.62	+12.90	+20.17	+18.93	+24.98	+19.32	+16.88	+16.77	+23.72	+21.45	+19.58

Table 2: Per-domain macro F1 score on the test set of our system and our baseline system for each language. Δ represents the relative improvements of our system over the baseline system.

transferability to other domains such as web questions and user queries. According to the baseline performance over the three domains, the ORCAS domain has the lowest score, which shows the challenges in recognizing named entities in user queries. However, our retrieved documents in KB can significantly ease the challenges in this domain and results in the highest improvement out of the three domains.

5.3 How Relevant Are the Retrieval Results to the Queries?

To evaluate the relevance of the retrieval results to the query, we define a character-level relevance metric, which calculates the Intersection-over-Union (IoU) between the characters of query and result. Assuming that the character sets[11] of query and retrieval result are A and B respectively, then the character-level IoU is $\frac{A \cap B}{A \cup B}$. We calculate the character-level IoU of the sentence and its top-1 retrieval result on all tracks, and plot its distribution on the training, development and test set in Figure 3. We have the following observations: 1) the IoU values are concentrated around 1.0 on the training and development sets of **EN**, **ES**, **NL**, **RU**, **TR**, **KO**, **FA**, which indicates that most of the samples were derived from Wikipedia. Therefore, by retrieving, we can obtain the original documents for these samples. 2) the distribution of data on the test set is consistent with the training and development sets for most languages, except for **TR**. On **TR**, the character-level IoU values of the samples and query results cluster at around 0.5. We hypothesize that this is because the source of the test set for **TR** is different from the training set. However, the model still performs strongly on this language, suggesting that the model can mitigate the difficulties caused by inconsistent data distribution by retrieving the context from Wikipedia.

[11]The sets take repeat characters as different characters.

5.4 How Important Can the Types of KB be?

We compare several types of KBs and contexts during our system building.

Online Search Engine In the early stage, we tried to use the knowledge retrieved from Google Search, which can retrieve related knowledge from a large scale of webs and is believed to be a strong multilingual search engine.

Three Context Types Retrieved from Wikipedia As we mentioned in Section 3.1, there are three context processing options, which are: 1) use the matched paragraph; 2) use the matched sentence; 3) use the matched sentence but remove the wiki anchors. We denote the three options as PARA, SENT and SENT$_{\text{-LINK}}$ respectively.

Entity Retrieval with Gold Entities We use gold entities on the development set to see whether the model performance can be improved. This can be seen as the most ideal scenario for iterative retrieval. We denote this process as ITER$_G$ and use PARA for the context type.

In Table 3, we can observe that: 1) For the three context options, PARA is the best option for **EN**, **ES**, **NL**, **RU**, **TR**, **KO**, **FA**, **MIX** and **MULTI**. SENT$_{\text{-LINK}}$ is the best option for **HI** and **BN**. For **DE** and **ZH**, SENT and SENT$_{\text{-LINK}}$ are competitive. As a result, we choose SENT for the two languages since we believe the wiki anchors from the Wikipedia can help model performance; 2) Comparing with the baseline, the knowledge from Google Search can improve model performance. Based on the best context option of each track, the knowledge from Wikipedia is better than the online search engine; 3) For ITER$_G$, we can find that the context can further

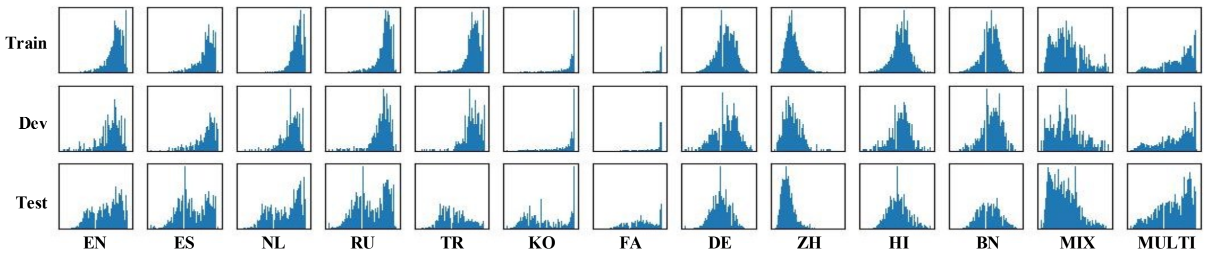


Figure 3: The distribution of the character-level IoU between the query and its top-1 result. Each subplot is a histogram on the corresponding dataset, where the x-axis indicates the IoU values ranging from 0 to 1.

	EN	ES	NL	RU	TR	KO	FA	DE	ZH	HI	BN	MULTI	MIX
Baseline	87.13	85.88	88.87	82.38	86.22	85.98	81.25	91.21	87.65	82.62	82.80	85.78	77.92
Google Search	92.46	88.68	91.58	85.88	89.83	88.95	82.96	93.56	89.16	84.27	84.38	87.84	86.26
Wiki-PARA	**95.82**	94.19	**97.53**	95.53	**97.40**	96.05	**95.93**	92.83	87.10	82.78	83.35	93.51	85.16
Wiki-SENT	87.62	89.33	92.90	79.41	89.00	91.49	95.99	94.42	**89.47**	84.55	84.12	89.34	78.65
Wiki-SENT-LINK	86.83	87.65	91.86	79.15	86.66	86.36	84.37	94.46	89.32	84.78	84.83	87.35	80.07
Wiki-PARA+ITER$_G$	94.89	**94.44**	97.45	**95.59**	96.89	**96.34**	95.83	**94.62**	88.47	**86.43**	**85.85**	**93.60**	**90.52**

Table 3: A comparison of the models utilizing different types of knowledge on the **development set**.

	HI	BN	MIX
Wiki-PARA+ITER$_G$	86.43	85.85	90.52
Wiki-SENT+ITER$_G$	85.69	86.57	91.38
Wiki-SENT-LINK+ITER$_G$	86.15	86.13	91.38
Wiki-Opt$_{\text{Best}}$	**84.78**	**84.83**	85.16
Wiki-Opt$_{\text{Best}}$+ITER$_P$	83.36	84.37	**88.97**

Table 4: Effectiveness of iterative retrieval. Opt$_{\text{Best}}$ represents using the best context option for each language.

	HI	BN	MIX
Wiki-Opt$_{\text{Best}}$	90.02	90.81	96.72
Wiki-Opt$_{\text{Best}}$-Mention	90.76	90.75	96.71

Table 5: A comparison of mention detection F1 score over NER models and mention detection models.

Module	Sentences/Second
Local Knowledge Base Retrieval	64.52
Google Search Retrieval	1.50
NER Module - Training	2.91
NER Module - Prediction	8.13

Table 6: Model speed of the knowledge retrieval module and NER module in our system.

improve the performance over 8 out of 13 tracks. However, there are only significant improvements for **HI**, **BN** and **MIX**.

Iterative Entity Retrieval with Predicted Entities Based on the results in Table 3, we further analyze how the predicted entity mentions can improve the retrieval quality. We denote the iterative entity retrieval with predicted mentions as ITER$_P$. In the experiment, we set $T = 2$.[12] We extract the predicted mentions of the development sets from the models based on the best context option for each track. We conduct the experiments over **HI**, **BN** and **MIX** which have significant improvement with ITER$_G$. In Table 4, we also list the performance of ITER$_G$ for reference, which can be seen as using the predicted mentions with 100% accuracy. From the results, we observe that only **MIX** can be improved.

Since iterative entity retrieval uses predicted mentions as a part of retrieval query, the performance of mention detection directly affects the retrieval quality. To further analyze the observation in Table 4, we evaluate the mention F1 score of the NER models with sentence retrieval. For comparison with mention detection performance of NER models, we additionally train mention detection models by discarding the entity labels during training. From the results in Table 5, we suspect the low mention F1 introduces noises in the knowledge retrieval module for **BN** and **HI**, which lead to the decline of performance as shown in Table 4. Moreover, the mention F1 of mention detection models (second row of Table 5) only outperform that of the NER models (first row of Table 5) in a moderate scale. Therefore, we train the ITER models only for the code-mixed track and use the NER models with sentence retrieval to predict mentions.

[12]Our preliminary experiments show that there is no significant improvement for $T = 3$.

	EN	ES	NL	RU	TR	KO	FA	DE	ZH	HI	BN	MIX	AVG.
XLM-R	92.46	88.68	91.58	85.88	89.83	88.95	82.96	93.56	89.16	84.27	84.38	84.52	88.02
CE	**92.49**	**88.97**	**92.20**	**86.21**	**90.47**	**89.01**	**83.53**	**93.96**	**89.40**	**84.86**	**85.38**	**87.35**	**88.65**

Table 7: A comparison of CE models and XLM-R models. Both kinds of models utilize the knowledge from Google Search. The scores are the averaged macro F1 score on the **development set**.

	EN	ES	NL	RU	TR	KO	FA	DE	ZH	HI	BN	MIX	AVG.
Baseline w/ MF	**87.13**	**85.88**	**88.87**	**82.38**	**86.22**	**85.98**	**81.25**	**91.21**	**87.65**	**82.62**	**82.80**	**77.92**	**84.99**
Baseline w/o MF	85.88	84.28	87.98	81.01	84.61	83.98	79.98	89.54	85.57	79.90	81.18	68.21	82.68

Table 8: A comparison of training the NER models with and without multi-stage fine-tuning (MF) for our baseline system on the **development set**.

	EN	ES	NL	RU	TR	KO	FA	AVG.
XLM-R	95.82	94.19	97.53	95.53	97.40	96.05	95.93	96.07
Ensem	96.56	95.11	97.83	**96.48**	97.57	96.54	96.15	96.61
ACE	**96.69**	**95.80**	**98.22**	96.46	**98.01**	**96.79**	**96.75**	**96.96**

Table 9: A comparison of ACE models, XLM-R models and an ensemble of the XLM-R models on the **development set**.

5.5 Model Efficiency

Table 6 shows the speed of each module in our system. In the table, we also show that the retrieval speed of our local KB is significantly faster than that of Google Search. The bottleneck of the system speed is the NER module rather than the knowledge retrieval module. The main reason for the slow speed of the NER module is that the input length of the knowledge-based system is significantly longer than the original input. Taking the EN test set as an example, there are on average 10 tokens for each input sentence in the original test set while there are 218 tokens for the input of our knowledge-based system. The longer inputs slow down the encoding at XLM-R embeddings.

5.6 Effect of Embedding Concatenation

We compare with some variants of our system that we designed but did not use in the test phase.

CE (Concatenation of Embeddings) CE is one of the usual approaches to NER, which concatenates different kinds of embeddings to improve the token representations. In the early stage of our system building, we compare CE with only using the XLM-R embeddings based on the knowledge retrieved from the Google Search. Results in Table 7 show that CE models are stronger than the models using XLM-R embeddings only in all the cases, which show the effectiveness of CE.

ACE (Automated Concatenation of Embeddings) ACE (Wang et al., 2021a) is an improved version of CE which automatically selects a better concatenation of the embeddings. We use the same embedding types as CE and the knowledge are from our Wikipedia KB. We experiment on EN, ES, NL, RU, TR, KO and FA, which are strong with PARA contexts. In Table 9, we further compare ACE with ensemble XLM-R models. Results show ACE can improve the model performance and even outperform the ensemble models[13].

The results in Table 7 and 9 show the advantage of the embedding concatenation. However, as we have shown in Section 5.5, the prediction speed is quite slow with the single XLM-R embeddings. The CE models further slow down the prediction speed since the models contain more embeddings. The ACE models usually have faster prediction speed than the CE models. However, training the ACE models is quite slow. It takes about four days to train a single ACE model. Moreover, the ACE models cannot use the development set to train the model since they use development score as the reward to select the embedding concatenations. Therefore, due to the time constraints, we did not use these two variants in our submission during the shared task period.

5.7 Effectiveness of Multi-stage Fine-tuning

In Table 8, we show the effectiveness of multi-stage fine-tuning on the development set for our baseline system. The result shows that multi-stage fine-tuning can significantly improve the model performance for all the tracks.

[13]Please refer to Appendix A.3 for detailed settings.

6 Conclusion

In this paper, we describe our knowledge-based system for the MultiCoNER shared task, which wins 10 out of 13 tracks in the shared task. We construct multilingual KBs and retrieve the related documents from KBs to enhance the token representations of input text. We show that the NER models can use the retrieved knowledge to facilitate complex entity prediction, significantly improving both the in-domain and out-of-domain performance. Multi-stage fine-tuning can help the monolingual models learn from the training data of all the languages and improve the model performance and training efficiency. We also show that the system presents a good balance between the model performance and prediction efficiency to meet the time requirement in the test phase. We believe this system can be widely applied to other domains for the task of NER. For future work, we plan to improve the retrieval quality and adopt the system to support other kinds of entity-related tasks.

Acknowledgements

This work was supported by Alibaba Group through Alibaba Innovative Research Program.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

Gosse Bouma, Djamé Seddah, and Daniel Zeman. 2021. From raw text to enhanced Universal Dependencies: The parsing shared task at IWPT 2021. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 146–157, Online. Association for Computational Linguistics.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium. Association for Computational Linguistics.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 20 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2983–2989.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

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

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer enhanced named entity recognition for code-mixed web queries. In *SIGIR '21*, SIGIR '21, New York, NY, USA. Association for Computing Machinery.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and

Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database: The Journal of Biological Databases and Curation*, 2016.

Jouni Luoma and Sampo Pyysalo. 2020. Exploring cross-sentence contexts for named entity recognition with BERT. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 904–914, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

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

Tianze Shi and Lillian Lee. 2021. TGIF: Tree-graph integrated-format parser for enhanced UD with two-stage generic- to individual-language finetuning. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 213–224, Online. Association for Computational Linguistics.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.

Beth M. Sundheim. 1995. Named entity task definition, version 2.1. In *Proceedings of the Sixth Message Understanding Conference*, pages 319–332.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

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

Severine Verlinden, Klim Zaporojets, Johannes Deleu, Thomas Demeester, and Chris Develder. 2021. Injecting knowledge base information into end-to-end joint entity and relation extraction and coreference resolution. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1952–1957, Online. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021a. Automated Concatenation of Embeddings for Structured Prediction. In *the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (**ACL-IJCNLP 2021**)*. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021b. Improving named entity recognition by external context retrieving and cooperative learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1800–1812, Online. Association for Computational Linguistics.

Zihan Wang, Jingbo Shang, Liyuan Liu, Lihao Lu, Jiacheng Liu, and Jiawei Han. 2019. CrossWeigh: Training named entity tagger from imperfect annotations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5154–5163, Hong Kong, China. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

A Detailed Experimental Setup

The detailed statistics of the MultiCoNER dataset are listed in Table 10 and the statistics of our KBs ares shown in Table 11.

A.1 Statistics of Datasets and Knowledge Bases

Track	Train	Dev	Test
English	15,300	800	217,818
Spanish	15,300	800	217,887
Dutch	15,300	800	217,337
Russian	15,300	800	217,501
Turkish	15,300	800	136,935
Korean	15,300	800	178,249
Farsi	15,300	800	165,702
German	15,300	800	217,824
Chinese	15,300	800	151,661
Hindi	15,300	800	141,565
Bangla	15,300	800	133,119
Multilingual	168,300	8,800	471,911
Code-mixed	1,500	500	100,000

Table 10: Statistics of the the MultiCoNER dataset (# of sentences). Note that the training and development sets of the multilingual dataset are a mixture of monolingual training and development sets respectively.

Language	Pages	Paragraphs	ES Docs
English	8,075,229	138,259,937	224,077,884
Spanish	1,813,109	29,767,543	47,248,391
Dutch	2,234,442	18,007,520	29,442,016
Russian	2,437,595	44,536,255	77,903,362
Turkish	728,950	8,196,825	12,685,674
Korean	905,976	11,965,418	16,326,787
Farsi	1,502,301	13,723,218	17,342,825
German	3,147,933	54,315,261	98,386,199
Chinese	1,659,253	20,342,685	14,888,964
Hindi	196,745	1,926,636	3,279,827
Bangla	203,869	2,526,333	4,342,959

Table 11: Detailed statistics on 11 languages.

A.2 System Configurations

For the knowledge retrieval module, we retrieve top-10 related results from the KB. For iterative entity retrieval, we set $T = 2$. In masked language model pretraining, we use a learning rate of 5×10^{-5}. For the NER module, we use a learning rate of 5×10^{-6} for fine-tuning the XLM-R embeddings and use a learning rate of 0.05 to update the parameters in the CRF layer following Wang et al. (2021b). Each NER model built by our system can be trained and evaluated on a single Tesla V100 GPU with 16GB memory. For the ensemble module, we train about 10 models for each track.

A.3 Settings of CE and ACE models

In Section 5.6, we compare our NER model with CE and ACE models. In CE and ACE models, we concatenate monolingual fastText (Bojanowski et al., 2017) word embeddings, monolingual/multilingual Flair embeddings (Akbik et al., 2018), ELMo embeddings (Peters et al., 2018; Che et al., 2018), XLM-R embeddings fine-tuned on the whole training data and XLM-R embeddings fine-tuned on the language data by multi-stage fine-tuning. We only feed the knowledge-based input into XLM-R embeddings and feed the original input into other embeddings because it is hard for the other embeddings (especially for LSTM-based embeddings such as Flair and ELMo) to encode such a long input. We use Bi-LSTM encoder to encode the concatenated embeddings with a hidden state of 1,000 and then feed the output token representations into the CRF layer. Following most of the previous efforts, we use SGD optimizer with a learning rate of 0.01. For ACE, we search the embedding concatenation for 30 episodes.

	DE	ZH	HI	BN	MIX	AVG.
Voting	**94.65**	**89.18**	**85.51**	**85.22**	**86.57**	**88.23**
CRF	94.04	88.96	85.37	85.12	85.33	87.76

Table 12: A comparison of ensemble approaches on the **development set**.

B More Analysis

B.1 Majority Voting Ensemble and CRF Level Ensemble

As we state in Section 3.3, we use majority voting as the ensemble algorithm in our system. We show an experiment about how the voting threshold affect the ensemble model performance during our system building on the development set. We ensemble the models on **DE**, **ZH**, **HI**, **BN**, **MIX** with PARA since these five tracks have relatively lower performance than the other 7 tracks. In Figure 4, we show how the threshold of the majority voting affects the model performance. From the figure, we can see that the best threshold varies over the language. Therefore, we simply choose 0.5 as there is no best threshold value. Moreover, we compare the majority voting ensemble and CRF level ensemble in Table 12. The CRF level ensemble averages the emission and transition scores in the Eq. 1 predicted by the candidate models and uses the Viterbi algorithm to get the prediction. The results show that CRF level ensemble performs inferior to the majority voting ensemble. The possible reason is that training with different random seeds may lead to different emission transition scores at different

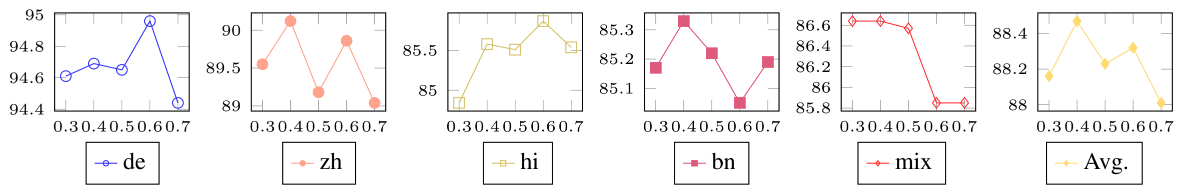


Figure 4: An illustration of majority voting threshold versus the ensemble model performance.

Test Context	PARA		Opt_{Best}	
Search KB	All	Language	All	Language
Wiki-PARA	84.57	84.94	-	-
Wiki+Opt_{Best}	-	84.96	84.38	84.78

Table 13: Test results for multilingual models with different context options and different KB size.

scales. As a result, the models with larger scales have higher weights in the ensemble.

B.2 How the Search Space and the Context Type Affects Multilingual Model Performance?

In the multilingual test set, we can find 304,905 sentences in the other monolingual test sets while there are 167,006 sentences that cannot be found. For these sentences, we can either search on the whole KB of all languages or first detect the language of the input sentence and then search in the specific language KB[14]. Moreover, as we discussed in Section 5.4, using different kinds of retrieved knowledge affects the model performance. As a result, we train two types of multilingual models. One is only using the PARA contexts for all language and another is using the best option for each language based on Table 3. From the results in Table 13, we can observe that: 1) searching over the language specific KB performs better than searching the whole KB, 2) using the language specific context option cannot improve the model performance. Therefore, we ensemble both types of the model for the final submission.

[14]We determine the language of the input sentence using the langdetector (https://pypi.org/project/langdetect/) tool.

Multilinguals at SemEval-2022 Task 11: Complex NER in Semantically Ambiguous Settings for Low Resource Languages

Amit Pandey*, Swayatta Daw*, Narendra Babu Unnam and **Vikram Pudi**
International Institute of Information Technology, Hyderabad, India
{amit.pandey, swayatta.daw}@research.iiit.ac.in
narendra.babu.unnam@research.iiit.ac.in
vikram@iiit.ac.in

Abstract

We leverage pre-trained language models to solve the task of complex NER for two low-resource languages: Chinese and Spanish. We use the technique of Whole Word Masking (WWM) to boost the performance of masked language modeling objective on large and unsupervised corpora. We experiment with multiple neural network architectures, incorporating CRF, BiLSTMs, and Linear Classifiers on top of a fine-tuned BERT layer. All our models outperform the baseline by a significant margin and our best performing model obtains a competitive position on the evaluation leaderboard for the blind test set.

1 Introduction

The Named Entity Recognition (NER) task aims to identify the named entities in an input sequence and categorize them into certain predefined categories or class labels. The task of NER can be further broken down into two subtasks: 1) identification of the entity span and 2) classification of the identified entity span into predefined class labels. For example, in the sentence: *New York City is the most densely populated major city in the United States.*, *New York City* is a named entity of type LOCATION with an entity span of 3 tokens.

The most popular NER task in the English language is CoNLL (Baevski et al., 2019), which is widely used as a benchmark for most NER models. Multiple models have been able to obtain sufficiently high performances in this task setting (Wang et al., 2021a; Zhou and Chen, 2021; Luoma and Pyysalo, 2020; Schweter and Akbik, 2020; Ye et al., 2021; Yamada et al., 2020; Wang et al., 2021b). The CoNLL training set consists of 14,987 train sentences, which comprise 203,621 tokens in total for English data. The entity space consists of 4 different types of entity type labels (locations, persons, organizations, and miscellaneous) to classify each named token. The English data was taken from the Reuters Corpus, which comprises of Reuters News Stories for one year. The training data source, and by extension the labeled named entities, comprises of majorly popular entities found in the general English textual content prevalent in the media. Hence, these entities were easier to classify into the correct classes due to the large prevalence of training data. With the use of pre-trained transformer-based language models, which are already trained on a large unlabelled corpus of English text, this task became even less challenging, as the nature of textual structure in these corpora largely overlap with that of CoNLL.

However, this task becomes challenging in practical settings where a multitude of varieties of named entities is possible. Many of these entities, like Creative Works (CW) and Products (PROD) have complex and ambiguous textual structural content. Such complex named entities rarely appear even in the large training data sets, and the length and structure of the named entities keep changing.

We investigate the task of complex, semantically ambiguous, and low-resource NER (Malmasi et al., 2022b). This task is based on the complex NER, search query and code-mixing NER challenges introduced by Meng et al. (2021) and Fetahu et al. (2021). The shared task of MultiCoNER (stands for multilingual complex NER) adds additional challenge by introducing rarer label types (like Creative Work, Product, etc.).

Another way to increase the difficulty of NER task is to perform it for low-resource languages. There is a significant dearth of both labeled and unlabelled data for such languages. The complexity is further enhanced by using rarer entity types in such languages. Therefore, the scarcity of training data, along with the rarity of entity types, makes it difficult for the models to perform better in the low-resource setting. The shared task of MultiCoNER introduces datasets in multiple low-resource lan-

*Equal Contribution

guages.

We leverage large pre-trained language models trained on low-resource language corpora to obtain competitive performances in the low-resource and complex NER setting. We show that simpler architectures successfully outperform other heavier counterparts. We use standard BERT+CRF-based models to obtain high performances in the evaluation set. We experiment on two low-resource datasets: Spanish and Chinese. The code is available at `https://github.com/AmitPandey-Research/Complex_NER`

We compare multiple architectures on the test and validation set of the shared task. The organisers provide a baseline of XLM-RoBERTa model (Conneau et al., 2019) finetuned on the training dataset of the specified language for the given task. In this paper, we treat the finetuned XLM-RoBERTa as baseline (two separate models are trained for Spanish and Chinese language) and compare the performances against our models. All our models beat the baseline by a significant margin. We describe the prior research work done with respect to both general and low-resource NER tasks in Section 2. We provide the formal task description in Section 3, the dataset details in Section 4, the method and the model architecture in Section 5. We provide details about the experimental implementation in Section 6. We discuss the results obtained and error analysis in Sections 7 and 8 respectively, and finally, we conclude the paper in Section 9.

2 Related Work

The task of low-resource NER has been investigated before by multiple researchers. This line of research focuses mainly on leveraging the cross-lingual contextual information obtained from low-resource languages. (Feng et al., 2018) use cross-lingual knowledge transfer to train the NER model for the low-resource target language. (Xie et al., 2018) use bilingual dictionaries to tackle the task of low-resource NER. (Rahimi et al., 2019b) proposes a Bayesian graphical model approach to improve performance on NER tasks.

NER models often use gazetteers (list of named entities) to improve performance in NER tasks. (Rijhwani et al., 2020) creates soft-gazetteers for low-resource languages, leveraging English Knowledge Bases. (Bari et al., 2019) focuses on an unsupervised approach for NER for to circumvent the label scarcity problem in low-resource languages. (Rahimi et al., 2019a) leverages multilingual transfer learning from multiple languages for low-resource NER tasks. (Hedderich et al., 2021) uses distant supervision in the low-resource setting for NER.

There are multiple approaches that have been undertaken in the recent past to improve the state-of-the-art in NER tasks. (Wang et al., 2020) uses concatenation of embeddings to outperform the state-of-the-art in NER tasks, as they infer that concatenation of embeddings leads to a better word representation. Their method automates the process of finding meaningful embeddings to concatenate for improved performance. (Zhou and Chen, 2021) propose a co-regularization framework for entity extraction comprising of multiple models with different architectures but different parameter initializations. This helps to tackle overfitting of large neural network-based models on low-resource training data for NER. (Schweter and Akbik, 2020) use document-level features to improve information extraction on entity-centric tasks.

NER and Relation Extraction are the core information extraction tasks in NLP. (Ye et al., 2021) models this as a span-pair classification problem, and they further improve the pair representations by considering the dependencies between the spans (pairs) by strategically packing the markers in the encoder. (Yamada et al., 2020) proposes a novel entity-aware self-attention framework for transformer based models for NER. (Wang et al., 2021b) extracts document-level context for sentences for which document information is absent. They treat the sentence as a query and use a search engine to extract the document level contextual information. (Luoma and Pyysalo, 2020) uses multiple neighbouring sentences as the contextual information for NER.

Pre-trained Language Models for NER : Ever since the introduction of BERT (Devlin et al., 2019), transformer based pre-trained language models have effectively utilized transfer learning for downstream NLP tasks. NER has been traditionally modeled as a sequence labeling problem. (Huang et al., 2015) proposed a Bidirectional LSTM with a CRF layer on top for classifying tokens as entities. (Jadhav, 2020) use a pretrained BERT model with a CRF layer on top for performing NER on the DailyHunt news dataset. We use a BERT-based model with a CRF layer on top and achieve competitive performance on low-resource

Label	Description
PER	Person
LOC	Location
GRP	Group
CORP	Corporation
PROD	Product
CW	Creative Work

Table 1: Entity types in the label space

NER tasks on multiple languages, beating the baseline by a significant margin in each case.

3 Task Description

In this task, we attempt complex NER for two low-resource languages: Spanish and Chinese. This task presents additional challenges in the form of test instances consisting of short search queries and low-context sentences. For this task, the systems had to identify the B-I-O format (Ramshaw and Marcus, 1999) (short for beginning, inside, outside) tags for six NER entity type labels: 1) Person, 2) Product, 3) Location, 4) Group, 5) Corporation, and 6) Creative Work. The description of these labels is shown in Table 1.

4 Dataset

The MultiCoNER dataset (Malmasi et al., 2022a) consists of multiple low-resource languages. We consider Chinese and Spanish languages in this paper. For the monolingual track, the participants have to train a model that works for a single language. We fine-tune the language model on the train set to obtain predictions on dev and test set. The labels from the blind test set are not disclosed. The dataset follows a BIO tagging scheme and there are 6 entity types in the label space. The statistics for the Chinese and Spanish datasets in the monolingual track for the train and dev set are provided in Table 2. The total number of test instances for both Spanish and Chinese languages exceed 150,000.

	Train	Dev
# sentences	15300	800

Table 2: Total sentences in Chinese and Spanish monolingual track

5 System Overview

At first, we pre-train the BERT language model on unlabelled corpora for the target low resource language. For Chinese, we use the strategy outlined by (Cui et al., 2020). BERT uses the WordPiece tokenizer (Wu et al., 2016) to split tokens into smaller fragments. It is easier for the masked language model to predict these masked fragments. However, for the Chinese textual texture, the Chinese characters are not formed by alphabet-like symbols, so the WordPiece tokenizer is unable to split the words into small fragments. Hence, we use the Chinese Word Segmentation (CWS) tool to split the text into separate words and then use Whole Word Masking (WWM) strategy for the masked language model objective. In comparison to masking small fragments, this Whole Word Masking strategy makes it harder for the model to predict whole masked words, leading to more robustness.

For the Spanish variant, we adopt the strategy outlined by Cañete et al. (2020). Similar to (Cui et al., 2020), they use the strategy of whole word masking for pre-training BERT language model on unlabelled Spanish corpus.

We adopt the strategy of finetuning these pre-trained BERT models on the downstream NER task for each language.

5.1 BERT+CRF

Conditional random fields (CRFs) are statistical modeling methods used for pattern recognition. They are better suited for tasks such as Part-of-Speech (POS) tagging and NER compared to classifiers based on softmax normalization. Classifiers based on softmax normalization assume the likelihood of the labels to be conditionally independent, and this causes label bias. CRF alleviates this issue of label bias by capturing inter-token dependencies in a graphical model and learning transition scores in addition to the hidden states. In our model, we use linear-chain CRF. In linear CRFs, prediction for each token is dependent only on its immediate neighbors. As shown in equation 2, CRF tries to maximize the ratio of the probability of an optimal sequence of labels to the probability sum of all the possible sequences of labels. Since CRF focuses on the sequence of labels, it can avoid errors like B-PER followed by an I-PROD. Therefore, based on emission scores provided by BERT layer, we calculate the log-likelihood of a sequence of labels. Now we explain the steps involved BERT+CRF

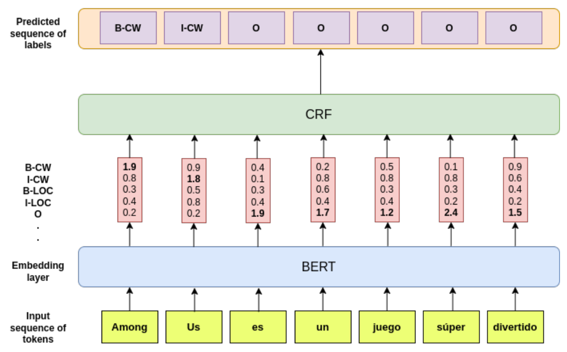


Figure 1: BERT+CRF architecture

architecture that is shown in Figure 1.

Firstly, we obtain token-level dense representations using a fine-tuned BERT-based embedding layer. For an input sequence of tokens $w = (w_1, w_2, w_3, ..., w_n)$, we obtain the ith token representation x_i of dimension d, where d is the dimension of BERT embeddings. The token embedding x_i is passed to a dense linear layer to transform the representation from d to k dimensional space (emission score), where k is the number of labels. We calculate emission scores for all the tokens of the given sequence. We then pass these emission scores to the CRF layer to obtain the probability for a sequence of labels. The emission scores, obtained from the previous layer as $P \in R^{n\times k}$, are passed to the CRF layer whose parameters are $A \in R^{k+2\times k+2}$. Element A_{ij} denotes the transition score from the ith to the jth label. 2 additional states are added to the start and end of the sequence. For a series of tokens $w = (w_1, w_2, w_3, ..., w_n)$ we obtain a series of predictions $y = (y_1, y_2, y_3, ..., y_n)$.

As described in (Lample et al., 2016), the score of the entire sequence is defined as :

$$s(w, y) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i} \qquad (1)$$

The model is trained to maximise the log probability of the correct label sequence:

$$\log(p(y|w)) = s(w, y) - \log(\sum_{\tilde{y}\in Y_w} e^{s(w,\tilde{y})}) \quad (2)$$

where Y_w are all possible label sequences.

5.2 BERT+BiLSTM+CRF

We obtain token-level contextual dense representations (BERT embeddings) using fine-tuned BERT layer. These embeddings are then passed to a BiLSTM layer which further extracts bidirectional information from the given sequence of vectors. The information is encoded in the hidden-state representations of the BiLSTM. We pass these hidden states to the CRF layer to obtain the likelihood of a sequence of labels. We use the pre-trained language model to map the tokens in each input sentence to a dense embedding representation. The BERT-based dense embeddings are passed to the BiLSTM-CRF layer, which is used to obtain the predicted label for each token in the entire sequence. More formally, for a sequence of tokens $w = (w_1, w_2, w_3, ..., w_n)$, we obtain the ith token representation x_i of dimension d, which is the dimension of the dense vector representations of the BERT-based embeddings obtained from the pre-trained language model. The sequence of token embeddings is taken as an input to the BiLSTM in each time step, and the forward hidden states $\overrightarrow{h_f} = (\overrightarrow{h_1}, \overrightarrow{h_2}, \overrightarrow{h_3}, ..., \overrightarrow{h_n})$ and the backward hidden states $\overleftarrow{h_b} = (\overleftarrow{h_1}, \overleftarrow{h_2}, \overleftarrow{h_3}, ..., \overleftarrow{h_n})$ are concatenated to form the combined hidden state representation $h = [\overrightarrow{h_f}, \overleftarrow{h_b}]$. The combined hidden state representation $h \in R^{n\times m}$, where m is the total dimension of BiLSTM, is transformed to a k dimensional space using a linear layer, where k is the total number of labels. Finally, the CRF layer outputs predicted sequence of labels.

5.3 BERT+Linear

This is the simplest architecture based on fine-tuned BERT layer. The input token sequence is mapped to a vector space of d dimension using a pre-trained BERT layer. These embeddings are then passed to a classifier that consists of two Fully Connected (FC) layers followed by a softmax normalization function. The classifier maps the d dimensional BERT embeddings to k dimensions, where k is the number of labels. These k dimensional vectors generated by the fully connected layers are softmaxed to provide a probability distribution across all labels.

6 Implementation Details

We implement all our transformer based models using Pytorch and Huggingface library. The Chinese language model with the Whole Word Masking (WWM) objective is trained on the Chinese Wikipedia unlabelled text corpus. We use the same training corpus of 3 billion unannotated Spanish tokens as Cañete et al. (2020) to pre-train the BERT

	BERT+CRF			BERT+BiLSTM+CRF			BERT+Linear		
Class Label	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
LOC	0.8368	0.8796	**0.8577**	0.8219	0.8759	0.8481	0.8194	0.8613	0.8399
PER	0.9065	0.9028	0.9047	0.9177	0.9028	**0.9102**	0.8933	0.9150	0.9040
PROD	0.6970	0.7468	0.7210	0.7278	0.7468	**0.7372**	0.6864	0.7532	0.7183
GRP	0.7952	0.7857	0.7904	0.7751	0.7798	0.7774	0.8061	0.7917	**0.7988**
CW	0.7965	0.7135	0.7527	0.7654	0.7135	0.7385	0.8107	0.7135	**0.7590**
CORP	0.8657	0.8227	**0.8436**	0.8397	0.7801	0.8088	0.8529	0.8227	0.8375
Average	0.8163	0.8085	**0.8117**	0.8079	0.7998	0.8034	0.8115	0.8096	0.8096

Table 3: Results of our models on validation dataset for Spanish language

	BERT+CRF			BERT+BiLSTM+CRF			BERT+Linear		
Class Label	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
LOC	0.9465	0.9365	**0.9415**	0.9239	0.9312	0.9275	0.9186	0.9259	0.9223
PER	0.8497	0.9225	0.9084	0.8971	0.9457	**0.9208**	0.8955	0.9302	0.9125
PROD	0.8867	0.8285	0.8566	0.8662	0.8504	**0.8582**	0.8593	0.8248	0.8417
GRP	0.7500	0.6923	**0.7200**	0.7727	0.6538	0.7083	0.6923	0.6923	0.6923
CW	0.8265	0.8617	**0.8437**	0.8556	0.8191	0.8370	0.8370	0.8191	0.8280
CORP	0.8615	0.8750	0.8682	0.8808	0.8854	**0.8831**	0.8883	0.8698	0.8789
Average	0.8610	0.8527	**0.8564**	0.8660	0.8476	0.8558	0.8485	0.8437	0.846

Table 4: Results of our models on validation dataset for Chinese language

language model on Spanish data. We implement 3 models: BERT+CRF, BERT+BiLSTM+CRF, and BERT+Linear, for our low resource NER task setting. We run our experiments between 1-100 epochs. We find that the best results are obtained at 10 epochs of training for each model after which the model starts to overfit. We use a dropout from 0.2 to 0.5 for all models. We employ Adam optimizer with default parameters for all experiments. We also experiment with a cyclic learning rate between $1e^{-4}$ to $1e^{-6}$ to avoid getting stuck in local minima. The size of each of the FC layers in the BERT+Linear model is 512. We validate the results of all models using our validation dataset. The hidden layer size of BiLSTM used in the BERT+BiLSTM+CRF model is 256.

7 Results

We compare the performances of all models in the low-resource setting for both Chinese and Spanish languages. From tables 3 and 4 we observe that the BERT+CRF model performs the best across both languages on validation set. We choose the best performing model to evaluate our results on the blind test set. The baseline model chosen by the organisers of this task is XLM-RoBERTa (Conneau et al., 2019)(base model). It is pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages (Conneau et al., 2019). Our approach beats the baseline by a significant margin and outperforms multiple models in the competition. We present the precision, recall and F1 scores for all 3 models in the Tables 3 and 4 for Spanish and Chinese language respectively. We also compare the results between the baseline and our models for the validation dataset in the Tables 5 and 6.

For the Spanish language, we observe that the BERT+CRF (0.8117 F1 score) beats BERT+Linear (0.8096 F1 score) by a slender margin. This can be attributed to the addition of the CRF layer, which exploits inter-token dependencies. The BERT+BiLSTM+CRF model is much heavier with a larger number of parameters and overfits the training dataset due to the smaller number of training instances.

For the Chinese language, we observe that both the BERT+CRF (0.8564 F1 score) and BERT+BiLSTM+CRF (0.8558 F1 score) beat BERT+Linear (0.846 F1 score).

Our models outperform the baseline for both the languages. Our best performing model BERT+CRF beats the baseline F1 score by around 5% for Spanish and by around 10% for Chinese.

The details of the performance of the

	Prec	**Rec**	**F1**
Baseline System	0.764	0.763	0.767
BERT + Linear	0.811	**0.809**	0.809
BERT+BiLSTM+CRF	0.807	0.799	0.803
BERT + CRF	**0.816**	0.808	**0.811**

Table 5: Comparison of models' performances with baseline on validation dataset for Spanish language

	Prec	**Rec**	**F1**
Baseline System	0.758	0.762	0.755
BERT + Linear	0.848	0.843	0.846
BERT+BiLSTM+CRF	**0.866**	0.847	0.855
BERT+CRF	0.861	**0.852**	**0.856**

Table 6: Comparison of models' performances with baseline on validation dataset for Chinese language

	BERT+CRF		
Class Label	Prec	Rec	F1
LOC	0.5768	0.6571	0.6144
PER	0.7641	0.7739	0.7690
PROD	0.6292	0.5141	0.5659
GRP	0.5727	0.5560	0.5642
CW	0.5331	0.5257	0.5294
CORP	0.6605	0.6005	0.6291
Average	0.6227	0.6046	0.6120

Table 7: Performance of Spanish model on test dataset

	BERT+CRF		
Class Label	Prec	Rec	F1
LOC	0.6930	0.7955	0.7407
PER	0.7952	0.6377	0.7078
PROD	0.6853	0.7232	0.7038
GRP	0.7254	0.4608	0.5636
CW	0.5520	0.6798	0.6093
CORP	0.6526	0.7361	0.6918
Average	0.6839	0.6722	0.6695

Table 8: Performance of Chinese model on test dataset

BERT+CRF model in the evaluation phase are shown in Table 7 for the Spanish language and the Table 8 for the Chinese language. We observe a drop in performance on the test dataset compared to the performance on the validation dataset. The model scores 0.6120 F1 for Spanish and 0.6695 F1 for the Chinese language.

8 Error Analysis

We perform error analysis on all 3 different models. We qualitatively analyze the predictions on the validation dataset for both languages. As the final evaluation test set in blind, we are unable to perform analysis on the same. We find that the labels GRP (Group), PROD (Product), and CW (Creative Work) are the most inaccurately predicted labels for the Spanish models. This conforms to our hypothesis that the long-tailed nature of these entities (which means the frequency of occurrence of such entity types in the general literature of the target language is rare). Hence, the model has the most difficulty in recognizing these entities from the contextual sentences. The other label types are more common and were present in the CoNLL dataset as well. We also notice that the BERT+Linear does marginally better than BERT+CRF on predicting such labels (for the Spanish language), despite it not being the best performing model overall. This can be attributed to it being a lighter model, imparting it the capability of generalizing better while training on a relatively lower amount of training instances. BERT+CRF benefits from having CRF along with the lower number of parameters compared to the BERT+BiLSTM+CRF model. This results in it having a better performance compared to both the other models. The drop in the performance of the model on the blind test dataset can be attributed to the model not generalizing well to handle instances of questions and short search queries in the additional test set.

9 Conclusion and Future Work

We have introduced strong improvements over the baseline for the shared task of complex NER for low resource languages. We leverage the Whole Word Masking objective to obtain a better performance in this low-resource setting. We perform extensive experiments and find that simple BERT-CRF based models perform strongly against other heavier models even in such low resource semantically ambiguous setting as evident by the final evaluation rankings. We find this approach to give a higher performance as it is able to utilize the contextual information from a sequence of tokens and learn inter-token dependencies to accurately predict the named entity labels. We also conduct qualitative error analysis and describe our findings. For future work, we aim to leverage these findings to circumvent the label scarcity problem in other low-resource languages and code mixed data.

10 Acknowledgment

We thank the SemEval team, the task organizers, and other participants of this shared task.

References

Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. Cloze-driven pretraining of self-attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5360–5369, Hong Kong, China. Association for Computational Linguistics.

M. Saiful Bari, Shafiq R. Joty, and Prathyusha Jwalapuram. 2019. Zero-resource cross-lingual named entity recognition. *CoRR*, abs/1911.09812.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668, Online. Association for Computational Linguistics.

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

Xiaocheng Feng, Xiachong Feng, Bing Qin, Zhangyin Feng, and Ting Liu. 2018. Improving low resource named entity recognition using cross-lingual knowledge transfer. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 4071–4077. AAAI Press.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Michael A. Hedderich, Lukas Lange, and Dietrich Klakow. 2021. ANEA: distant supervision for low-resource named entity recognition. *CoRR*, abs/2102.13129.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Swapnil Ashok Jadhav. 2020. Detecting potential topics in news using bert, CRF and wikipedia. *CoRR*, abs/2002.11402.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Jouni Luoma and Sampo Pyysalo. 2020. Exploring cross-sentence contexts for named entity recognition with BERT. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 904–914, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019a. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019b. Multilingual ner transfer for low-resource languages. *ArXiv*, abs/1902.00193.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Shruti Rijhwani, Shuyan Zhou, Graham Neubig, and Jaime Carbonell. 2020. Soft gazetteers for low-resource named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8118–8123, Online. Association for Computational Linguistics.

Stefan Schweter and Alan Akbik. 2020. FLERT: document-level features for named entity recognition. *CoRR*, abs/2011.06993.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. Automated concatenation of embeddings for structured prediction. *CoRR*, abs/2010.05006.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021a. Automated concatenation of embeddings for structured prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2643–2660, Online. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021b. Improving named entity recognition by external context retrieving and cooperative learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1800–1812, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379, Brussels, Belgium. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Deming Ye, Yankai Lin, and Maosong Sun. 2021. Pack together: Entity and relation extraction with levitated marker. *CoRR*, abs/2109.06067.

Wenxuan Zhou and Muhao Chen. 2021. Learning from noisy labels for entity-centric information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5381–5392, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

AaltoNLP at SemEval-2022 Task 11: Ensembling Task-adaptive Pretrained Transformers for Multilingual Complex NER

Aapo Pietiläinen   **Shaoxiong Ji**
Aalto University
{aapo.pietilainen; shaoxiong.ji}@aalto.fi

Abstract

This paper presents the system description of team AaltoNLP for SemEval-2022 shared task 11: MultiCoNER. Transformer-based models have produced high scores on standard Named Entity Recognition (NER) tasks. However, accuracy on complex named entities is still low. Complex and ambiguous named entities have been identified as a major error source in NER tasks. The shared task is about multilingual complex named entity recognition. In this paper, we describe an ensemble approach, which increases accuracy across all tested languages. The system ensembles output from multiple same architecture task-adaptive pretrained transformers trained with different random seeds. We notice a large discrepancy between performance on development and test data. Model selection based on limited development data may not yield optimal results on large test data sets.

1 Introduction

SemEval-2022 shared task 11: MultiCoNER (Malmasi et al., 2022b) was about complex Named Entity Recognition (NER) in the multilingual context. Transformer-based models such as BERT (Devlin et al., 2019) have reached high accuracy in standard NER tasks. However, the models still struggle with complex and code-mixed named entities (Meng et al., 2021; Fetahu et al., 2021). The shared task tries to address the challenges regarding complex NER in a multilingual context. Out of the 13 available tracks, we focus on 5 monolingual tracks. We build models for Bangla, English, Farsi, German, and Korean.

Our strategy is to build an ensemble approach to increase accuracy compared to the baseline model. To tackle multilingualism, we build an approach that starts from the same XLM-RoBERTa (Conneau et al., 2019) encoder and after fine-tuning, achieves good performance across languages. We propose two ensembling approaches: (1) naive ensembling and (2) end-to-end (E2E) ensembling. Naive ensembling is a test-time ensembling where class scores from individually trained models are added together. E2E ensembling trains two encoder models jointly by concatenating their results and passing the concatenated predictions to the final layer. In addition to these strategies, we adapt the encoder model to data context using continued pretraining (Gururangan et al., 2020).

We discover that naive ensembling produces good results with few models and by just using different random seeds for training. We also discover that performance can drastically vary between data sets and model selection based on development data may not yield good results on large test sets. Our final rankings are 11th for Farsi, 12th for Bangla, 13th for German, 15th for Korean, and 25th for English. For each language, our ranking is on the lower half of the participants. The code for our submission has been released [1].

2 Background

In NER, complex and ambiguous entities are hard to classify correctly. Out of the 13 available tracks, we participated in the monolingual tracks for Bangla, English, Farsi, German, and Korean. For training, we used only the competition data sets (Malmasi et al., 2022a) provided by the organizers. The task is to detect named entities in given sentences. Figure 1 contains an example of the task setting. This example can be considered complex, as Madagascar is ambiguous as it could refer to the country or the movie.

Transformer-based models (Vaswani et al., 2017) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) produce state-of-the-art results on variety of natural language processing (NLP) tasks including NER. Transformer models trained

[1] Code is available here https://github.com/aapop/multiconer_AaltoNLP

O	O	O	O	B-LOC	O
it	is	known	from	madagascar	.

Figure 1: An example of NER task. A sentence from the development data set and the corresponding labels.

on multilingual data, such as XLM (Conneau and Lample, 2019), can be used for cross-lingual tasks. XLM-RoBERTa (XLM-R) (Conneau et al., 2019) combines the XLM and RoBERTa architectures and produces state-of-the-art results across languages. XLM-R is used as the basis in our system, which we built on top of.

Gururangan et al. (2020) showed that continuing pretraining of a transformer model with the domain or task-specific data often increases the performance. We utilize this finding and apply task-adaptive pretraining on multilingual models to increase performance in a specific language.

Ensemble learning is widely used in machine learning, in which, predictions from several different models are combined. The ensemble learning method combines multiple weak learners and provides a strong learner (Mohri, 2012). This works especially well when the weak learners [2] are diverse and make different errors. Speck and Ngonga Ngomo (2014) showed that ensemble methods can significantly increase NER accuracy. Combining transformer-based model predictions has been shown to increase NER accuracy (Li et al., 2020; Zhao et al., 2021; Souza et al., 2020). To keep applicability on languages with limited number of pretrained models, our approach is to form an ensemble model based on training the same model with different random seeds.

3 System overview

The motivation for our system is to produce an approach that can be applied across languages. Initial testing with the baseline system provided by the organizers suggested that XLM-R$_{\text{large}}$ (Conneau et al., 2019) produces comparable or even better performance than tested monolingual pretrained models for targeted languages. Therefore, we resort to building on top of a multilingual encoder model XLM-R$_{\text{LARGE}}$ and the provided baseline system.

Inspired by results using task-adaptive pretraining (Gururangan et al., 2020), we start by training task-adaptive language models. We hypothesize that task-adaptive pretraining could provide two sources of improvement: (1) adapting to the data domain and (2) improving the capability in the specific language. Task-adaptive pretraining is conducted for the XLM-R$_{\text{LARGE}}$ using the provided training data, which we prepare for pretraining by removing labels and constructing line-by-line sentences. The pretraining is applied to each language separately yielding us five encoder models named *koala/xlm-roberta-large-**XX***. The **XX** is replaced with the corresponding language identifier. The pretraining is continued from the XLM-R$_{\text{large}}$ checkpoint with only the preprocessed task data. The pretraining objective remains the same. The models are available at the Huggingface model repository [3]. In this paper, we refer to the models as koala-XLM-R$_{\text{large}}$ interchangeably for all languages.

After obtaining the enhanced encoder models, we propose two ensemble learning approaches, i.e., naive ensemble and E2E ensemble.

3.1 Naive ensemble

As the first ensemble strategy, we combine predictions from multiple independently trained models at test time. Randomness and choosing a random seed can have a significant effect on the model training and final accuracy. Based on that, our hypothesis was to leverage this randomness to increase the accuracy.

To ensure applicability in languages, where a limited number of available models are available, we train the models using the same architecture and only vary the random seed associated with training. The koala-XLM-R$_{\text{large}}$ is used as the encoder model for each language respectively. After the encoder model, we use the linear layer as the prediction head and apply softmax transformation to obtain class probabilities. The models were trained separately and optimized with AdamW optimizer (Loshchilov and Hutter, 2018) and negative log-likelihood loss function (NLLLoss).

The ensembling is conducted during test time. We simply add the class probabilities together from each of the models,

$$y^* = \operatorname{argmax}_j \sum_n p_{nj}, \qquad (1)$$

where p_{nj} is the probability of model n for the class j. The predicted class y^* is the class with the highest sum of probabilities. For the ensemble weights,

[2]also called base models

[3]https://huggingface.co/koala

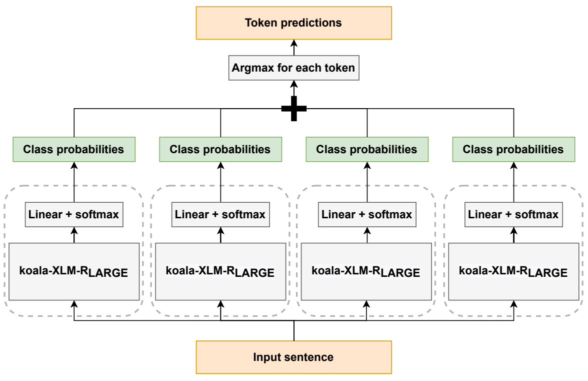


Figure 2: System diagram for the Naive ensemble model. Each of the models are trained separately with different random seeds and ensembling conducted at test time.

we also tried entropy minimized ensemble (Wang et al., 2021), which is based on the intuition that good ensemble weights should decrease entropy (Shannon, 1948; Wang et al., 2020). The algorithm is elaborated in Algorithm 1. This approach did not

Algorithm 1 Entropy minimization for ensembling

```
1: Inputs: scores $\mathbf{p}$, uniform weights $\mathbf{w}^0$
2: for t in $0, 1, \ldots$ steps $= T$ do
3:     Calculate entropy of scores $E \leftarrow \text{Entropy}(\sum_n w_n^t * p_{nj})$
4:     Compute gradient $g^t = \nabla_w E$
5:     Update weights $\mathbf{w}^{t+1} \leftarrow \text{Update}(\mathbf{w}^t, g^t)$
6: end for
7: Output: final prediction $y^* = \sum_n w_n^T * p_{nj}$
```

provide noticeable improvement in our tests and thus, we proceeded with uniform weights.

Our ensemble model consists of four distinct models. The number of four models was chosen as a compromise on the trade-off between accuracy and inference time. The model system is illustrated in Figure 2.

3.2 E2E ensemble

Our second ensemble strategy centered around the idea of jointly training the models. The idea is to include sub-networks (encoder models) in the model, ensemble their predictions, and optimize the loss function jointly. Our system uses koala-XLM-R$_{\text{large}}$ and XLM-R$_{\text{base}}$ as the encoder models.

The input data is passed to both encoder models. After that, we apply linear and softmax layers. For the second encoder model, the linear layer outputs $2 *$ `num classes` scores. We made this decision to add variation between the two sub-networks. The choice to use twice the amount of parameters was arbitrary. We tried a different number of parameters but the performance seemed similar.

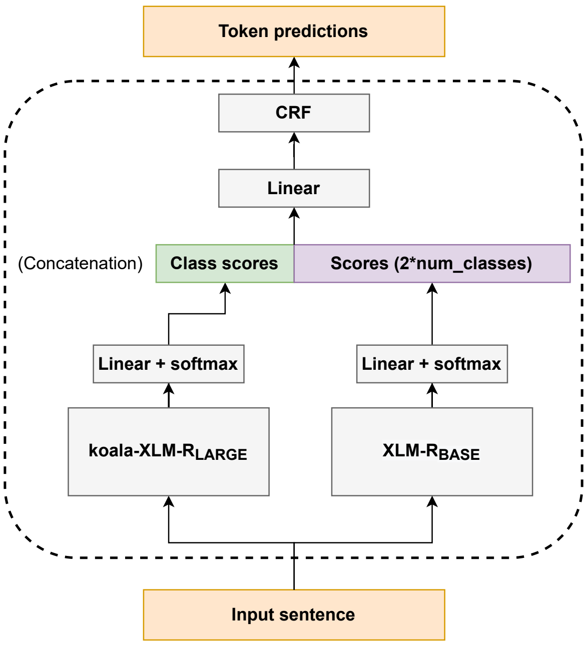


Figure 3: System diagram for the E2E ensemble model.

After outputting the scores from the two sub-networks, the scores are concatenated together. Also, summation was tested when using same-sized outputs, but concatenation seemed to perform better. After that, the concatenated scores are passed to linear and CRF layers. The architecture of E2E ensemble model is illustrated in Figure 3.

The E2E ensemble model consists of two sub-networks. We tried using also three and four sub-networks but adding more models did not improve the performance. Therefore, we settled on using two sub-networks, which also help with the computational burden. We experimented with different sub-network architectures and noticed that using different encoder models provided the best performance despite varying the linear layer size. Also, using koala-XLM-R$_{\text{large}}$ instead of XLM-R$_{\text{base}}$ as the second sub-network did not improve accuracy. Hence, the smaller model was used.

The model was optimized using AdamW optimizer with Viterbi loss as the loss function.

4 Experimental setup

Our experimental setup was based on the provided baseline model (Malmasi et al., 2022b) and the data splits (Malmasi et al., 2022a). Models were trained only using the training data and the best model was selected based on performance on development data. After selecting the best model, we did not do any training with development data. For continued pretraining, we preprocessed the training data. The data were converted from CoNLL format into line-by-line text. Hence, we removed the labels and reconstructed the sentences into a single line of text data.

We selected suitable hyperparameters based on performance on the development data. We selected two learning rates, first for the encoder model and a second one for all subsequent layers (decoder). We set the encoder learning rate to 10^{-6} and the decoder learning rate to 10^{-3}. To prevent overfitting, we used a dropout rate of 0.1. We also used early stopping with development data. The training was stopped when accuracy on development data had not increased in the last three epochs. We selected batch size based on the hardware constraints. For baselines and sub-models in the naive ensemble, we used batch size 64. For E2E ensemble, batch size is 20.

The main Python libraries we used are PyTorch, PyTorch Lightning, AllenNLP, and Transformers. Further details available in the repository [4].

The performance in this shared task is evaluated using prediction accuracy on unseen test data set. The teams are ranked by their macro-averaged F1 classification score.

5 Results

The official metrics were based on the evaluation phase with test data. The participants were ranked by their Macro F1 scores. The performance of built models on test data is shown in Table 1. The best score is bolded and is the model used for ranking. It can be seen that the naive ensemble model performs better than the best individual model across all languages. E2E ensemble model shows mixed results and performance varies run by run. It outperforms naive ensemble on Korean and Farsi test data.

We noticed a rather large discrepancy in accuracies between the development and test data set. The details of model performance on development data are presented in Table 2. Compared to the results on test data, the performance on development data was significantly better. The discrepancies between performance on development and test data suggest that model selection on such limited development data yields sub-optimal results.

Our systems suffer with some of the classes. As can be seen from Figures 4 and 5, PROD and CW classes have significantly lower accuracy than other classes. Performance on GRP, LOC, and PER classes is much higher. Our systems suffer with the no-tag-class O, which has the majority of the misclassifications.

Model	en	ko	fa	de	bn
XLM-R$_{\text{large}}$	0.639	0.589	0.515	0.695	0.465
koala-XLM-R$_{\text{large}}$	0.653	0.590	0.527	0.695	0.476
Naive ensemble	**0.669**	0.610	0.569	**0.714**	**0.518**
E2E ensemble	0.623	**0.618**	**0.589**	0.678	0.511
Final rank	25	15	11	13	12

Table 1: Macro F1 scores and final ranking on test data.

Model	en	ko	fa	de	bn
XLM-R$_{\text{large}}$	0.830	0.802	0.756	0.870	0.763
koala-XLM-R$_{\text{large}}$	0.829	0.805	0.745	0.861	0.758
Naive ensemble	**0.841**	**0.824**	**0.764**	**0.864**	**0.782**
E2E ensemble	0.821	0.811	**0.764**	0.856	0.755

Table 2: Macro F1 scores on development data.

5.1 Task-adaptive pretraining

We hypothesized that task-adaptive pretraining would adapt the multilingual encoder model into the data and language domains. We tested the performance by training the model four times using different random seeds and comparing the average micro F1 score between XLM-R$_{\text{large}}$ and koala-XLM-R$_{\text{large}}$ models. The results of task-adaptive pretraining are elaborated in Table 3. The improvements on this task are small and compared to the original paper, in which, the accuracy increase was considerably larger on some data sets. For German, the accuracy even decreased significantly. The decrease is mainly caused by random variation as one of the four models for koala-XLM-R$_{\text{large}}$ achieved only accuracy of 0.81 when all other models (XLM-R$_{\text{large}}$ and koala-XLM-R$_{\text{large}}$) were in the range of $0.85 - 0.88$.

Model	en	ko	fa	de	bn
XLM-R$_{\text{large}}$	0.836	0.788	0.757	0.873	0.738
koala-XLM-R$_{\text{large}}$	0.837	0.793	0.761	0.850	0.746
Improvement	0.1%	0.6%	0.5%	-2.6%	1.1%

Table 3: Average micro F1 scores for four models. Models trained with different random seeds.

[4] https://github.com/aapop/multiconer_AaltoNLP/blob/master/requirements.txt

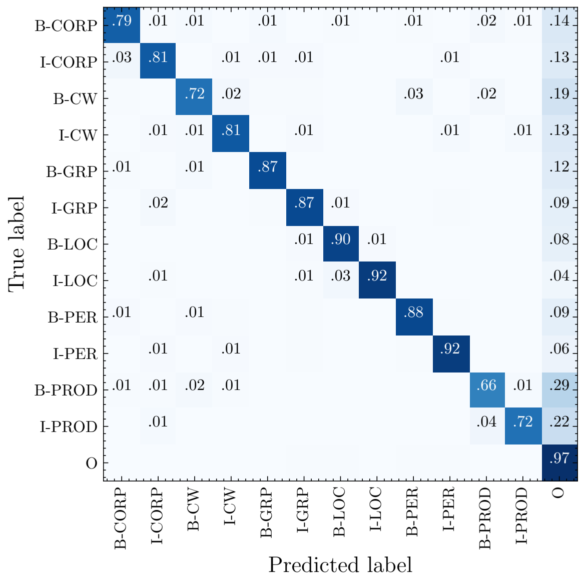


Figure 4: Confusion matrix of Farsi Naive ensemble model normalized on the true labels.

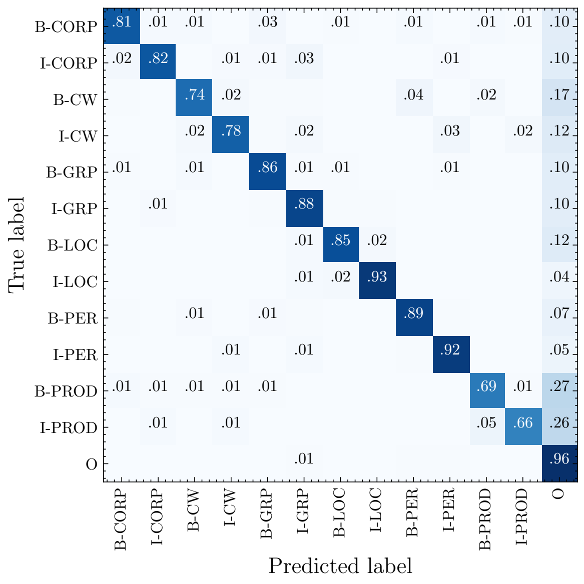


Figure 5: Confusion matrix of Farsi E2E ensemble model normalized on the true labels.

EMEA	Steps	Lr	Batch size	F1
No	-	-	-	**0.782**
Yes	10	10	3	0.779
Yes	10	100	3	0.776
Yes	10	500	3	0.777
Yes	25	10	1	0.771
Yes	30	15	3	0.775
Yes	100	10	3	0.777

Table 4: EMEA ensemble for Bangla

5.2 EMEA

As discussed earlier, we tried to improve our ensembling strategy using entropy minimization. The results are reported in Table 4. Despite testing different approaches, no sign of improvement is present. The testing was conducted on the Bangla development data set. Our setting differs from the original authors as we are not using off-the-shelf language adapters. We probably have too few models and they are not diverse enough.

6 Conclusion

From our efforts, we conclude that naive ensembling improves accuracy with just four models of the same architecture trained with different random seeds. The ensemble of four models outperforms the best individual model across all the tested languages. The E2E ensemble model can provide good accuracy, but the results vary drastically between runs. Task-adaptive pretraining, which has in some cases improved accuracy significantly, yielded only a slight improvement in this task. We also notice a large difference between performance in development and test data. With such limited development data and a large test set, model selection on solely development data yields sub-optimal results. More attention should be paid to model selection and model's generalizability.

Acknowledgments

We thank the Aalto Science-IT project and CSC - IT Center for Science, Finland, for providing the computational resources, that were necessary to participate in this task. We would like to thank the shared task organizers for the excellent arrangements. We thank Feiyi Wang for the initial discussion.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised Cross-lingual Representation Learning at Scale. *CoRR*, abs/1911.02116.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks.

Xiangyang Li, Huan Zhang, and Xiao-Hua Zhou. 2020. Chinese clinical named entity recognition with variant neural structures based on bert methods. *Journal of biomedical informatics*, 107:103422.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Mehryar Mohri. 2012. Foundations of machine learning.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2020. Bertimbau: pretrained bert models for brazilian portuguese. In *Brazilian Conference on Intelligent Systems*, pages 403–417. Springer.

René Speck and Axel-Cyrille Ngonga Ngomo. 2014. Ensemble learning for named entity recognition. In *International semantic web conference*, pages 519–534. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. 2020. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*.

Xinyi Wang, Yulia Tsvetkov, Sebastian Ruder, and Graham Neubig. 2021. Efficient Test Time Adapter Ensembling for Low-resource Language Varieties.

Lingyun Zhao, Lin Li, Xinhao Zheng, and Jianwei Zhang. 2021. A bert based sentiment analysis and key entity detection approach for online financial texts. In *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 1233–1238. IEEE.

DANGNT-SGU at SemEval-2022 Task 11: Using Pre-trained Language Model for Complex Named Entity Recognition

Dang Tuan Nguyen, Huy Khac Nguyen Huynh
Saigon University, Ho Chi Minh City, Vietnam
`{dangnt, hnkhuy}@sgu.edu.vn`

Abstract

This paper describes a system that we built to participate in the SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition, specifically the track Mono-lingual in English. To construct this system, we used Pre-trained Language Models (PLMs). Especially, the Pre-trained Model base on BERT is applied for the task of recognizing named entities by fine-tuning method. We performed the evaluation on two test datasets of the shared task: the Practice Phase and the Evaluation Phase of the competition.

1 Introduction

In natural language processing, Named Entity Recognition (NER) is traditionally a fundamental task that aims to recognize groups of words used to identify people, organizations, places, times, etc. in documents. SemEval-2022 Task 11 focuses on detecting semantically complex and ambiguous entities (Malmasi et al., 2022b). This is a challenging NLP task that has not sufficiently received attention from the research community. In practice, available named entity recognition models find it hard to recognize complex named entities, such as titles of works (*movie/book/song/software titles*) that are not simple nouns (Malmasi et al., 2022b).

Type	Description
Complex entities	Not all entities are proper noun -Noun phrases: *Eternal Sunshine of the Spotless Mind* -Gerunds: *Saving private Ryan* -Full clauses: *Mr. Smith goes to Washington…*
Ambiguous entities and Contexts	Not always entities *-Inside out (movie),* *-Bonanza (game)* *-On the Beach (movie)…*
Emerging Entities	New books/songs/movies released daily, weekly…

Table 1: SemEval-2022 Task 11 illustrates some types of complex and ambiguous entities

To enable the recognition of such entities, we believe that it is necessary to make full use of Pre-trained Language Models (PLMs) to increase efficiency as well as reduce training time, which is also the method we applied in this study.

2 Related Work

Nowadays, there are many NER works and different approaches to solving this problem, such as LSTM (Long Short-Term Memory) (Nut Limsopatham and Nigel Collier, 2016), Transformers (Ashish Vaswani et al., 2017). In particular, the BERT model (Jacob Devlin et al., 2019). Many relevant studies also approach the direction of combining deep learning models and adjusting model parameters to improve the efficiency of named entity recognition in the text (Zheng Yuan et al., 2021). However, entities with complex and ambiguous names appear continuously in increasing numbers. This has posed emerging challenges for the natural language processing field. Many research show that clarifies that named substance acknowledgement is particularly troublesome in circumstances with a low context or in scenarios where the named substances are especially complex (Meng et al., 2021). The misrecognition of entities with complex and ambiguous names significantly affects the performance of natural language processing systems (Andreas Hanselowski et al., 2018). In this paper, we will use PLMs to find ways to improve accuracy in such cases.

PLMs are Pre-trained Language Models with large datasets to be utilized in specific tasks. In particular, BERT has drawn great attention and it is the favorite research direction of the NLP community after achieving state-of-the-art on 11 NLP tasks (Jacob Devlin et al., 2019). Based on BERT, there have been many research directions from the NLP research community: distilBERT (Victor Sanh et al., 2019), RoBERTa (Yinhan Liu et al., 2019), and so on.

3 Data

We will use the training and evaluation **English** dataset provided at SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition to conduct the research with 15300 training examples and 800 validation examples (Malmasi et al,. 2022a).

Language	Train	Dev
BN-Bangla	15,300	800
DE-German	15,300	800
EN-English	15,300	800
ES-Spanish	15,300	800
FA-Farsi	15,300	800
HI-Hindi	15,300	800
KO-Korean	15,300	800
NL-Dutch	15,300	800
RU-Russian	15,300	800
TR-Turkish	15,300	800
ZH-Chinese	15,300	800
Total	168,300	8,800

Table 2: Lists the sizes of the datasets

The data will be formatted according to CoNLL (Malmasi et al., 2022a)

Text	Format
kingdom	B_CW
hospital	I-CW
,	O
lewiston	B-LOC
from	O
stephen	B-PER
king	I-PER
of	O
the	O
same	O
name	O

Table 3: Data Format use in Task 11: Multi-CoNER Multilingual Complex Named Entity Recognition

For the missions of the competition, we will focus on 6 types of entities:

- PER: *Person*
- LOC: *Location*
- GRP: *Group*
- CORP: *Corporation*
- PROD: *Product*
- CW: *Creative Work*

In total, there are 13 tags: B-PER, I-PER, B-LOC, I-LOC, B-GRP, I-GRP, B-CORP, I-CORP, B-PROD, I- PROD, B-CW, I-CW, and O with B is the beginning of a named entity, I is all the words inside an entity and all words outside an entity (Malmasi et al., 2022b)

4 Methodology

In this paper, we use the Transformer library of HuggingFace (Wolf et al.,2020). Currently, the HuggingFace library is considered the most powerful and widely accepted Pytorch interface to deal with BERT. In addition to the support for a wide range of pre-trained language models, the library also includes pre-built modifications of BERT tailored to specific tasks. Then we tested several models based on BERT downloaded from Huggingface as BERT, RoBERTa, and XML-RoBERTa with different versions.

4.1 BASE-BERT

BERT is a transformers model pretrained on a large corpus of English data in a self-supervised fashion (Jacob Devlin et al., 2019). We use the most basic version of BERT and are downloaded from Huggingface with two versions which are **bert-base-uncased** and **bert-large-uncased.**

4.2 RoBERTa

BERT is trained simultaneously with 2 tasks called *Masked LM* (to predict the missing word in a sentence) and *Next Sentence Prediction* (NSP-to predict the next sentence in the current sentence). Meanwhile, to enhance the training process, instead of using the next sentence prediction mechanism from BERT, **RoBERTa** uses a dynamic masking technique, thereby the mask tokens will be changed during the process. The use of a larger batch size shows better performance in training (Yinhan Liu et al., 2019), we applied on **roberta-base** version from Hugging face.

4.3 XML-RoBERTa

XLM-RoBERTa model pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages. It was introduced in (Conneau, Alexis et al., 2019). The **xlm-roberta-large** version we chose to use.

4.4 Experimental Setup

Resources are required to get started; as a large neural network will have to be trained, we use Google Colab Pro to make the best use of the GPU and TPU resources from this environment.

Regarding the hyper-parameter, we have inspected many different values around the "*work well across all tasks*" value recommendations (according to Jacob Devlin et al., 2019) and this is the parameter set that gives the best results: *Batch size: 64, Learning rate: 2e-5, Number of epochs: 10, Maximum sequence length: 128.* Besides, with the desire to increase the recognition efficiency, we try to add a Linear layer and a softmax layer on top to recognize the entity, however, it has no effect on the results.

We import BERT's word tokenizer, which is used to convert text into tokens corresponding to BERT's vocabulary set. BERT requires specifically formatted inputs. For each encoded input sentence, we need to generate *input ids*, *segment mask, attention mask, label*

When the input data is correctly formatted, there comes the fine-tuning stage of the BERT models. For this task, we start with the modification of the pre-trained BERT model to provide outputs for the named entity recognition task, and then we continue to train the model on the dataset until the whole model is correspondent to the task. This is a prominent advantage of using Huggingface Pytorch library which already contains a set of interfaces designed for many NLP tasks.

To evaluate the model, we will use the **Macro Precision, Recall, and F1-score** indexes. The Precision ratio of the number of positive points correctly predicted by the model to the total number of points predicted by the model is Positive. The Recall is the ratio of the number of positive points the model correctly predicted to the total number of points that are actually Positive. F1-score is the harmonic mean of precision and recall (assuming these two quantities are different from 0) (Powers, David, 2008)

$$macro - Precision = \frac{tp}{tp + fp}$$

$$macro - Recall = \frac{tp}{tp + fp}$$

$$macro - F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

5 Results

This section describes the evaluation results of our system with PLMs based on BERT. These results are calculated using the script provided by SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition (Malmasi et al., 2022b)

Model	PER			LOC			CW			GRP			CORP			PROD		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
base-bert-uncased	0.924	0.959	0.941	0.823	0.893	0.857	0.667	0.761	0.711	0.827	0.879	0.852	0.860	0.798	0.828	0.658	**0.864**	**0.747**
bert-large-uncased	**0.959**	0.966	0.962	0.852	**0.910**	**0.880**	0.743	0.790	0.766	0.848	**0.937**	0.890	**0.904**	**0.829**	**0.865**	**0.706**	0.735	0.720
roberta-large	0.906	0.897	0.901	**0.863**	0.889	0.876	0.674	0.716	0.694	0.793	0.847	0.819	0.875	0.798	0.835	0.687	0.776	0.728
xml-roberta-large	0.956	**0.976**	**0.966**	0.857	0.893	0.874	**0.772**	**0.830**	**0.800**	**0.874**	0.911	**0.892**	0.859	0.819	0.838	0.671	0.789	0.725

Table 4: Summary of the scores of all the models tested with 6 types of entities in this paper use dataset on the SemEval 2022 Task 11 (Malmasi et al., 2022a)

Based on the results of table 4, the two models bert-large-uncased and xml-roberta-large are slightly better than the other two models in each entity type. In which xml-roberta-large gives the best results in 3 entity types: PER, CW, GRP, while bert-large-uncased is slightly better in two entity types LOC, CORP and interestingly base-bert-uncased gave the best results in the PROD type.

According to the results in the table 5, it can be observed that testing with the XML-RoBERTa model (Conneau, Alexis et al., 2019). with version xml-roberta-large gives good results compared to the rest of the models when it comes to average performance, followed by the bert-large-uncased model (Jacob Devlin et al., 2019).

Model	Precision	Recall	F1-score
base-bert-uncased	0.804	0.869	0.835
bert-large-uncased	**0.849**	0.876	**0.863**
roberta-large	0.812	0.832	0.822
xml-roberta-large	0.845	**0.882**	**0.863**

Table 5: Summary of the scores of all the models tested with the value of macro average performance in this paper on the SemEval 2022 Task 11 (Malmasi et al., 2022a)

However, in general, the results between the models are not much different, so to improve the efficiency of the NER model, in addition to using PLMs, it is necessary to learn more about other improvement methods to achieve positive results. When analyzing each measure, the BERT model with the bert-large-uncased version gave the highest Precision result (0.849), showing that the BERT model has the highest accuracy in entity labeling. Among the tested models, xml-roberta-large gives the best results in 2 measures: recall and F1-score. Therefore, to participate in the Evaluation phase of SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition (Malmasi et al., 2022b) we selected the model with PLM xml-roberta-large performed on the data set of this phase. with data size of 15300 training examples and 217.818 test examples (Malmasi et al., 2022a), the result is F1-score of 0.67, details as below table 6:

	Precision	Recall	F1-score
Type	**0.6651**	**0.677**	**0.6689**
LOC	0.6401	0.7405	0.6866
PER	**0.8233**	**0.8729**	**0.8474**
PROD	0.6092	0.6433	0.6258
GRP	0.6448	0.6449	0.6448
CW	**0.5508**	**0.5781**	**0.5641**
CORP	0.7222	0.5823	0.6448

Table 6: Results of our system evaluation in dataset validation of Evaluation Phase (Malmasi et al., 2022a)

In Evaluation Phase, the test dataset in this phase is 14 times larger in size than the training dataset (217,818 vs 15300 examples), which is a big challenge for this year's competition.

6 Conclusion

In this paper, we proudly introduced our semantically complex and ambiguous entity recognition system. We tested on different PLMs to evaluate the effectiveness of entity recognition. We trained each model with the dataset provided by the contest (Malmasi et al., 2022a): model BERT with two versions bert-base-uncased and bert-large-uncased, model RoBERTa with roberta-large version and model XML-RoBERTa with xml-roberta-large, and we also tried to fine-tune the hyperparameters to increase recognition efficiency, but there were no significant changes. The evaluation results in the Practice Phase are quite positive. Finally, based on the evaluation results, we used the xml-roberta-large model to participate in the Evaluation phase of the competition, but the results were not good.

In future work, we plan to continue testing on other PLMs. In addition, we will also extend the approach by applying deep learning techniques and theories of language to improve the accuracy in the task of recognizing possible entities, especially complex entities

References

Malmasi, Shervin and Fang, Anjie and Fetahu, Besnik and Kar, Sudipta and Rokhlenko, Oleg. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Malmasi, Shervin and Fang, Anjie and Fetahu, Besnik and Kar, Sudipta and Rokhlenko, Oleg. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recog- nition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Lin- guistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Lan- guage Technologies*, pages 1499–1512.

Nut Limsopatham, Nigel Collier. 2016. Bidirectional LSTM for Named Entity Recognition in Twitter Messages. *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pp: 145–152.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. 2017. Attention Is All You Need. *arXiv:1706.03762 [cs.CL].*

Zheng Yuan, Yijia Liu, Chuanqi Tan, Songfang Huang, Fei Huang. 2021. Improving Biomedical Pretrained Language Models with Knowledge, *BioNLP 2021, arXiv:2104.10344 [cs.CL].*

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1. 4171–4186.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2) co-located with the Thirty-third Conference on Neural Information Processing Systems (NeurIPS 2019)*. 1–5.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692(2019)*

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

Conneau, Alexis & Khandelwal, Kartikay & Goyal, Naman & Chaudhary, Vishrav & Wenzek, Guillaume & Guzman, Francisco & Grave, Edouard & Ott, Myle & Zettlemoyer, Luke & Stoyanov, Veselin. 2020. Unsupervised Cross-lingual Representation Learning at Scale. *8440-8451. 10.18653/v1/2020.acl-main.747.*

Powers, David. (2008). Evaluation: From Precision, Recall and F-Factor to ROC, *Informedness, Markedness & Correlation. Mach. Learn. Technol*, 2.

OPDAI at SemEval-2022 Task 11: A hybrid approach for Chinese NER using outside Wikipedia knowledge

Ze Chen, **Kangxu Wang**, **Jiewen Zheng**
Zijian Cai, **Jiarong He** and **Jin Gao**
Interactive Entertainment Group of Netease Inc., Guangzhou, China
{jackchen, wangkangxu, zhengjiewen}@corp.netease.com
{caizijian01, gzhejiarong, jgao}@corp.netease.com

Abstract

This article describes the OPDAI submission to SemEval-2022 Task 11 on Chinese complex NER. First, we explore the performance of model-based approaches and their ensemble, finding that fine-tuning the pre-trained Chinese RoBERTa-wwm model with word semantic representation and contextual gazetteer representation performs best among single models. However, the model-based approach performs poorly on test data because of low-context and unseen-entity cases. Then, we extend our system into two stages: (1) generating entity candidates by using neural model, soft-templates and Wikipedia lexicon. (2) predicting the final entity results within a feature-based rank model. For the evaluation, our best submission achieves an F_1 score of 0.7954 and attains the third-best score in the Chinese sub-track.

1 Introduction

Named Entity Recognition (NER)(Yadav and Bethard, 2019) aims to detecting the boundaries of named entities and recognizing their categories(e.g., person or location). It plays an important role in many downstream tasks, such as information extraction and question answering.

SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition(MultiCoNER) (Malmasi et al., 2022b) is a shared task which encourages participants to develop NER system to detect semantically ambiguous and complex entities in short and low-context settings for 11 languages. Participants can build a NER model that works only for one language or for all the languages. And an additional track with code-mixed data are offered in this task. Different from ordinary NER, this task focuses on complex and unseen entities. Complex entities, like the titles of creative works(movie/book/software names) are harder to recognize. Additional test sets on questions and short search queries are offered in the test phase, which contains large proportion of unseen entities.

Our main interest is to build a NER system which can process complex entities and adapt to other domains in practical scenarios in Chinese language. This paper describes our two-stage hybrid approach for Chinese NER. A description of datasets provided in this shared task and additional datasets adopted in our system is given in Section 3. The implementation details of our system are listed in Section 4. We first experiment with model-based methods and integrate word semantic feature and gazetteer feature with neural model to improve inference performance. Further, we extend our model system to a two-stage prediction system: entity candidates generation and entity confidence ranking. Confidence ranking is used to pick out high-confidence entities from candidates. With the help of Wikipedia lexicon and soft templates for generating entity candidates, our hybrid system shows good performance and great domain adaption capability in the final evaluation phase.

2 Related Work

Our work is mainly related to the pre-trained language models and some specific strategies for Chinese NER task.

2.1 Pre-trained language models

Transformer-based Language Models e.g BERT (Devlin et al., 2018) have demonstrated that rich, unsupervised pre-training is an integral part of many natural language processing system. RoBERTa(Liu et al., 2019) is a BERT-based model with better performance which bring by different training strategies including training the model longer; bigger batches over more data; removing the next sentence prediction objective; training on longer sequences; and dynamically changing the masking pattern applied to the training data. RoBERTa-wwm (Cui et al., 2021) is a pre-trained language model which modifies the masked language model (MLM) task as a language correction manner and mitigates the discrepancy of the pre-training and fine-tuning stage. It could give significant gains in most various Chinese NER task (Yin et al., 2021).

2.2 Chinese NER

Compared with NER in English, Chinese NER is more difficult since there are no explicit word boundaries in sentences. Word-level information cannot be well modeled. Some approaches resort to performing Chinese NER directly at character-level (Sui et al., 2019; Ding et al., 2019) and some others perform word segmentation first (He and Sun, 2017). However, incorrect word segmentation will result in propagation errors in entity detection, and purely char-based approach will miss the word information. Pre-trained language models, such as BERT, can generate contextual embedding which can outperform other character or word-based approaches (Hu et al., 2020). More importantly, BPE subword segmentation method is employed by BERT-based models and word-level information is not explicitly modeled. Consequently, some researches introduce lexicon information into neural models which results in significant improvement (Ma et al., 2019; Liu et al., 2021). In this task, we implement similar strategies to integrate discrete lexicon and neural representation.

3 Data

We experiment using the datasets shared by MultiCoNER(Malmasi et al., 2022a) on Chinese monolingual track, which consist of 15300 training sentences, 800 validation sentences and 151661 test sentences. Entities are labeled using BIO scheme, and six entity types are involved: person(PER), location(LOC), group(GRP), corporation(CORP), product(PROD) and creative work(CW).

In order to make our model better adapt to ambiguous semantics and insufficient context, we built an entity lexicon from Wikipedia data. We parsed a Wikidata dump and mapped all the entities to our NER taxonomy following the rule from Table 1. We extracted about 3.8 million entities for Chinese language which were mapped to the entity types. Wikipedia entities with multiple categories can be mapped to different entity types in MultiCoNER. Moreover, pre-trained static word embeddings (Song et al., 2018), which provides 200-dimension vector representations for over 12 million Chinese words and phrases pre-trained on large-scale high-quality data, are adopted in this work for integrating the word-level information.

MultiCoNER Entity Types	Wikidata Entity Types
PER	human
	fictional human
GRP	music organization
	sports organization
	newspaper
	educational organization
	cultural institution
CORP	business
	enterprise
CW	creative work
LOC	location
PROD	product

Table 1: The entity types mapping between Wikipedia and MultiCoNER

4 Methodology

Our system classifies NER task into two stages: entity candidates generation and entity confidence ranking. Based on BERT-based model ensembles, soft-template methods and Wikipeida lexicon, we can first generate entity candidates. And then, handcrafted features for each entity candidate are extracted. Finally, a machine learning based rank model is used for entity confidence rank, the entities whose confidence score is above the threshold are regarded as the final predictions. Figure 1 gives an overview of our hybrid approach.

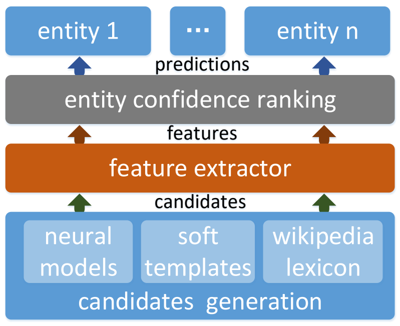


Figure 1: Overview of our hybrid approach

4.1 Entity Candidates Generation

Three approaches are adopted for entity candidates generation: model-based, template-based and lexicon-based. BERT-based model ensembles can achieve named entities recognition, however, its recall performance significantly decreases when dealing with low-context or new-entity cases. To alleviate this effect, soft templates and Wikipedia lexicon is integrated for candidates recall.

4.1.1 Model based approach

Figure 2 gives a glimpse of our model architecture, which consists of three layers: *encoding layer*, *aggregation layer* and *inference layer*. Pre-trained language models are adopted for sentence encoding which can grasp contextual information. However, sentences in Chinese are not naturally segmented, resulting in difficulties in Chinese NER task. Therefore, word semantic representation and contextual gazetteer representation are used in *aggregation* module for incorporating word lexicon information and boundary information into character representations. For further improvement, model ensemble methods are tried for prediction.

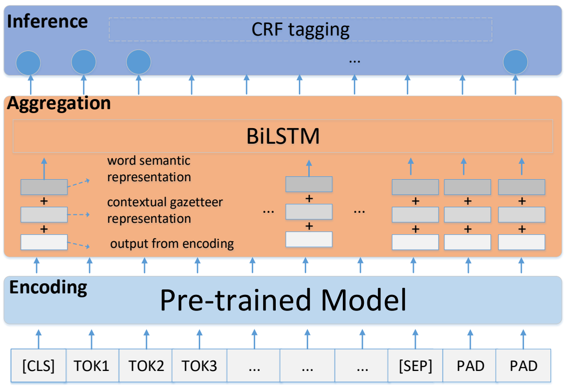


Figure 2: Model Architecture

Encoding: This layer is meant for sequence modeling to capture contextual semantic representation. BERT-based pre-trained models, such as BERT, RoBERTa and RoBERTa-wwm, which have been shown to capture implicit syntactic and semantic knowledge, are tried here.

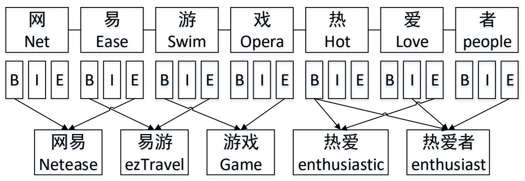
Aggregation: This layer focuses on incorporating word-level information and boundary-dependent features. We use a BiLSTM neural architecture to integrate encoding output with word semantic representation (Ma et al., 2019) and contextual gazetteer representation (Fetahu et al., 2021). An example is presented in Figure 3 for feature constructions. For each character c_i in the input sentence $s = c_1, c_2, ..., c_n$, three word sets(B/I/E) are constructed by:

$$B(c_i) = \{w_{i,k}, \forall w_{i,k} \in eL, i < k \leq n\}$$

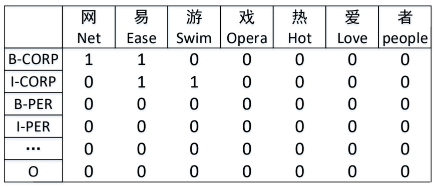
$$I(c_i) = \{w_{j,k}, \forall w_{i,k} \in eL, 1 \leq j < i < k \leq n\}$$

$$E(c_i) = \{w_{j,k}, \forall w_{i,k} \in eL, 1 \leq j < i\}$$

Here, eL represents the Wikipedia entity lexicon. $w_{i,k}$ stands for the span that begins with c_i and ends with c_k. The average word embeddings of each word set are concatenated as a 600-dimension vector(200 for each word set) is regarded as the final word semantic representation. Contextual gazetteer representation for each character is a 13-dimension binary vector, which is introduced by Meng et al. (2021) .



(a) word semantic representation



	网 Net	易 Ease	游 Swim	戏 Opera	热 Hot	爱 Love	者 people
B-CORP	1	1	0	0	0	0	0
I-CORP	0	1	1	0	0	0	0
B-PER	0	0	0	0	0	0	0
I-PER	0	0	0	0	0	0	0
...	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0

(b) contextual gazetteer representation

Figure 3: Features introduced in Aggregation Layer

Inference: Sequential conditional random field(CRF), which can capture the dependency between successive labels, is used in the inference layer for final prediction.

Ensemble: Majority voting method is applied to integrate different model results. In detail, for N different models, if more than N/2 models consider a predicted span belongs to the same entity category, the span is used for final prediction.

4.1.2 Lexicon based approach

An entity lexicon is built from wikidata with category mapping rule listed in Table 1. And then Aho-Corasick algorithm [1] is applied for span extraction. When predicting, word span in that Wikipedia lexicon is extracted as an entity candidate, if that word span maps to multiple entity categories, multiple candidates are generated.

4.1.3 Soft-templates based approach

Soft-templates are mined from training data by a simple statistical strategy. More specifically, we first replace entities in sentences with entity category placeholders, and those replaced frequently occurring sentences are regarded as soft-templates without manually labeled. To make improvements, templates differ only in placeholders are removed.

[1]`https://pyahocorasick.readthedocs.io/`

In the evaluation phase, additional soft-templates are mined from test data. Different from the procedures on training data, pseudo entity labels are generated from model prediction results.

4.2 Entity Confidence Ranking

The following paragraphs describe how we select high confident ones from entity candidates for the final predictions. We build a machine learning model with hand-crafted features to calculate the confidence score of each entity candidate. The hand-crafted features consist of 6-dimension lexical features and 16-dimension statistical features. An example of features extracted for an entity candidate in a sentence is given in figure 4.

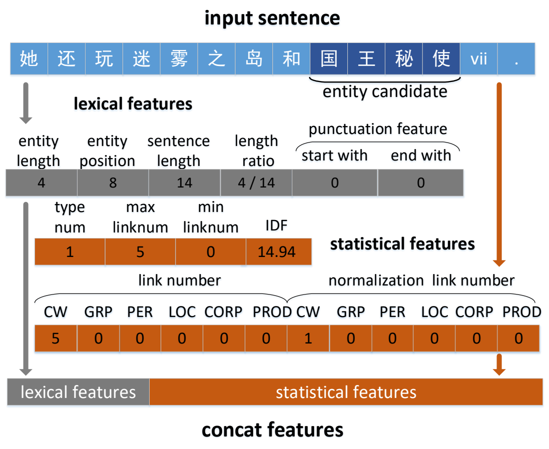


Figure 4: Handcraft Features

A. Lexical Features

- **entity length:** the length of the entity candidate.
- **entity position:** the start position of first character of entity candidate in the input sentence.
- **sentence length:** the length of the input sentence.
- **length ratio:** length ratio between entity candidate and the sentence.
- **punctuation feature:** a two-dimension binary feature indicating whether there is a punctuation at the beginning or end of the entity candidate.

B. Statistical Features

These features are extracted from wiki pages, which can capture entity type information from the link relationships among wiki entities. Two entities are thought to have a link relationship if there is any sitelink in one page which can link to the other. For each entity candidate e_i, the associated wiki pages $P(e_i) = \{p_1, p_2, ..., p_n\}$ are those who have the same label name or alias name to e_i. The number of links which redirect to an entity whose type is $etype_j$ in page p_t is represented as $linknum(p_t, etype_j)$. $etype_j, j = 0, ..., 5$ is one of the six entity types(e.g. PER,LOC,etc.). Therefore, the total number of links for e_i are calculated as $linknum(e_i, etype_j) = \sum_{t=1}^{n} linknum(p_t, etype_j)$.

- **IDF**: inverse document frequency of an entity candidate is calculated from wiki pages.
- **link number**: $\{linknum(e_i, etype_j), j = 0, 1, ..., 5\}$, a six-dimension vector, indicating the number of links in the associated wiki pages of six entity types separately.
- **normalization link number**: normalization value of link number based on entity types, $\{\frac{linknum(e_i,etype_j)}{\sum_j linknum(e_i,etype_j)}, j = 0, 1, ..., 5\}$
- **maximum link number**: maximum value of link number, $\max_{0 \le j \le 5} linknum(e_i, etype_j)$
- **minimum link number**: minimum value of link number, $\min_{0 \le j \le 5} linknum(e_i, etype_j)$
- **number of link types**: the number of entity types where $linknum(e_i, etype_j) > 0$

Then, we adopt LightGBM (Ke et al., 2017) as the multi-label classifier, which is a gradient boosting tree model and performs well on unbalanced classification tasks. For an entity candidate, the confidence on its entity type is calculated by this classifier. If the confidence score is greater than the threshold, e_i is used for the final prediction.

5 Experiments and Results

5.1 Experiment Setup

Our implementation is based on a powerful NLP framework Flair(Akbik et al., 2019), and the Transformers library by HuggingFace(Wolf et al., 2019) for the pre-trained models and corresponding tokenizers.

We first experiment on development dataset with different encoding and aggregation strategies, and we later do an ensemble of these models. During training, the data is processed by batches of size 32 and the maximum length of each sentence is set to 256. In all experiments, we use AdamW optimizer with learning rate set to 2e-5 and train our models for a maximum of 30 epochs. Then, we implement LightGBM for entity confidence ranking with learning rate set to 5e-2, maximum number of leaves set to 50, max-depth set to 6. For the evaluation phase, we mix train and development data and split it randomly for 10-fold cross validation.

Model	Dev P	Dev R	Dev F1
BERT	0.824	0.840	0.832
RoBERTa	0.850	0.837	0.842
RoBERTa-wwm	0.851	0.846	0.847
RoBERTa-wwm-large (I)	0.856	0.860	0.858
+ WE (II)	0.854	**0.863**	0.859
+ GZ (III)	**0.887**	0.842	0.859
+ WE + GZ(IV)	0.867	0.858	**0.861**
Ensemble (V)	**0.913**	0.853	**0.875**

Table 2: Best results achieved by each model on dev dataset, WE and GZ are the shorthand of word semantic representation and gazetteer representation separately

Methods	Dev P	Dev R	Dev F1
Ensemble(V)	0.913	0.853	0.875
Lexicon + LightGBM	0.842	0.779	0.810
Lexicon + V + LightGBM	**0.904**	**0.881**	**0.890**

Table 3: Hybrid results on dev dataset

5.2 Neural Model Results

Table 2 shows the performance of different models and their ensemble approach. The experimental results show that RoBERTa with whole word mask(RoBERTa-wwm) can outperform others in this Chinese language task. The first four rows show the performance of pre-trained Chinese language models. Model I to IV represent different aggregation strategies: no feature introduced, word semantic feature(WE for short) only, gazetteer feature(GZ for short) only, combination of word semantic representation and gazetteer feature representation. We find that the pre-trained model with WE introduced can achieve higher recall and with GZ introduced can achieve higher precision. The last row gives us the results of an ensemble(V) of model I to IV. Model ensemble results in an improvement of about 0.6% on macro-F1 score, indicating that predictions of model I to IV have good complementarity.

5.3 Hybrid Approach Results

Table 3 shows the results of the hybrid approach on dev dataset. We can find that the recall value of model ensemble(V) is far less than precision from the first row. By integrating lexicon-based and ensemble model-based results for entity candidates generation, and adopting LightGBM for entity confidence ranking, the recall value improves more than 3% and the macro-F1 increases by 1.4%.

The results of hybrid approaches on test dataset are given in Table 4. We can find that with lexicon-based entity candidates generation, the recall improves a lot. By integrating lexicon, model V with a LightGBM confidence ranking model, the F1 score increases more than 10%. To make further improvements, soft templates are used for additional candidates generation, which helps gain an improvement of 0.9% on F1 socre.

Methods	Test P	Test R	Test F1
Ensemble(V)	0.710	0.673	0.678
Lexicon + LightGBM	0.484	0.858	0.359
Lexicon + V + LightGBM	0.786	0.806	0.786
+ soft-template(Hybrid)	**0.805**	**0.794**	**0.795**

Table 4: Results of different approaches on test

Methods	LOWNER F1	Orcas F1	MSQ F1
Ensemble(V)	**0.854**	0.582	0.683
Hybrid	0.852	**0.747**	**0.822**

Table 5: Results of different domains on test

To further analyze the differences and respective advantages of different approaches, detail results of different domains are listed in Table 5. For the ensemble model V, when compared to LOWNER results, the results for MSQ and ORCAS are worse. This large gap shows that the existing model approach cannot generalize well.

6 Conclusion

In this paper, we introduce a hybrid approach for Chinese NER, which contains two stages: entity candidates generation and confidence ranking. We find that integrating word semantic representation and gazetteer representation can improve the performance of neural model-based approach. In particular, our ensemble model of different aggregation strategies performs better. However, due to the limitation of training data, the performance of neural model-based approach drops sharply in other new domains.

Considering that there are new entities and other domain sentences in test sets, we can use Wikipeida lexicon and soft templates to help recall unseen entities, and an entity confidence ranking model is built which results in significant improvement on test sets. For future work, semi-supervised approaches or data augmentation methods could be explored to alleviate the limitation of training data.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ruixue Ding, Pengjun Xie, Xiaoyan Zhang, Wei Lu, Linlin Li, and Luo Si. 2019. A neural multi-digraph model for chinese ner with gazetteers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1462–1467.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Hangfeng He and Xu Sun. 2017. A unified model for cross-domain and semi-supervised named entity recognition in chinese social media. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Yuting Hu, Suzan Verberne, D Scott, N Bel, and C Zong. 2020. Named entity recognition for chinese biomedical patents. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 627–637. International Committee on Computational Linguistics.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced chinese sequence labeling using bert adapter. *arXiv preprint arXiv:2105.07148*.

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

Ruotian Ma, Minlong Peng, Qi Zhang, and Xuanjing Huang. 2019. Simplify the usage of lexicon in chinese ner. *arXiv preprint arXiv:1908.05969*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 175–180.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2019. Leverage lexical knowledge for chinese named entity recognition via collaborative graph network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3830–3840.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Vikas Yadav and Steven Bethard. 2019. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*.

Xunwei Yin, Shuang Zheng, and Quanmin Wang. 2021. Fine-grained chinese named entity recognition based on roberta-wwm-bilstm-crf model. In *2021 6th International Conference on Image, Vision and Computing (ICIVC)*, pages 408–413.

Sliced at SemEval-2022 Task 11: Bigger, Better? Massively Multilingual LMs for Multilingual Complex NER on an Academic GPU Budget

Barbara Plank
Center for Information and Language Processing (CIS), LMU Munich, Germany
Department of Computer Science, ITU Copenhagen, Denmark
bplank@itu.dk

Abstract

Massively multilingual language models (MMLMs) have become a widely-used language representation method, and multiple large MMLMs were proposed in recent years. A trend is to train MMLMs on larger text corpora or with more layers. In this paper we set out to evaluate recent popular MMLMs on detecting semantically ambiguous and complex named entities with an academic GPU budget. Our submission of a single model for 11 languages on the SemEval Task 11 MultiCoNER shows that a fine-tuned XLM-R$_{large}$ outperforms the more recent RemBERT, ranking 9th from 26 submissions in the multilingual track. Compared to RemBERT, the XLM-R model has the additional advantage to fit on a slice of a multi-instance GPU. As contrary to expectations and recent findings, we found RemBERT to not be the best MMLM, we further set out to investigate this discrepancy with additional experiments on multilingual Wikipedia NER data. While we expected RemBERT to have an edge on that dataset as it is closer to its pre-training data, surprisingly, our results show that this is not the case.

1 Introduction

Pre-trained language models have revolutionized the field of Natural Language Processing (NLP) in recent years (Peters et al., 2018; Devlin et al., 2019; Zhuang et al., 2021). Especially for cross-lingual transfer learning or creating a single multilingual model, pre-trained massively multilingual language models (MMLMs) have become a de-facto standard (Conneau et al., 2020; Chung et al., 2021).

MMLMs such as BERT and XLM-R share the same underlying idea: multilingual representations are obtained by learning from large text collections in multiple languages and are trained using a language modeling objective. MMLMs, in contrast to alternative cross-lingual transfer strategies, thus do not rely on explicit alignment via parallel data and explicit transfer via translations with e.g. annotation projection. MMLMs, together with the pre-training and fine-tuning paradigm, have enabled impressive results (Conneau et al., 2020; Lauscher et al., 2020; Müller-Eberstein et al., 2021).

This paper describes our submission to Task 11 on Multilingual Complex Named Entity Recognition (MultiCoNER) (Malmasi et al., 2022b,a). We evaluate several recent MMLMs in a fine-tuning regime to answer the following main research question (RQ1): *To what extent are more recent larger LMs outperforming earlier MMLMs for the task of multilingual complex NER?* To do so, we test four MMLMs (mBERT, XLM-R base and large and the most recently proposed RemBERT). As other NER datasets exists, albeit with different labels, we further explore multi-task learning (RQ2): *To what extent can we improve upon MultiCoNER by using cross-lingual cross-domain NER data as auxiliary data?* Finally, as the MMLMs were pre-trained on different kinds of data, we include experiments on a second NER dataset, to answer RQ3: *To what extent does RemBERT outperform XLM-R when the pre-training data is a closer match?*

Our contributions are:

- We train a single multilingual model on MultiCoNER, which to our knowledge, is the largest multilingual NER evaluation campaign to date in terms of manually-annotated multilingual training data availability. We compare four MMLMs for the task and examine task performance and GPU budget (here: 20gb).
- Surprisingly, we find that RemBERT does not work well. To shed more light on this, we run additional experiments on a NER benchmark which is closer to RemBERT's pre-training data and prior work. Overall we find XLM-R$_{large}$ to provide the best performance and a good space/training-time trade-off.

Model	Model Name	Language Variety	Languages	Vocab	H_dim	Layers	Params
mBERT	`bert-base-multilingual-cased`	Wikipedia	104	120k	768	12	110M
XLM-R_{base}	`xlm-roberta-base`	CommonCrawl	100	250k	768	12	270M
XLM-R_{large}	`xlm-roberta-large`	CommonCrawl	100	250k	1024	24	550M
RemBERT	`google/rembert`	CommonCrawl+Wikipedia	110	250k	1152	32	559M

Table 1: Overview of pre-trained massively multilingual language models (MMLMs) used in this work. All 12 languages used in MultiCoNer are part of the pre-training data of all MMLMs.

2 Experimental Setup

This section describes the model, all data sets and the multilingual language models used in this work.

2.1 Transformer-CRF

We use a transformer-CRF model for NER, implemented in the MaChAmp toolkit (van der Goot et al., 2021) (v0.3 beta). The toolkit enables easy exchange of pre-trained LM for fine-tuning as well as multi-task learning. Our model is a single MMLM fine-tuned with a single CRF decoder.

To train MaChAmp, we use the proposed default parameters (van der Goot et al., 2021), which have shown to work well across tasks with a learning rate of $lr = 0.0001$. We train the model for 20 epochs and select the best checkpoint using the provided dev data (`multi_dev`).

2.2 MMLMs

As our main RQ is to test the effect of using different pre-trained massively multilingual language models, we opted for four well-known variants: multilingual BERT (Devlin et al., 2019), XLM-R with both the *base* (XLM-R_b) and *large* (XLM-R_l) version (Conneau et al., 2020), and the more recently introduced RemBERT (Chung et al., 2021).

An overview of the MMLMs is provided in Table 1, and summarized as follows:

- mBERT: trained using both Masked-Language Model (MLM) and next-sentence prediction tasks on Wikipedia data and trained with an exponentially decaying smoothing distribution over languages with $\alpha = 0.7$, i.e., to down-scale high-resource languages and up-scale lower-resource languages; 12 layers.

- XLM-R: trained using only MLM on CommonCrawl data and trained with an exponentially decaying smoothing distribution with $\alpha = 0.3$ (more aggressive smoothing); with 12 (base) or 24 (large) layers.

- RemBERT: trained using only MLM[1] on Wikipedia and CommonCrawl data, with an exponentially decaying smoothing distribution $\alpha = 0.5$; trained with decoupled input and output embeddings and parameters redistributed over 32 layers.

We observe that the MMLMs are trained on 100, 104 and 110 languages, where RemBERT is the MMLM with the largest amount of language coverage (110). We note that for MultiCoNER, all 11 target languages are included in the the pre-training material of all four MMLMs. What differs amongst others is the vocabulary size, the number of layers, the pre-training method, the pre-training data and the number of parameters. XLM$_{large}$ and RemBERT are closest in terms of total number of parameters, with RemBERT having an additional 9M parameters over XLM-R_{large}; mBERT and XLM-R_{base} are a fifth and half of the number of parameters, respectively. What stands out is the type of pre-training data, as the language variety that these MMLMs were trained on differs. While the ones created by researchers at Google opted mainly for Wikipedia data (mBERT, RemBERT), the XLM-R models from FAIR research are trained on a cleaned CommonCrawl dump (Wenzek et al., 2020).

2.3 MultiCoNER Data

The training dataset provided by MultiCoNER organizers is one of the largest and most diverse multilingual NER datasets available to date in terms of training/dev/test sizes and language/domain coverage (presumably manually annotated).[2]

In contrast, the Panx (WikiAnn) dataset (Pan et al., 2017; Rahimi et al., 2019) covers a much larger span of languages (over 200) but its coverage is limited to Wikipedia data, and it contains fewer entity types and semi-automatic annotation. Moreover, MultiCoNER is set up for complex NER,

[1] `shorturl.at/pwyFQ`

[2] At the time of writing this system paper we did not have further information on how the data was annotated.

Multilingual	Train	dev	test
sentences	168.3k	8,800	472k
token	2,752,814	144,641	4,565,160
word types	360,679	47,570	556,102
type/token	0.13	0.33	0.12
entities	237,212	12,513	n/a
PER	43,953	2,342	n/a
PROD	25,001	1,848	n/a
CORP	32,890	1,738	n/a
CW	38,373	2,015	n/a
LOC	54,030	2,932	n/a
GRP	32,846	1,638	n/a

Table 2: Overview of the MultiCoNER dataset. Every sentence in train and dev contains at least one entity.

Model	**micro F1**	**train h**	**GPU m**	**fits?**
$lr = 0.00001$				
mBERT	81.31	4h20	8gb	✓
RemBERT	83.89	7h40	27gb	✗
$lr = 0.0001$				
RemBERT	85.03	7h30	21gb	✗
XLM-R$_b$	83.89	7h13	10gb	✓
XLM-R$_l$	**86.32**	10h30	**16gb**	✓
RemBERT+aux	85.00	9h30	23gb	✗
XLM$_l$+aux	85.89	11h30	19gb	✓

Table 3: Result of training the transformer-CRF with different MMLMs on the multilingual development set. Training time (in hours), GPU max memory usage (approximate) and whether the training fits a single slice of a A100 GPU (the GPU was partitioned into two 20gb slices using NVIDIA's mig mode). XLM$_l$ takes more time but less memory, which enables parallel training of two XLM$_l$ models on a single sliced GPU.

which include further challenges such as detecting named entities on search queries and code-mixed data (Meng et al., 2021; Fetahu et al., 2021). MultiCoNER contains 6 entity types (person, location, product, corporation, groups but also complex entities) for the following 11 languages: English (en), Spanish (es), Dutch (nl), Russian (ru), Turkish (tr), Korean (ko), Persian/Farsi (fa), German (de), Chinese (zh), Hindi (hi), and Bangla (bn). While we focus on the multilingual track, and the data provided for it (`multi`), the shared task further features monolingual tracks and a code-mixed track containing data of some of these languages.

Table 2 provides an overview of the MultiCoNER multilingual data. It is a very large training set of 168.3k sentences, 2.7M tokens and over 237k entities spanning 11 languages. Table 6 in Table 2 shows the size of the dev and test portions (note the very large test data with 472k sentences) and the distribution over the 6 entity types in the training and dev data. Location (LOC) and person names (PER) constitutes the largest portion of entities, followed by CW, corporate names (CORP) and group names (GRP) and the least frequent entity type is product names (PROD). The appendix lists sizes of individual language test files.

2.4 Auxiliary or Matching Data?

For RQ2, we use a multi-task learning setup, and model the MultiCoNER task as the main task with a CRF output decoder, and add a second CRF-decoder that predicts NER types from the union of three cross-lingual cross-domain datasets. In particular, we use German data (Benikova et al., 2014), and two recently proposed derivatives annotated on top of Danish (Plank et al., 2020) and EWT-NNER on top of the English Web Treebank (Plank, 2021). While all three data sources were annotated for nested NERs (a two-level annotation scheme, which annotates e.g., a location for "Birmingham" inside "University of Birmingham"), we here use only their inner layer entities. In addition, these data contain slightly different entity types with finer-grained subsets: location, person, organization and miscellaneous entities with two additional suffixes that add derivations (e.g., adjectival forms like "Brasilian" and partial NERs like "Nintendo-based"). The total auxiliary training data (we take the union of English, Danish and German data) consists of a total of 21.4k sentences (roughly 8% of the MultiCoNER main task data) with 42.6k entity annotations. We train the multi-task model jointly by full-parameter sharing, no loss weighting and selecting the best checkpoint using the sum over both main and auxiliary task development span-F1.

For RQ3, we compare the two best MMLMs on a NER benchmark derived from Wikipedia (Pan et al., 2017) (WikiAnn/Panx). We follow the setup of Lauscher et al. (2020) and use 12 languages: Indian (in), English (en), Arabic (ar), Finnish (fi), Hindi (hi), Japanese (ja), Russian (ru), Turkish (tr), Basque (eu), Hebrew (he), Italian (it), Korean (ko), Swedish (sv) and Chinese (zh), which use the same splits as RemBERT (Chung et al., 2021), which were provided by Rahimi et al. (2019).

2.5 NER evaluation: macro-F1 vs micro-F1

For evaluation and model selection, we use the CoNLL span-F1 which is a micro-F1 over span-based entity F1 scores. We note that some earlier work report Accuracy, which can be misleading for NER due to the high number of non-entity tokens typical for the task. The MultiCoNER organizers opted for span-based *macro* F1. While both micro and macro F1 consider entities correct only if both the entity boundaries and the labels match, the MultiCoNER macro entity-based F1 is typically lower and hence a more conservative measure, particularly when entity types are unbalanced, which is the case for MultiCoNER. Hence, we adapted a Python-version of the Conlleval scorer to include macro-F1.[3] We report macro-F1 for the aggregated evaluation measures.

3 Results

Table 3 provides the main results of a single model trained on the MultiCoNER `multi_train` data and evaluated on the `multi_dev` portion.

Bigger is not always better From the MMLM comparison in Table 3 we first observe that mBERT is outperformed substantially by more recent MMLMs. As expected, XLM-R$_l$ outperforms XLM-R$_b$. However, regarding RQ1 our results show that bigger is not always better: XLM-R$_l$ outperforms the more recent and even larger RemBERT model. Surprisingly, this is consistently the case over all target languages.

XLM-R$_l$ is more space efficient While XLM-R$_l$ is the best MMLM in terms of F1 on multilingual complex NER, for an academic GPU budget XLM-R$_l$ is also more efficient: it still just fits a single 20gb GPU slice (NVIDIA A100 40GB GPU split into two slices), at a cost of slightly longer training duration. This means we can fine-tune 2 XLM-R large models in parallel, while only one RemBERT. Further details are given in Table 3.

Test set results Table 4 provides the results on the MultiCoNER test set (last column, `multi_test`). The results confirm the findings from dev: XLM-R$_l$ results in the best model, and substantially outperforms RemBERT. This model ranks 9th in the multilingual track out of 26 participating systems. The auxiliary task (RQ2) fails to provide additional signal, which we hypothesize might be due to the high-resource setup (already a high amount of training data exists), but this would need further investigation. While `multi_test` already contains test data from all 11 languages, the shared task provides further monolingual (and code-mixed) test sets, which is not equal to the sum in `multi_test`. We submitted runs of the best models on these individual test sets which confirm that XLM-R$_l$ remains the strongest MMLM for NER on MultiCoNER.

[3]Available at https://github.com/bplank/conlleval

4 Discussion

In this section, we provide a deeper analysis of our results, adding an additional experiment on data outside of the shared task. First, we examine the per-class F1 score on MultiCoNER, to rule out that the strong results of XLM-R$_l$ are purely due to strong results on a few frequent entity types. Second, we compare RemBERT vs XLM-R$_l$ on a Wikipedia NER dataset to answer RQ3. Previous findings suggest that RemBERT is a stronger model for multilingual NER (Chung et al., 2021) by testing it on *on Wikipedia* data, which is also closer to its pre-training data. So we test whether this is the case with our model as well.

Test set results per entity Figure 1 provides a breakdown of the three best models, showing the per-class F1 scores for RemBERT, XLM-R$_l$ and XLM-R$_l$+aux. The results show that XLM-R$_l$ outperforms RemBERT over all six entity classes, and that the multi-task setup consistently hurts over all entity types.

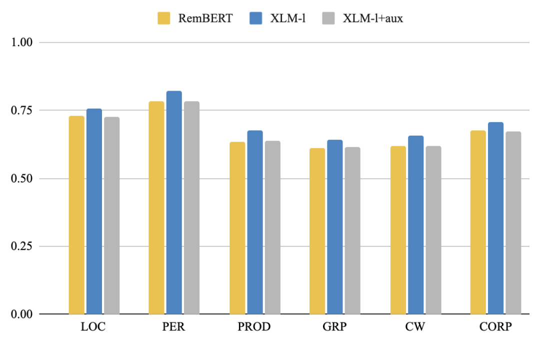


Figure 1: Per-class F1 score on the multilingual test set.

Bigger and better because of better pre-training match? Contrary to our findings, previous work suggests that RemBERT outperforms XLM-R on NER (Chung et al., 2021). We hypothesize that this is not the case on MultiCoNER due to the more

macro-F1	bn	de	en	es	fa	hi	ko	nl	ru	tr	zh	mix	multi
RemBERT	57.14	77.94	73.39	73.62	61.49	60.77	64.59	76.24	70.99	64.97	62.46	70.99	67.61
RemBERT+aux	58.06	77.19	73.27	73.48	62.39	60.09	65.03	75.55	71.10	65.42	62.61	70.79	67.66
XLM-R$_l$	63.05	78.90	74.54	75.11	68.66	67.00	70.66	77.66	73.73	68.77	65.21	72.74	**71.07**

Table 4: Results on the test set. XLM$_l$ is the model that achieved the 9th rank in the multilingual track. Evaluation of this model on all single language and the code-mixed test set included for completeness.

diverse data. A manual inspection reveals that the MulitCoNER data includes both Wikipedia-style data, but also questions without question symbols and short search queries. Therefore, we investigate whether RemBERT instead outperforms XLM-R on a Wikipedia NER benchmark, which matches RemBERT's pre-training data better, cf. Table 1.

	EN	ZH	TR	RU	AR	HI	EU
RB	86.2	82.4	94.1	90.9	90.7	90.7	92.9
XLM	85.9	83.0	93.9	91.3	91.5	92.2	93.2
	FI	HE	IT	JA	KO	SV	avg
RB	92.0	89.5	93.2	76.6	89.6	95.6	89.5
XLM	92.5	90.3	93.1	77.9	90.9	95.9	**90.1**

Table 5: RemBERT (RB) vs XLM-R$_l$ (XLM) on Wikipedia/Panx. Except for EN, TR and IT, XLM-R outperforms RB. Macro-averaged span micro F1 of 90.1.

Table 5 show the results on 13 languages (averaged over 3 runs). The results show that RemBERT has a slight advantage on 3 out of the 13 languages (EN, TR, IT), but overall and on 9 out of the 13 languages XLM-R$_l$ performs substantially better. This additional experiment surprisingly dis-confirms our hypothesis that RemBERT would have an advantage on this Wikipedia/Panx NER data due to the better pre-training data match. While this is in contrast to previous findings on Panx (Chung et al., 2021), the reason is less clear. We studied the literature and found a study on another task (quality estimation) that similarly reports negative results: replacing the XLM-R decoder with a RemBERT decoder performs better only for 1 language pair out of 4 in quality estimation (Treviso et al., 2021). We conclude that XLM-R$_l$ remains the best MMLM for multlingual NER, as tested across two benchmarks (MultiCoNER and WikiAnn).

5 Limitations

We provided a study on four MMLMs for transformer-based multilingual complex NER to shed lights on MMLMs and GPU usage. Our study is limited in number of MMLMs tested, and the fact that we provide only approximate GPU memory consumption figures.

6 Conclusions

We test four massively multlingual language models as encoders for multilingual complex NER. Our results show that XLM-R$_l$ results in the overall best model, and surprisingly outperforms the more recently proposed RemBERT, also in terms of GPU memory consumption. While auxiliary-task training did not further prove promising, we additionally studied the discrepancy between RemBERT and XLM-R on a second benchmark (PANX/WikiAnn data). While we hypothesized that RemBERT would outperform XLM-R, our results show that this is not the case. Overall, a bigger model might not be the best choice, especially not for an academic GPU budget.

Code, scripts and shared task prediction files (labels only) available at: `https://github.com/bplank/multiconer2022`

Acknowledgements

I would like to thank Rob van der Goot for feedback on earlier drafts of this paper. Thanks to my DFF project number 9063-00077B for providing the GPUs used in this work.

References

Darina Benikova, Chris Biemann, and Marc Reznicek. 2014. NoSta-D named entity annotation for German: Guidelines and dataset. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2524–2531, Reykjavik, Iceland. European Language Resources Association (ELRA).

Hyung Won Chung, Thibault Févry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2021. Rethinking embedding coupling in pre-trained language models. In *ICLR*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Max Müller-Eberstein, Rob van der Goot, and Barbara Plank. 2021. Genre as weak supervision for cross-lingual dependency parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4786–4802, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

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

Barbara Plank. 2021. Cross-lingual cross-domain nested named entity evaluation on English web texts. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1808–1815, Online. Association for Computational Linguistics.

Barbara Plank, Kristian Nørgaard Jensen, and Rob van der Goot. 2020. DaN+: Danish nested named entities and lexical normalization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6649–6662, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Marcos Treviso, Nuno M. Guerreiro, Ricardo Rei, and André F. T. Martins. 2021. IST-unbabel 2021 submission for the explainable quality estimation shared task. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 133–145, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. 2021. A robustly optimized BERT pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1218–1227, Huhhot, China. Chinese Information Processing Society of China.

A Data statistics MultiCoNER

Lang	**Sentences/entities**	
	dev	test
ba	800/800	133,119/-
de	800/1,239	217,824/-
en	800/1,230	217,818/-
es	800/1,176	217,887/-
fa	800/1,213	165,702/-
hi	800/828	141,565/-
ko	800/1,302	178,249/-
nl	800/1,157	217,337/-
ru	800/1,042	217,501/-
tr	800/1,245	136,935/-
zh	800/1,281	151,661/-
multi	8,800/12,513	471,911/-

Table 6: Data statistics of dev/test of MultiCoNER.

Infrrd.ai at SemEval-2022 Task 11: A system for named entity recognition using data augmentation, transformer-based sequence labeling model, and EnsembleCRF

JiangLong He, Akshay Uppal, Mamatha N,
Shiv Vignesh, Deepak Kumar, Aditya Kumar Sarda
Infrrd.ai
{jianglong,akshayuppal,mamathan}@infrrd.ai
{shivvignesh,deepakumar,adityasarda}@infrrd.ai

Abstract

In low-resource languages, the amount of training data is limited. Hence, the model has to perform well in unseen sentences and syntax on which the model has not trained. We propose a method that addresses the problem through an encoder and an ensemble of language models. A language-specific language model performed poorly when compared to a multilingual language model. So, the multilingual language model checkpoint is fine-tuned to a specific language. A novel approach of one hot encoder is introduced between the model outputs and the CRF to combine the results in an ensemble format. Our team, **Infrrd.ai**, competed in the MultiCoNER competition. The results are encouraging where the team is positioned within the top 10 positions. There is less than a 4% percent difference from the third position in most of the tracks that we participated in. The proposed method shows that the ensemble of models with a multilingual language model as the base with the help of an encoder performs better than a single language-specific model.

1 Introduction

In conll-2003 (Sang and De Meulder, 2003), a shared task was conducted to identify the named entities such as person (PER), location (LOC), organization (ORG), and miscellaneous (MISC). Over a period of time, there was an improvement in the developed systems (Marrero et al., 2009) which resulted in an improved performance that resulted in an increased number of entities. The named entities which needed to be identified and extracted were now six. They are person (PER), location (LOC), organization (ORG), group (GRP), product (PROD), and creative work (CW). Some of these named entities are created by coining a new word that may be non-existent or a combination of existing words in the entire corpus of words in a language. People are most likely interested in coining a new term from the list of words to have an identity tag of a location, an event, or similar concepts. The newly coined words become novel or emerging entities in the list of entities (Derczynski et al., 2017). These words form a most part of ambiguous words, where the word belongs to a particular entity or not, and is difficult to judge. For example, Microsoft is an organization, Windows is a product, and Microsoft Windows is a software product. Apart from this, creating a non-existent word as an entity is an expression of the creativity of the creator which belongs to the creative word entities. A competition was conducted in WNUT 2017 to identify the novel and emerging entities that look for unseen entities as described earlier (Derczynski et al., 2017). These types of entities are complex enough that even a person may miss the context of the entity and represent them differently.

The MultiCoNER competition brings in the additional dimension of complexity of low resources (Malmasi et al., 2022a,b). In a low resource language, the amount of training data available is limited within which the model has to learn to discriminate between the entities and identify them correctly. The model is exposed in this scenario to understand the unseen word that was not part of the training data and doesn't have annotated information to predict. Hence, the development of a model to examine the competition test data is more challenging than ever.

Our contributions and observations are summarized as follows:

- We explore various word-level data augmentation strategies such as LwTR, SR, MR, SiS, and bert-based token augmentation to improve the dataset size when training the transformer-based sequence labeling models. It is shown that the data augmentation increases the model generalization on the test set.

- Performing transfer learning using a multilingual sequence labeling model as an initializer improves the performance on language-specific tracks.
- Introduced the ensemble learning model, 'EnsembleCRF,' that solves the IOB scheme constraint when using majority voting. By learning to optimally combine the model predictions, EnsembleCRF also learns to avoid mistakes made by single sequence labeling models.

2 Related Work

The transformer-based language models fared better in the identification and extraction of entities from a given text. Since there is a necessity for a large amount of training data which provides a much-needed boost in accuracy. Sometimes, a model trained on a huge data in one or more languages is used in another or different language. These types of models are commonly known as cross-lingual language model (Conneau and Lample, 2019) or multilingual language model (Conneau et al., 2019). A fine-tuned language model has a better performance compared to a multilingual language model. But, the multilingual language model is more adaptable across different languages, which is not available for a fine-tuned language model. The researchers started exploring the amount of data required to train a language model. In some cases, the amount of data available in a language with annotation is very limited. These languages are termed low-resource languages. Due to this limitation, the model may not be aware of the complete set of words in the low-resource languages. Here, we are exploring to understand the performance of a transformer-based language model in low-resource constraints. The challenges involved in recognizing complex entities in low-resource environments (Meng et al., 2021; Fetahu et al., 2021) have led to the creation of the competition data (Malmasi et al., 2022a,b).

In low-resource scenarios, different approaches have been adapted to overcome the constraints. The available fine-tuned transformer-based model such as BERT is bootstrapped to improve the accuracy of NER (Yu et al., 2020). A prompt-guided attention layer is used as part of a transformer model by creating a semantic-aware answer space for tuning the model for further betterment (Chen et al., 2021). Sentence reconstruction approach to enhance low resource sequence tagging by utilizing the knowledge of high resource data (Perl et al., 2020). A common approach used on low resource languages is to use cross-lingual transfer learning, where a model trained on high resource language is used as the reference. An active learning mechanism was used to improve the performance of NER (Chaudhary et al., 2019). A teacher-student knowledge transfer model technique has shown to give effective results on low resource NER tasks (Izsak et al., 2019). An unsupervised cross-language transfer learning method where the encoders trained on the source and target language together using adversarial learning followed by augmented fine-tuning technique (Bari et al., 2020).

3 Methodology

Individual pre-trained models were used to evaluate the training and the dev set. Based on the empirical results, we decided to train a baseline language with multiple languages for 20 epochs and then perform transfer learning to train the monolingual models.

3.1 Main architecture

The training set with the following split, 151470 train + 8800 dev + 16830 test sentences, is tokenized and fed to the model xlm-roberta-large to generate the baseline multilingual checkpoint. The embeddings of the transformer model are then passed through the dropout layers. We have three types of dropouts in the mix as shown in Figure 1. The standard dropout with the probability of 0.3, the word dropout with the probability of 0.05, and finally the locked dropout with the probability of 0.5 were used. These embeddings are then linearly reprojected into a vector of size 1024. The reprojected vector is passed through a BiLSTM layer with 256 nodes, which generates a vector of size 512. This output vector is then passed through a CRF layer to generate the class label with IOB sequence tags.

The model is trained for 20 epochs to obtain the starting checkpoint for all the monolingual models. The performance of the monolingual models got a significant boost with cross-language training. We tried to train the last, the last two, the last three, and the last four layers of the transformer but did not get a significant boost while training more layers, so for the final step of training, we resorted to training the last layer of the transformer.

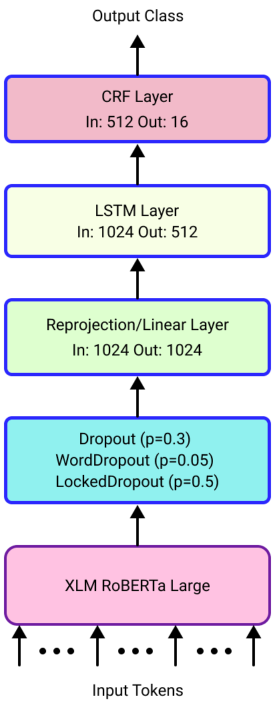


Figure 1: The architecture of the multilingual model was developed to train the training set of the competition data.

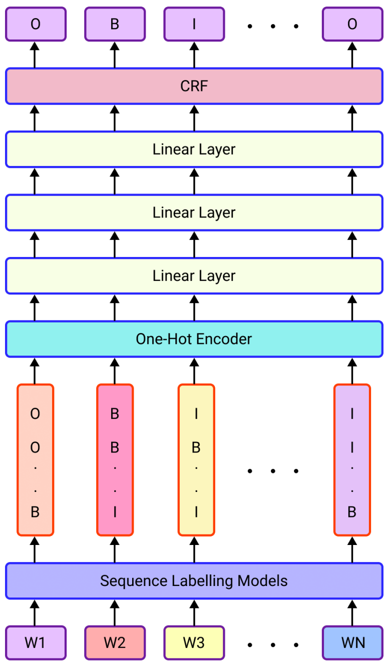


Figure 2: The architecture of EnsembleCRF model. Given a sentence as input, each of the sequence labeling models will output the name entity prediction in IOB format. One hot encoder combines them to generate ensemble output from a Conditional Random Field model.

The generated baseline multilingual checkpoint is loaded and fed with the augmented dataset for each monolingual task to obtain the respective language models. We have used different augmentation techniques described in the Training and Evaluation section. All the sentences provided by the competition organizers were used to generate the augmented data.

3.2 Ensemble architecture

A simple ensemble strategy of *Majority Voting* is developed. Given a set of M sequence labeling models denoted as $C = \{c_1, c_2, ..., c_M\}$ and an input sentence denoted as $S = \{w_1, w_2, ..., w_n\}$, where each w is a word from S. Each model from C will output a sequence of predictions for each word w in sentence S. Let $O^s_{c_i} = \{O^{c_i}_{w_1}, O^{c_i}_{w_2}, ..., O^{c_i}_{w_n}\}$ denote the prediction output of model C_i on sentence S. $O^{c_i}_{w_j}$ denotes the prediction of model C_i on word w_j in IOB format. The set of outputs for all models in C on sentence S will be denoted as

$$O_S = \{O^s_{c_1}, O^s_{c_2}, ..., O^s_{c_M}\} \tag{1}$$

The *Majority Voting* strategy takes all model's predictions of word w_j and outputs the most frequent prediction as to the final prediction for w_j. An obvious issue with *Majority Voting* is the IOB scheme constraint. The final ensemble result is not guaranteed to be passing all the constraints where (I) tag must follow and (B) tag and the entity of neighbor (B) and (I) tag must be the same. We introduce an ensemble learning approach via sequence labeling called 'EnsembleCRF' as shown in Figure 2.

The model outputs are stacked together and passed through a one-hot encoder, three linear layers, and CRF. The CRF layer is trained to optimally combine the model predictions to form a new set of predictions. The addition of the three linear layers helped in the performance improvement. The EnsembleCRF model is of the form

$$C_{en} = EnsembleCRF(C = \{c_1, c_2, ..., c_M\}, \\ D_{en} = \{X, Y\}) \tag{2}$$

D_{en} is the ensemble learning dataset composed of X and Y. $X = \{O_{s_1}, O_{s_2}, ..., O_{s_k}\}$ is created by using model set C and set of input sentences $\{S\} = \{S_1, S_2, ..., S_k\}$ with size K. Each element of X is defined as equation (1). Y is the ground

Table 1: The language models used for different languages with the strategies adapted and the hyperparameters used for training the model. The last column shows the macro-averaged F1 score on the competition test set.

Model No	Language Model	BiLSTM	CRF	Transfer Learning	External Dataset	Aug. Data	Train with Dev	Learning Rate	F1-score
	English								
1	roberta-large	Y	Y	N	N	Y	N	3e-3	0.7072
2	roberta-large	Y	Y	N	N	Y	Y	3e-3	0.715
3	xlm-roberta-large	Y	Y	Y	N	Y	N	1e-3	0.739
4	xlm-roberta-large	Y	Y	Y	Y	Y	Y	3e-3	0.7273
	Spanish								
1	xlm-roberta-large + flair-es	Y	Y	N	N	N	N	1e-3	0.6031
2	xlm-roberta-large	Y	Y	N	N	N	N	3e-3	0.6505
3	mbert-uncased	Y	N	N	N	N	N	3e-3	0.6724
4	xlm-roberta-large	Y	Y	Y	N	Y[a]	Y	3e-3	0.738
	Dutch								
1	dutchembedding + xlm-roberta-base	Y	Y	Y	N	N	N	3e-3	0.7603
2	xlm-roberta-large	Y	Y	Y	N	N	Y	3e-3	0.7603
3	xlm-roberta-base	Y	Y	Y	N	Y	Y	3e-3	0.7246
	Korean								
1	xlm-roberta-base	N	N	N	N	N	N	3e-3	0.6315
2	xlm-roberta-base	Y	Y	N	N	N	N	3e-3	0.6481
3	xlm-roberta-base	Y	Y	Y	N	Y	N	3e-3	0.6407
4	xlm-roberta-large	Y	Y	Y	N	N	Y	3e-3	0.6729
5	xlm-roberta-large	Y	Y	Y	N	Y	Y	3e-3	0.6688
	German								
1	germanembedding + xlm-roberta-base	Y	Y	Y	N	N	N	3e-3	0.7683
2	xlm-roberta-large	Y	Y	Y	N	N	Y	3e-3	0.7446
3	xlm-roberta-base	Y	Y	Y	N	Y	Y	3e-3	0.7402

[a] additionally translated sentences are used

truth entity label in IOB format. During the training phase for some models in set C, we included both provided training and dev datasets. Thus, we choose to perform the ensemble learning using an augmented dev set created using the data augmentation strategies explained in Section 4. Since the second layer classifier is a CRF layer, we solved the problem of breaking the IOB constraints. By learning to optimally combine the model predictions, EnsembleCRF also learns to avoid mistakes made by single sequence labeling models.

We experimented with creating D_{en} with not only the augmented dev dataset but also the augmented training dataset. However, we found that there is no positive correlation between the number of models in C and the macro-averaged F1 score on the test dataset. Treating every possible combination of set C as a hyperparameter to optimize will yield the optimal result.

4 Training and Evaluation

We have used the flair framework (Akbik et al., 2019) which uses pytorch and huggingface transformers to build and experiment with our approaches. The roberta transformer model was used as the base model. However, for some monolingual language training, we stacked a language-specific embedding layer provided by flair. For Dutch, German, and Spanish, flair language-specific embeddings were prepended before transformer embedding for experimentation.

We experimented with two different initial checkpoints loaded to train the transformer model. One checkpoint was from the huggingface library (hug), for both roberta-base and roberta-large. The other was to load the xlm-roberta model trained on the competition dataset as the initial checkpoint for the training of monolingual tasks. However, the models trained with the initial checkpoint from xlm-roberta performed better due to the transfer of knowledge from the multilingual checkpoint to the monolingual checkpoint. The language models used in the experiment for different languages with hyperparameters and macro averaged F1-score on the competition test set are tabulated in Tables 1 and 2.

4.1 Competition Data

The dataset provided by the competition organizers had 15300 train sentences and 800 dev sentences for each language. However, the entity distribution per language varied (Malmasi et al., 2022a,b). The number of sentences used for training is very less when compared to the number of sentences for testing, which provides the low-resource constraint designed by the competition organizers.

Table 2: The language models used for different languages with the strategies adapted and the hyperparameters used for training the model. The last column shows the macro-averaged F1 score on the competition test set (cont.).

Model No	Language Model	BiLSTM	CRF	Transfer Learning	External Dataset	Aug. Data	Train with Dev	Learning Rate	F1 score
Chinese									
1	bert-base-chinese	Y	Y	N	N	N	Y	3e-5	0.6468
2	bert-base-chinese	N	N	N	N	N	N	3e-3	0.608
3	yechen/ bert-large-chinese	Y	Y	N	N	N	Y	1e-3	0.6237
4	hfl/ chinese-roberta-wwm-ext	Y	Y	N	N	N	Y	1e-3	0.617
5	xlm-roberta-large	Y	Y	Y	N	N	Y	3e-3	0.645
Hindi									
1	monsoon-nlp/hindi-bert	Y	Y	N	N	N	N	3e-3	0.501
2	mbert-cased	Y	Y	N	N	N	N	3e-3	0.493
3	neuralspace-reverie/ indic-transformers-hi-bert	Y	Y	N	N	N	N	3e-2	0.4846
4	indic-distilbert	Y	Y	N	N	N	N	3e-2	0.5087
5	flax-community/roberta-hindi roberta-hindi	Y	Y	N	N	N	N	3e-3	0.2448
Bangla									
1	indic-distilbert	Y	Y	N	N	N	N	1e-3	0.4121
2	xlm-roberta-large	Y	Y	N	N	N	N	3e-3	0.5915
3	xlm-roberta-large	Y	Y	Y	N	N	N	3e-3	0.6019
Multilingual									
1	xlm-roberta-large	N	N	N	N	N	N	3e-3	0.6648
2	xlm-roberta-large	Y	Y	N	N	N	N	3e-3	0.6829
3	xllm-roberta-large	Y	Y	N	N	N	Y	3e-3	0.6924
4	xlm-roberta-large	Y	Y	N	N	Y	Y	3e-3	0.6704

4.2 Data Augmentation

Transformer-based language models require huge amounts of data to produce a good performance, but this requires a lot of labeled data. In the real world, such large labeled datasets are not available easily, especially in some specific domains. We need expert knowledge to annotate the data, which is time-consuming. However, we made use of simple data augmentation techniques for token-level (Dai and Adel, 2020). Here, the method concentrates on expanding the training data using smaller training sets and applying transformations to the training instances without changing their labels. We made use of all the techniques (Dai and Adel, 2020) namely Label-wise token replacement (LwTR), Synonym replacement (SR), Mention replacement (MR), Shuffle within segments (SiS), as well as the mixture of all the techniques to augment the training and development datasets. This produced improvement for a few of the languages even over strong baselines, where no augmentation was used. Although there is no clear single winner, applying all augmentation techniques outperformed single augmentation techniques on an average. We have tabulated the results and their explanation in Section 5.

We also made use of other open-source datasets (Samal, 2021) which were related to different domains like history, political parties, and particularly different from the competition training datasets in context. In addition to this, we made use of `nlpaug` (Ma, 2019) to generate the synthetic data without manual effort. A bert-based model was used to augment the original sentences which were later processed to match the number of token labels. These external datasets did not provide improvements to the performance of the baseline model.

5 Discussion and Results

The training strategies for all the tracks fall into these categories namely external dataset, data augmentations, model architecture searching, transfer learning, and ensemble learning as described in Sections 3 and 4. We used a gazetteer as the last option, which didn't improve the performance.

5.1 English

We trained 12 models using a combination of data augmentation, transfer learning, ensemble learning, and model architecture search. Out of the 12 trained models, we observed that using a multilingual model checkpoint for transfer learning on English data gives better performance on the dev set. We also observed that adding BiLSTM and CRF layer gives slightly better performance than using the linear layer as the classifier. Data augmentation didn't show any difference in the performance on the dev set but the model trained using data augmentation performs better in the test set evaluation.

Table 3: The macro-averaged F1 score for English language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.9122
2	USTC-NELSLIP	0.8547
3	PAI	0.7837
4	ML-HUB	0.7814
5	RACAI	0.7578
6	**Infrrd.ai**	**0.7471**
7	EURECOM	0.7457
8	Sliced	0.7454
9	MaChAmp	0.7448
10	Raccoons	0.7418
11	YNUNLP	0.7317
12	LMN	0.725
13	brotherhood	0.7235
14	L3i	0.7196
15	Multilinguals	0.7174
16	KDDIE	0.7173
17	MarSan_AI	0.7145
18	Cardiff NLP	0.7094
19	Lone Wolf	0.6977
20	MIDAS	0.6962
21	UC3M-PUCPR	0.6924
22	CSECU-DSG	0.6924
23	Sartipi-Sedighin	0.6751
24	Enigma	0.6719
25	DANGNT-SGU	0.6689
26	AaltoNLP	0.6685
27	SPDB Innovation Lab	0.6511
28	silpa_nlp	0.6342
29	BaselineExtending-Pokemons	0.6324
30	MultiCoNER Baseline	0.612
31	AutoNER	0.5572

The final model is an EnsembleCRF model trained with 4 Sequence Labeling models. There was a drop in the F1 score when all 12 models were used. So, we eventually kept the 2 models trained with the dev set and for the rest 10 models, we picked the best 2 models on the dev set. For all models in Table 1, we set the maximum epoch to be 100 with a mini-batch size of 50. We used stochastic gradient descent as the optimizer. We skipped the warmup learning rate as it did not show any improvement on the macro-averaged F1 score of the dev dataset. We observed the training to terminate in around the 15th to 20th epoch. For the EnsembleCRF model, we used Adam optimizer with a learning rate of 1e-3 and a weight decay of 0.01. We set the model to train for a maximum of 100 epochs with a mini-batch size of 126. The results of the proposed method for the English language are tabulated in Table 3.

Table 4: The macro-averaged F1 score for Spanish language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.8994
2	USTC-NELSLIP	0.8544
3	RACAI	0.7562
4	**Infrrd.ai**	**0.7526**
5	MaChAmp	0.752
6	Sliced	0.7511
7	YNUNLP	0.7317
8	brotherhood	0.7069
9	L3i	0.6893
10	PA Ph&Tech	0.6893
11	MarSan_AI	0.683
12	SPDB Innovation Lab	0.6731
13	CSECU-DSG	0.6562
14	EURECOM	0.6277
15	Multilinguals	0.612
16	Sartipi-Sedighin	0.607
17	BaselineExtending-Pokemons	0.6008
18	MultiCoNER Baseline	0.574
19	UC3M-PUCPR	0.5679

Table 5: The macro-averaged F1 score for Dutch language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.905
2	USTC-NELSLIP	0.8767
3	RACAI	0.7841
4	Sliced	0.7766
5	MaChAmp	0.7699
6	**Infrrd.ai**	**0.764**
7	YNUNLP	0.7582
8	brotherhood	0.7304
9	PA Ph&Tech	0.7205
10	MarSan_AI	0.7113
11	L3i	0.7096
12	CSECU-DSG	0.6794
13	EURECOM	0.667
14	BaselineExtending-Pokemons	0.6325
15	MultiCoNER Baseline	0.616
16	Sartipi-Sedighin	0.5837

5.2 Spanish

Since Spanish and English languages have a lexical similarity of about 30–50%, we tried translating the English dataset to Spanish and included it in the model training. Unfortunately, the translation experiment did not help in improving the performance of the model. The final model for the Spanish language included all the 4 techniques of token-level augmented data (Dai and Adel, 2020) along with train and dev datasets. We added up augmented datasets to our training pipeline to provide more exposure and to increase the diversity of available data. We used stochastic gradient descent as the optimizer and trained for about 20 epochs after which the performance turned out to be constant. The results on the test set are tabulated in Table 4.

5.3 Dutch

All the strategies with a language-specific embedding layer were used for experimentation namely roberta-base, roberta-large, and xlm-roberta-large models. The final model is an ensemble CRF model trained with 2 sequence labeling models, an xlm-roberta trained on the Dutch dataset and an xlm-roberta trained on the multilingual dataset. We trained the Dutch model on both the train and dev datasets provided by the competition organizers. A test set is created by splitting the (train and dev) dataset internally while training. The model was trained for 20 epochs. The generated predictions on the test set are propagated through an Ensemble-CRF model to ensure consistency in labeling and to improve the labeling accuracy. The results are tabulated in Table 5.

5.4 Korean

We trained 6 models using a combination of strategies as mentioned in Sections 3 and 4. The final model is an xlm-roberta-large followed by a BiLSTM and CRF while using the multilingual model as an initial checkpoint. The Korean model was trained with the training and dev set provided for 30 epochs with a learning rate of 1e-3 and a weight decay of 0.10. Monte Carlo Dropout (MCD) ensemble (Gal and Ghahramani, 2016) was also used for Korean as an ensemble strategy, 15 different inferences were done with varying architecture with a dropout of 0.3 and a majority voting strategy was used for the final submission. Upon analysis, the best performing model and the entire 15 model ensemble had similar performances. We could potentially cherry-pick models from all the 15 possible candidates to improve the scores but time being a limiting factor, it was dropped. The results are tabulated in Table 6.

Table 6: The macro-averaged F1 score for Korean language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.8859
2	USTC-NELSLIP	0.8636
3	RACAI	0.7174
4	CMB AI Lab	0.707
5	Sliced	0.7066
6	YNUNLP	0.7033
7	C-3PO	0.6749
8	UA-KO	0.6749
9	brotherhood	0.6741
10	**Infrrd.ai**	**0.6729**
11	MaChAmp	0.6545
12	EURECOM	0.6496
13	L3i	0.6268
14	MarSan_AI	0.6226
15	CSECU-DSG	0.6205
16	AaltoNLP	0.6182
17	BaselineExtending-Pokemons	0.5895
18	MultiCoNER Baseline	0.546

Table 7: The macro-averaged F1 score for German language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.9065
2	USTC-NELSLIP	0.8905
3	RACAI	0.7939
4	Sliced	0.789
5	MaChAmp	0.7838
6	YNUNLP	0.7732
7	L3i	0.7723
8	ML-HUB	0.7614
9	brotherhood	0.7594
10	**Infrrd.ai**	**0.759**
11	EURECOM	0.7443
12	MarSan_AI	0.7312
13	CSECU-DSG	0.7249
14	AaltoNLP	0.7137
15	PA Ph&Tech	0.6675
16	BaselineExtending-Pokemons	0.6659
17	MultiCoNER Baseline	0.634

5.5 German

The German language embedding layer was used with roberta-base, roberta-large, and xlm-roberta models, and all the strategies were evaluated. The submitted model is an ensemble CRF model trained with 2 sequence labeling models, an xlm-roberta trained on the German dataset and an xlm-roberta trained on the multilingual dataset. We trained the German model on both the train and dev datasets provided by the competition organizers. A test set is created by splitting the (train and dev) dataset internally while training. The model was trained for 20 epochs. The generated predictions on the test set are propagated through an EnsembleCRF model to ensure consistency in labeling and to improve the labeling accuracy. The results are tabulated in Table 7.

5.6 Chinese

The amount of work carried out on the Chinese dataset is limited due to time limitations. Most of the experiments were limited to model architecture

Table 8: The macro-averaged F1 score for Chinese language sentences.

Position	Team Name	Macro-averaged F1 score
1	USTC-NELSLIP	0.8169
2	CASIA	0.797
3	OPDAI	0.7954
4	DAMO-NLP	0.7806
5	NetEase.AI	0.7777
6	CMB AI Lab	0.7636
7	NCUEE-NLP	0.7418
8	QTrade AI	0.74
9	CSECU-DSG	0.6722
10	Multilinguals	0.6695
11	L3i	0.6691
12	Sliced	0.6521
13	**Infrrd.ai**	**0.6468**
14	MaChAmp	0.6381
15	EURECOM	0.634
16	RACAI	0.627
17	YNUNLP	0.6138
18	brotherhood	0.6086
19	MarSan_AI	0.5664
20	SPDB Innovation Lab	0.5574
21	BaselineExtending-Pokemons	0.528
22	MultiCoNER Baseline	0.511

searching and transfer learning. We tried various pre-trained Chinese language models that include pre-trained Chinese language models trained on other NER datasets. The best architecture observed is a Chinese BERT model followed by BiLSTM and CRF. Our final model was set to train for a maximum of 50 epochs with a mini-batch size of 24. We use AdamW optimizer (Loshchilov and Hutter, 2017) with a weight decay rate of 0.01. We also used Warmup Learning Rate Scheduler with 10% total training steps as a linear warmup period and rest steps with linear decay. The training terminates at the 24th epoch due to an early stopping mechanism. Our final model with the proposed method was trained for 24 epochs, and the results are tabulated in 8.

5.7 Hindi

We tried various Hindi language-based transformer-word-embeddings to include in the flair framework. Word-embeddings like hindi-bert from monsoon-nlp, multilingual-bert-cased, hindi-bert, and distil-bert of indic transformers were evaluated on the dev set. But none of these outperformed our proposed architecture model. We also tried adding language-specific embeddings which would usually help the model better understand the data. But this did not improve our baseline model performance. Hence, we did not include any additional language-specific embeddings. Our final model is xlm-roberta, which was trained using multilingual train and dev datasets. The model was trained for 30 epochs, and the results are tabulated in Table 9.

Table 9: The macro-averaged F1 score for Hindi language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.8623
2	USTC-NELSLIP	0.8464
3	RACAI	0.6808
4	Sliced	0.67
5	NetEase.AI	0.6663
6	**Infrrd.ai**	**0.6572**
7	brotherhood	0.6423
8	YNUNLP	0.6339
9	OPDAI	0.6294
10	MaChAmp	0.6173
11	CSECU-DSG	0.5768
12	MarSan_AI	0.5631
13	EURECOM	0.5278
14	silpa_nlp	0.5149
15	BaselineExtending-Pokemons	0.499
16	L3i	0.4973
17	Enigma	0.4862
18	MultiCoNER Baseline	0.469

Table 10: The macro-averaged F1 score for Bangla language sentences.

Position	Team Name	Macro-averaged F1 score
1	USTC-NELSLIP	0.8424
2	DAMO-NLP	0.8351
3	NetEase.AI	0.7088
4	RACAI	0.6628
5	**Infrrd.ai**	**0.6399**
6	YNUNLP	0.638
7	Sliced	0.6305
8	Team Atreides	0.5975
9	brotherhood	0.5863
10	MaChAmp	0.5646
11	MarSan_AI	0.5422
12	EURECOM	0.5257
13	AaltoNLP	0.5179
14	silpa_nlp	0.5139
15	CSECU-DSG	0.5055
16	BaselineExtending-Pokemons	0.4507
17	L3i	0.4481
18	Enigma	0.4268
19	MultiCoNER Baseline	0.391

5.8 Bangla

It was a challenging task to find good embeddings to represent the Bangla language. We performed a few experiments by including Bangla Indic transformer word embeddings in the flair framework. Similar to the Hindi language, even this embedding did not perform better than our proposed method. We also tried adding language-specific embeddings

Table 11: The macro-averaged F1 score for multiple language sentences.

Position	Team Name	Macro-averaged F1 score
1	DAMO-NLP	0.8531
2	USTC-NELSLIP	0.853
3	QTrade AI	0.7766
4	SeqL	0.7549
5	CMB AI Lab	0.7369
6	UM6P-CS	0.7249
7	RACAI	0.721
8	Cardiff NLP	0.7165
9	Sliced	0.7107
10	IIE_KDSEC	0.7089
11	BaselineExtending-Pokemons	0.7069
12	OPDAI	0.6948
13	brotherhood	0.6942
14	MarSan_AI	0.6928
15	**Infrrd.ai**	**0.6924**
16	HaveNoIdea	0.6879
17	EURECOM	0.6808
18	MaChAmp	0.6768
19	YNUNLP	0.6685
20	DSUG	0.6522
21	UPB	0.6473
22	CSECU-DSG	0.644
23	NSU-AI	0.6423
24	SPDB Innovation Lab	0.6322
25	L3i	0.6123
26	MultiCoNER Baseline	0.541
27	HaveNoIdea	0.5403

which would usually help the model better understand the data. But this did not improve our baseline model performance. Hence, we did not include any additional language-specific embeddings. Our final model is xlm-roberta, which was trained using multilingual train and dev datasets. The model was trained for 40 epochs, and the results are tabulated in Table 10.

5.9 Multilingual

The multilingual task was challenging in itself and our choice of framework made it even harder since we did not have a multi gpu support to conduct all the experiments. The experiments conducted are bucketed mainly into three parts, the architecture search, the data strategy, and the ensemble strategy. After experimenting with various architectures and embeddings, we resorted to xlm-roberta-large for the task. The data strategy was tricky considering all the languages. Open source datasets and translation APIs didn't provide improvements.

We decided to train a stable model and use it as an initial checkpoint for all the other languages. The model was trained for 30 epochs with a learning rate of 1e-3. The final model is xlm-roberta-large followed by a BiLSTM and CRF trained with the entire corpus of train and dev set. Various attempts were made to include the above-mentioned data augmentation techniques but due to the huge model and data along with limited time and resources, we could only do very limited experiments for this model. We tried with the MCD ensemble and took 15 inferences through the varying architectures and used the majority voting strategy to obtain the final submissions. The single best-performing model was at par with the MCD ensemble with majority voting. The results for multiple language sentences are tabulated in Table 11.

6 Conclusion

The recognition of entities from multiple languages with low resources is more complex. The problem lies with the ambiguous entities formed by newly coined words. The syntax of grammar in the sentences was not followed to capture the context between the words. We tried a single multilingual transformer approach, which didn't provide much-expected results. We had used gazetteers for all the languages sourced from the training and wiki data. Both of them didn't produce improvements over the model results.

We trained a multilingual transformer model and performed transfer learning to the individual languages. Since there were multiple languages in the task. We performed unique experiments on one language and then adapted it to the others based on the performance. In the discussion section, the approaches used for experiments are covered and vary for the individual languages. Overall, we created an ensemble of different models which resulted in improvements over the single model. The ensemble architecture covered different types of transformer-based language models. The results reached closer to the top positions with this approach. We also observed that the data augmentation used to improve the performance for a few languages and drop in the performance for the other languages. Our results are above 15% on an average in the participated sub-tasks over the MultiCoNER Baseline results.

We would like to explore the multilingual T5 transformer model, which couldn't be covered during the competition. We would like to explore different augmentation techniques with external data, which couldn't be completed due to time constraints.

References

Hugging face. https://huggingface.co/docs/transformers/index. Accessed: 2022-02-23.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

M Saiful Bari, Shafiq Joty, and Prathyusha Jwalapuram. 2020. Zero-resource cross-lingual named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7415–7423.

Aditi Chaudhary, Jiateng Xie, Zaid Sheikh, Graham Neubig, and Jaime G Carbonell. 2019. A little annotation does a lot of good: A study in bootstrapping low-resource named entity recognizers. *arXiv preprint arXiv:1908.08983*.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner. *arXiv preprint arXiv:2109.00720*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. *arXiv preprint arXiv:2010.11683*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR.

Peter Izsak, Shira Guskin, and Moshe Wasserblat. 2019. Training compact models for low resource entity tagging using pre-trained language models. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 44–47. IEEE.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mónica Marrero, Sonia Sánchez-Cuadrado, Jorge Morato Lara, and George Andreadakis. 2009. Evaluation of named entity extraction systems. *Advances in Computational Linguistics, Research in Computing Science*, 41:47–58.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Tal Perl, Sriram Chaudhury, and Raja Giryes. 2020. Low resource sequence tagging using sentence reconstruction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2692–2698.

Debasis Samal. 2021. Named entity recognition dataset (ner). https://www.kaggle.com/debasisdotcom/name-entity-recognition-ner-dataset. Accessed: 2022-02-23.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003, Edmonton, Canada*, pages 142–147.

Houjin Yu, Xian-Ling Mao, Zewen Chi, Wei Wei, and Heyan Huang. 2020. A robust and domain-adaptive approach for low-resource named entity recognition. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, pages 297–304. IEEE.

UM6P-CS at SemEval-2022 Task 11: Enhancing Multilingual and Code-Mixed Complex Named Entity Recognition via Pseudo Labels using Multilingual Transformer

Abdellah El Mekki[1] **Abdelkader El Mahdaouy**[1] **Mohammed Akallouch**[1],[2]
Ismail Berrada[1] **Ahmed Khoumsi** [3]
[1]School of Computer Sciences, Mohammed VI Polytechnic University, Morocco
[2]Faculty of Sciences Dhar EL Mahraz, Sidi Mohamed Ben Abdellah University, Morocco
[3]Dept. Electrical & Computer Engineering, University of Sherbrooke, Canada
{firstname.lastname}@um6p.ma

Abstract

Building real-world complex Named Entity Recognition (NER) systems is a challenging task. This is due to the complexity and ambiguity of named entities that appear in various contexts such as short input sentences, emerging entities, and complex entities. Besides, real-world queries are mostly malformed, as they can be code-mixed or multilingual, among other scenarios. In this paper, we introduce our submitted system to the Multilingual Complex Named Entity Recognition (MultiCoNER) shared task. We approach the complex NER for multilingual and code-mixed queries, by relying on the contextualized representation provided by the multilingual Transformer XLM-RoBERTa. In addition to the CRF-based token classification layer, we incorporate a span classification loss to recognize named entities spans. Furthermore, we use a self-training mechanism to generate weakly-annotated data from a large unlabeled dataset. Our proposed system is ranked 6th and 8th in the multilingual and code-mixed MultiCoNER's tracks respectively.

1 Introduction

Recent named entity recognition (NER) models have achieved great performance for many languages and using various benchmark datasets such as CoNLL2003 and OntoNotes 5.0 (Devlin et al., 2019). However, it is unclear whether or not these systems can handle ambiguous and complex entities, especially in the case of short and low-context settings (Augenstein et al., 2017). It is also unclear weather these systems can be deployed in real-world scenarios where the input data can be in different languages or code-mixed (Luken et al., 2018; Hanselowski et al., 2018). In fact, to illustrate these issues, if consider the example of complex named entities such as the titles of creative works (movies, songs, books ...), they are hard to be recognized by simple NER systems. This is due to their syntactic ambiguity and the form they can take from one context to another. For instance, they can be as an imperative clause ("Dial M for Murder") or a proposition ("On the beach") which refers to the name of a movie. Thus, it is important to check the performance of NER systems in these scenarios.

The complexity of named entities can be due to three main reasons:

1. **Complex entities:** These entities can be represented as complex infinitives (To Kill a Mockingbird) or full clauses (Mr.Smith Goes to Washington). Additionally, they can be represented as noun phrases or gerunds. State-of-the-art systems (Aguilar et al., 2017) have shown that it is hard to recognize such entities.

2. **Ambiguous entities and contexts:** These types of entities are context-dependent as they can refer to named entities in some contexts, but not in others. “Among Us” which refers to the name of a video game is an example of this challenge. This situation is even more challenging (Mayhew et al., 2019) in the case of short sentences with minimal context such as questions or search queries, which most of the time lack some features such as capitalization or punctuation.

3. **Emerging entities:** This challenge mimics the real-world scenario with many unseen entities, as new named entities are always appearing due to the release of new books, songs, or movies within a short period of time.

It is well known that the state-of-the-art performance reached by current NER systems is mainly due to the presence of easy entities and well-formed input texts (Augenstein et al., 2017). Nevertheless, they yield weak performance when applied in multilingual/code-switched input texts having complex or unseen entities. It is also worth mentioning that the reached state-of-the-art results by current NER systems have been achieved using Transformers-based pre-trained language models (Devlin et al., 2019) that are known to encode the context of input texts and tokens.

In this paper, we introduce our participating system to the MultiCoNER shared task (Malmasi et al., 2022b), in particular, the complex MultiCoNER tracks of multilingual and code-switched queries. Our system relies on a deep learning model based on the multilingual Transformer-based pre-trained language model XLM-RoBERTa (Conneau et al., 2020). To handle the complexity of the two tracks, our model is trained using two optimization objectives, as well as for self-training. The main components of our system can be summarized as follow:

- **Optimization of entities span loss:** We train our model to recognize entities spans as an auxiliary task. The aim is to help the model detect the full entities expressions in an input sentence.

- **Incorporation of Conditional Random Field (CRF) (Lafferty et al., 2001) layer:** We incorporate a CRF layer on top of the Transformer representations to fully exploit the mutual information between tokens in an input sentence.

- **Self-training on unlabeled data:** we create a weakly-supervised data based on the model predictions on unlabeled data.

The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 describes the dataset and the sub-tasks of SemEval-2022 Task 11 (Malmasi et al., 2022b). In section 4, we present our system overview. Section 5 summarizes and discusses the obtained results for both multilingual and code-mixed tracks. Finally, Section 5 concludes the paper.

2 Related Work

During the last years, neural network-based approaches have contributed to improving the performance of NER systems (Panchendrarajan and Amaresan, 2018; Devlin et al., 2019). This was mainly achieved thanks to word embeddings. Static word embeddings are fed to BiLSTM-CRF models and have helped eliminate manual feature engineering while achieving better performance. On the other hand, Transformer-based Language Models (Devlin et al., 2019; Conneau et al., 2020) have greatly improved the NER results thanks to their contextualized word representations.

However, these models may fail when recognizing new or complex entities (Luken et al., 2018; Hanselowski et al., 2018). These challenges are reflecting the real-world setting. Recently, Meng et al. (2021) have proposed an approach to tackle these challenges by using a contextual Gazetteer Representation encoder which can be fused with word-level models. The results have shown that this method enhances the F1-score by +49% in the uncased setting. This work has been mainly applied to the English language. Finally, in another work, Fetahu et al. (2021) have explored the code-mixed NER scenario using multilingual Transformers. They have combined the Mixture-of-Experts model with existing multilingual Transformers models to incorporate the multi-lingual gazetteers. The experiments have demonstrated that their proposed approach enhances the F1-score by +31% in the Code-Mixed NER over the baseline model.

3 Data

The dataset of the MultiCoNER (Multilingual Complex Named Entity Recognition) shared task (Malmasi et al., 2022a) is provided to tackle thirteen different tracks, eleven tracks cover the monolingual cases, while the remaining two tracks cover code-mixed and multilingual data. In this paper, we focus on the last two tracks.

The multilingual track covers eleven languages (English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla), while the code-mixed track covers a subset of these languages. In both tracks, the entities are annotated into six types: PER, LOC, GRP, CORP, PROD, and CW. Table 1 presents the size of the train and test datasets provided for these tracks. We notice that the multilingual track has more data than the code-mixed track, and the test datasets are larger than the train datasets for both tracks.

The provided datasets are labeled using the IOB format which is used for sequence labeling tasks. Table 2 presents the distribution of the entities and the spans in the train datasets for both tracks.

	Multilingual	Code-mixed
Train	168300	1500
Test	471911	100000

Table 1: The size of the train and the test datasets for the multilingual and code-mixed tracks.

4 Methodology

Before discussing the methodology of our proposed system, we assume that we have n labeled sentences with named-entities $D = \{(x_i^t, y_i^t)\}_1^n$ and m unlabeled sentences $U = \{(x_i^u)\}_1^m$, where $x_i = \{x_{i,1}, x_{i,2}, ..., x_{i,l}\}$ represents a sentence containing l tokens, while $y_i = \{y_{i,1}, y_{i,2}, ..., y_{i,l}\}$ are the l labels corresponding to these l tokens.

The proposed system incorporates 4 components used on top of the pre-trained Transformer encoder. In the following, we describe each component of our system.

4.1 Multilingual Transformer encoder

To encode each word in the input sentence, we use the XLM-RoBERTa (XLM-R) (Conneau et al., 2020). It is a multilingual pre-trained transformer encoder network. We choose to use this encoder for the following reasons: 1) XLM-R is the state-of-the-art encoder in the multilingual and code-mixed settings, and 2) It has been trained on 100 languages including the languages covered in the multilingual and code-mixed tracks. This ensures a good contextualized representation for the input sentences despite their language.

This model was mainly trained using the Mask Language Modeling (Devlin et al., 2019). For an input sentence, 15% of the words are randomly masked, then the model tries to predict the masked words. As a result of this training process, the model learns the representations of dimension d for the input words of 100 languages that can be fine-tunned on a downstream tasks such as sequence classification or sequence labeling.

4.2 Span classification module

Span classification is a span-wise classifier, where the aim is to classify whether or not, a sequence of tokens are representing a named entity span based on their semantics. It is a binary classification task as the model predicts **1** if the span is a named entity while it predicts **0** if not.

Given $H = [h_1, h_2, ..., h_k]$, the vector representations of the k sub-tokens contained in a span S, to learn a representation that encodes all the span tokens, we follow the same approach used in (Essefar et al., 2021; El Mekki et al., 2021b; El Mahdaouy et al., 2021), where an attention layer (Bahdanau et al., 2015) learns the span representations SH based on its tokens, as follows:

$$C = tanh(HW^a)$$

$$\alpha = softmax(C^T W^\alpha)$$

$$SH = \alpha \cdot H^T$$

where $W^a \in \mathbb{R}^{d\times1}$, $W^\alpha \in \mathbb{R}^{k\times k}$ are the trainable parameters of the attention layer, $C \in \mathbb{R}^{k\times1}$ and $\alpha \in [0,1]^k$ weights the word representations according to their importance for the task at hand.

The vector representations SH of all the spans in an input sentence are then fed to a feed-forward neural network which classifies whether or not, the input representation refers to a named-entity span.

As illustrated in table 2, we notice that the majority of spans do not represent a named entity. To tackle this imbalanced data issue, we follow (Li et al., 2020) in using the Focal Loss (FL) (Lin et al., 2017). The FL is given by:

$$FL(y, \hat{p}) = -\alpha_y \left(1 - \hat{p}_y\right)^\gamma \log(\hat{p}_y) \qquad (1)$$

where, $y \in \{0,1\}$ denotes the span's label, $\hat{p} = (\hat{p}_0, \hat{p}_1)$ is a vector representing the predicted probability distribution over the labels, α_y is the weight of label y, and γ controls the contribution of high-confidence predictions in the loss. The performed experiments showed that $\gamma = 0.5$ gives the best result.

As the provided dataset in this shared task has been annotated based on named entities, we adjust it for the span classification task. Therefore, we label all the named entities spans as **1** while the rest of spans has been labeled as **0**.

4.3 NER classification using CRF-Layer

Most NER systems using Transformers rely on using the first sub-token of each word as input to the classification layer (Devlin et al., 2019). In our system, we follow the work of (Ács et al., 2021) in using a pooling of the sub-tokens of each word in the input sentence.

Given $H = [h_1, h_2, ..., h_p]$, the vector representations of the all the sub-tokens contained in a word

		Classification tasks							
		Named entity						Entity Span	
	Class	LOC	PER	PROD	GRP	CORP	CW	Entity Span	Not Entity Span
Mutilingual	%	22.02	18.76	15.0	14.0	14.11	16.12	38.82	61.17
Code-Mixed	%	18.27	16.63	17.88	13.9	16.63	16.69	41.71	58.28

Table 2: The distribution of named entities and entities spans in the train dataset for the multilingual and code-mixed tracks

w, an attention layer learns the word pooled representations based on its sub-tokens following the same method explained in section 4.2.

The pooled representations are then fed to the classification layer which is a Conditional Random Field (CRF) layer in our system. We opt for CRF mainly because the Softmax layer does not take into consideration the dependencies between tokens. The self-attention mechanism, performed by the Transformer encoder, encodes these dependencies in the output vectors of the input sentence. While the CRF which is common in sequence labeling tasks ensures output consistency, it transforms the sequence of input word representations to a sequence of probability distributions, therefore, each label prediction depends on the other predictions in the same input sentence.

4.4 Self-training

To take profit from the provided unlabeled dataset in this shared task, we generate a weakly-annotated dataset and re-train the developed model on it. This method has been applied differently in several works (Khalifa et al., 2021; El Mekki et al., 2021a; Huang et al., 2021). In our work, we apply the following pipeline:

1. We train a model M (based on span classification and NER-CRF explained in the previous subsections) using the provided labeled data D.

2. We use the trained model M to predict the labels of the provided unlabeled data. Then we build a weakly-annotated data U

3. We concatenate the datasets T and U and re-train the sequence labeling model.

It is worth mentioning that during the self-training phase, we remove the span classification module.

5 Experiments and results

5.1 Experimental setup

We use the PyTorch framework and the Transformers libraries for the implementation of our proposed system. The training of the model is performed on a server with a single Nvidia Tesla P100 with 16GB of RAM. XLM-RoBERTa Large is used as our multilingual Transformer encoder. Adam optimizer with a learning rate of $1e-5$ is used for all experiments. The system is trained with a batch size of 16 and for 20 epochs.

For the multilingual track, we train our model on the provided labeled multilingual data, then we use the best epoch's model to leverage pseudo-labels from the unlabeled test data, and we re-train the model again from scratch. Besides, for the code-mixed track, we combine the provided data with the training data of the multilingual track and we follow the same training pipeline of the multilingual track.

5.2 Results

Table 3 presents the submitted results for the multilingual and code-mixed tracks using our system. The first row in the table presents the baseline results on the test set published by the shared task organizers. The performance achieved by our best submission largely outperforms the baseline results in both tracks. In fact, the performance using our model boosts the baseline score by 18.39 and 21.11 F1-score points in the multilingual track and the code-mixed track, respectively. The table also reports the performance of our system during the three stages (ablation study):

- The BERT-CRF model that incorporates the named-entity recognition classification layer with CRF,

- The span classification objective, and

- The self-training.

	Multilingual track			Code-mixed track		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Baseline	-	-	54.10	-	-	58.10
BERT-CRF	68.59	69.30	68.00	78.47	78.92	78.78
+ Span Classification	70.71	70.56	70.25	78.30	78.77	78.52
+ Self-training	**73.21**	**72.51**	**72.49**	**79.38**	**79.08**	**79.21**

Table 3: Official Complex NER F1-scores on the multilingual and code-mixed tracks using the proposed system.

	Multilingual track			Code-mixed track		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
LOC	72.24	81.41	76.01	80.37	83.04	81.68
PER	85.20	81.40	83.13	88.02	88.67	88.35
PROD	70.54	70.24	70.00	82.68	80.51	81.58
GRP	69.78	63.6	66.26	70.85	73.02	71.92
CW	67.58	68.74	67.95	75.00	74.27	74.63
CORP	73.91	69.68	71.60	79.36	74.99	77.11

Table 4: Official Complex NER F1-scores per entity on the multilingual and code-mixed tracks using the proposed system.

The results show that the span classification stage enhances the performance of both tracks: the F1-scores achieved using the span classification are 70.25% for multilingual track and 78.52 % for the code-mixed track. However, we notice that the span classification has significantly boosted the F1-score compared to the BERT-CRF model for the multilingual track, while there is a small performance loss in the case of the code-mixed track. When performing self-training on the predictions extracted from the model using the span classification stage, a large gain has been achieved in both tracks. For the multilingual track, the F1-score obtained using self-training is 72.49% with a gain of 3.18% compared to the system without self-training. For the code-mixed track, our system has achieved the F1-score of 79.21% with a gain of 0.87% compared to the system without self-training.

Finally, Table 4 presents the performance of our best submission for the multilingual and code-mixed tracks. The proposed system fails the most in predicting the **GRP** entities for both tracks, meanwhile, it gives its best performance when predicting the **PER** entities.

6 Conclusion

In this paper, we present our Named Entity Recognition (NER) system for complex scenarios on multilingual and code-mixed queries. Our system relies on 4 components: a multilingual transformer encoder, an entity span classification module, a CRF-layer, and a self-training mechanism that leverages information from unlabeled data. We use our system to submit our predictions in the SemEval-2022 Task 11 within the multilingual and code-mixed tracks. The results show that the use of multilingual Transformer and self-training enhances the results in both multilingual and code-mixed cases. Moreover, the incorporation of the span classification module and the CRF layer allow better recognition of complex named entities.

References

Judit Ács, Ákos Kádár, and Andras Kornai. 2021. Subword pooling makes a difference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2284–2295, Online. Association for Computational Linguistics.

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153, Copenhagen, Denmark. Association for Computational Linguistics.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech Language*, 44:61–83.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Abdelkader El Mahdaouy, Abdellah El Mekki, Kabil Essefar, Nabil El Mamoun, Ismail Berrada, and Ahmed Khoumsi. 2021. Deep multi-task model for sarcasm detection and sentiment analysis in Arabic language. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 334–339, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Abdellah El Mekki, Abdelkader El Mahdaouy, Ismail Berrada, and Ahmed Khoumsi. 2021a. Domain adaptation for Arabic cross-domain and cross-dialect sentiment analysis from contextualized word embedding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2824–2837, Online. Association for Computational Linguistics.

Abdellah El Mekki, Abdelkader El Mahdaouy, Kabil Essefar, Nabil El Mamoun, Ismail Berrada, and Ahmed Khoumsi. 2021b. BERT-based multi-task model for country and province level MSA and dialectal Arabic identification. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 271–275, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Kabil Essefar, Abdellah El Mekki, Abdelkader El Mahdaouy, Nabil El Mamoun, and Ismail Berrada. 2021. CS-UM6P at SemEval-2021 task 7: Deep multi-task learning model for detecting and rating humor and offense. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1135–1140, Online. Association for Computational Linguistics.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

Kaiyu Huang, Junpeng Liu, Degen Huang, Deyi Xiong, Zhuang Liu, and Jinsong Su. 2021. Enhancing Chinese word segmentation via pseudo labels for practicability. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4369–4381, Online. Association for Computational Linguistics.

Muhammad Khalifa, Muhammad Abdul-Mageed, and Khaled Shaalan. 2021. Self-training pre-trained language models for zero- and few-shot multi-dialectal Arabic sequence labeling. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 769–782, Online. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online. Association for Computational Linguistics.

T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. 2017. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, Los Alamitos, CA, USA. IEEE Computer Society.

Jackson Luken, Nanjiang Jiang, and Marie-Catherine de Marneffe. 2018. QED: A fact verification system for the FEVER shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 156–160, Brussels, Belgium. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th*

International Workshop on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics.

Stephen Mayhew, Tatiana Tsygankova, and Dan Roth. 2019. ner and pos when nothing is capitalized. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6256–6261, Hong Kong, China. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Rrubaa Panchendrarajan and Aravindh Amaresan. 2018. Bidirectional LSTM-CRF for named entity recognition. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong. Association for Computational Linguistics.

CASIA at SemEval-2022 Task 11: Chinese Named Entity Recognition for Complex and Ambiguous Entities

Jia Fu[1,2], **Zhen Gan**[1,3], **Zhucong Li**[1,2], **Sirui Li**[4], **Dianbo Sui**[1],**Yubo Chen**[1,2], **Kang Liu**[1,2], **Jun Zhao**[1,2]

[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences,Beijing, China
[3] Beijing University of Chemical Technology, Beijing, China
[4] Department of Computer Science, Emory University, Altlanta, GA, USA

{zhucong.li, dianbo.sui,yubo.chen, kliu, jzhao}@nlpr.ia.ac.cn, fujia2021@ia.ac.cn,ganzhen@mail.buct.edu.cn, sirui.li@emory.edu

Abstract

This paper describes our approach to develop a complex named entity recognition system in SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition,Track 9 - Chinese. In this task, we need to identify the entity boundaries and category labels for the six identified categories of CW, LOC, PER, GRP, CORP, and PORD.The task focuses on detecting semantically ambiguous and complex entities in short and low-context settings. We constructed a hybrid system based on Roberta-large model with three training mechanisms and a series of data augmentation. Three training mechanisms include adversarial training, Child-Tuning training, and continued pre-training. The core idea of the hybrid system is to improve the performance of the model in complex environments by introducing more domain knowledge through data augmentation and continuing pre-training domain adaptation of the model. Our proposed method in this paper achieves a macro-F1 of 0.797 on the final test set, ranking second.

1 Introduction

SemEval-2022 Task 11: MultiCoNER Multilingual Complex Named Entities Recognition(Malmasi et al., 2022b). This task aims to address the problem of complex and ambiguous named entities in practical and open domain environments .

The task has 13 tracks, with track 1 being a multilingual track where participants need to train a multilingual model using data from 11 languages. The model should be able to handle monolingual data from any of the languages and code-mixed cases. Tracks 2-12 are monolingual tracks, including English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi and Bengali. Participants are required to train a model for only one language. Track 13 is Code-mixed(Fetahu et al., 2021). This test data contains code-mixed samples. These samples include tokens from any of the 11 mentioned languages in the shared task. We participated in the Chinese monolingual track.

Dealing with complex and ambiguous entities in practical and open domain environments is a challenging NLP task(Meng et al., 2021),however, which has not received sufficient attention from the research community. The challenges of this task are mainly in three facts: 1)The complex entities problem, in which complex noun phrases, verbs, infinitives, or complete sentences, and other entities lack proper nouns, making it difficult to identify them(Ashwini and Choi, 2014).2)The ambiguous entities problem, in which some words are entities in some fields but not in others, especially in search, ASR (Automatic Speech Recognition), and other fields.3)The emerging entities problem, such as books, songs and movies, new works are released every week, and the problem in these fields is that the entities are growing faster, and it is more difficult to identify these newly growing entities.

For the complex entities problem, we use adversarial training and Child-Tuning to increase the representation capability at the parameter level, preventing the model from overfitting due to complex entities, so that the model does not simply remember the complex entities. For the ambiguous entities problem, we use a context-independent data augmentation strategy to replace entities in the data for semantic augmentation, so that the model can reduce the context dependency.For the emerging entities problem, we propose a progressive domain-adaptive pre-training mechanism to improve the performance of the model, so that the preceding methods yielded considerable results, with the F1 value in the Chinese monolingual track reaching 0.797.

2 Related Work

2.1 Named Entity Recognition

Named Entity Recognition is a fundamental problem in natural language processing, and it's a key component of many Natural Language Processing(NLP) activities like information extraction, question-answer systems, syntactic analysis, and machine translation. In general, the aim of named entity identification is to detect three major categories (entity, time, and number) of named entities in the text, as well as seven minor categories (person, institution, location, time, date, currency, and percentage).

NER usually consists of two parts: (1) entity boundary identification and (2) entity category determination. Entity boundaries are easier to identify in English because named entities have more visible indicators (the first letter of each word in the entity should be capitalized), and the work concentrates on establishing entity categories. In contrast to the entity category labeling subtask, the Chinese named entity identification task is more complex, and identifying entity boundaries is more challenging.

2.2 Complex and Ambiguous NER

For complex entities, such as titles of creative works (movie/book/song/software titles), not simple nouns and more difficult to identify. They can be any language component, such as a gerund ("Dial M for Murder"), and they don't appear to be traditional entities (names of people, places, organizations). Because of the syntactic ambiguity, identifying them based on their context is difficult. Finally, these entities increase at a quicker rate than traditional categories.

In datasets like CoNLL03/OntoNots, neural network models (e.g., Transformers) have gotten high scores(Devlin and Ming-Wei Chang, 2019). However these scores are driven by the usage of well-formed news texts, the existence of "easier" entities (e.g., names), and memorization due to entity overlap between training and test sets(Augenstein et al., 2017). These models perform significantly worse on complex/unseen entities. The failure of the NER system to recognize complicated items is responsible for a huge portion of their errors(Hanselowski et al., 2018).

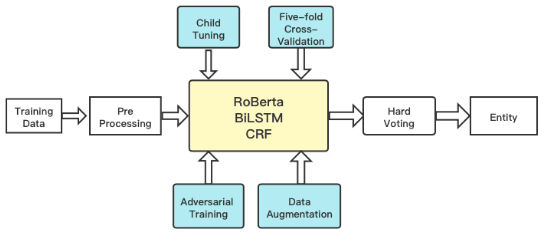


Figure 1: Overall system structure.

3 Our Method

3.1 Overall Approach

The structure of the base model we use is Roberta-wwm-ext-large+BiLSTM+CRF. The sequence samples are pre-trained to obtain their embedding representation, which is then contextually encoded by the BiLSTM(Xu et al., 2017; Ma and Hovy, 2016; Lample et al., 2016) layer, and the contextual encoding is decoded by the CRF(Lafferty et al.; Sutton et al., 2012) to obtain the final annotation result.

In addition, based on the structure of the base model, we first use a back-translation approach to introduce more domain-relevant training data. Related studies have shown that domain-adapted pre-training can improve the performance of the model for the corresponding domain-specific tasks, both with low and high resources. Second, data augmentation is used to extract entities in other languages translated into Chinese and match different contexts to obtain new data, and allow the model to do supervised training using the new data. Finally, we used five-fold cross-training, adversarial training, and child-tuning to improve the performance of the model. Our overall system structure is shown in Figure 1.

3.2 Model Structure

Our basic model structure is shown in Figure 2. The sequence samples get their embedding representation through the pre-training model. Then BiLSTM is connected to the embedding representation for context encoding, and CRF is used to decode the context representation. Finally the annotation result is obtained.

The Bert model, among them, uses Roberta-wwm-ext-large, a joint publication of Harbin Institute of Technology and Pengcheng Lab that uses a full-word Mask scheme in the pre-training phase(Cui et al., 2020). If part of a complete word WordPiece subword is masked, other parts of the

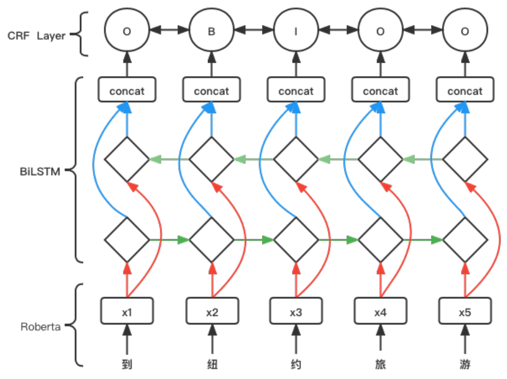


Figure 2: Our basic model structure.

same word will be masked as well, cancels the Next Sentence Prediction, and uses the training model with max len=512.

The underlying model structure is of the form Roberta-wwm-ext-large+BiLSTM+CRF.

3.3 Training Method

Five-fold Cross-voting: We use five-fold cross-validation to divide the training set into five different datasets, and the inconsistencies of entity labeling in each dataset are various. We fix the same model structure, train five models on five training sets, and integrate their prediction results on the same test set by hard voting.

Adversarial Training: To obtain a model with better robustness, we use adversarial training to improve the stability of the model.Referring to the FGM(Miyato et al., 2016) adversarial training mechanism, we directly impose a small disturbance on the embedding representation of the model and assume the embedding representation of the input text sequence $[v_1, v_2, \dots, v_T]$ as x. Then the small disturbance r_{adv} applied each time is:

$$r_{adv} = \epsilon \cdot g/\|g\|_2 \tag{1}$$

$$g = \nabla_x L(\theta, x, y) \tag{2}$$

The meaning of the formulas is to move the input one step further in the direction of rising loss, which will make the model loss rise in the fastest direction, thus forming an attack. In contrast, the model needs to find more robust parameters in the optimization process to deal with attacks against samples.

Among them, applying a small disturbance to the embedding characterization simulates the natural error of the dataset in the labeling to a certain extent. It encourages the model to find more robust parameters during the training process to weaken the impact of aleatoric uncertainty. Then the model's embedding representation will be optimized together with the model. Adversarial training will make the model more tolerant of changes brought about by model parameter fluctuations, thereby decreasing the impact of epistemic uncertainty.

Child-Tuning: In the Fine-tuning process， there is a mismatch between the "high number of parameters" of the large-scale pre-trained model and the "limited number of labeled samples" in the Fine-tuning phase. Child-Tuning proposes, like regular Fine-tuning, using the entire model's parameters to encode the input samples in the forward direction, but without adjusting the huge number of parameters when updating the parameters in the backward direction, i.e. using only a portion of the Child Network. Child-Tuning can be divided into two stages:

- Confirmation of Child Network is found in the pre-trained model, and 0-1Mask of Gradients corresponding to Weights is generated;
- After the gradient is calculated by backward propagation, only the parameters in Child Network are updated, while the other parameters remain unchanged.

Among the above steps,Step 2 is the simplest of the above steps to change the parameters. It's done with a gradient mask, which means that after computing the gradient of each parameter position, it's multiplied by a 0-1 matrix gradient mask, with the positions belonging to the Child Network parameters corresponding to 1 and those not belonging to 0, and the parameters are updated.

The key to this method is to identify the Child Network mentioned in the preceding steps, one of which is the task-independent algorithm Child-Tuning F, whose main advantage is that it is simple and effective; in the Fine-tune process, it only needs to get a Gradients Mask by sampling from the Bernoulli distribution in each update iteration, which is equivalent to randomly discarding part of the gradients when the network parameters are updated.

$$w_{t+1} = w_t - \eta \frac{\partial \zeta(w_t)}{\partial w_t} \odot M_t \tag{3}$$

$$M_t \sim Bernoulli(P_F) \tag{4}$$

Another is the task-related algorithm Child-Tuning-D, which overcomes the disadvantage that Child-Tuning-F treats different downstream tasks with the same policy and treats different model parameters equally.Child-Tuning uses the Fisher Information Matrix (FIM)(Tu et al., 2016) to estimate the importance of each parameter for the downstream task, and, in line with previous work, approximates the diagonal matrix of FIM to calculate the importance score of each parameter relative to the downstream task (i.e., assuming that the parameters are independent of each other), and then selects the parameter with the highest score as the Child-Network.

$$F^(i)(w) = \frac{1}{|D|}\sum_{i=1}^{|D|}(\frac{\partial logp(y_i|x_j;w)}{\partial w^(i))^2} \quad (5)$$

3.4 Data Augmentation

Since the amount of data in the test set is 10 times the amount of data in the training set, the amount of data in the training set is obviously insufficient, so we use data augmentation to get more data. Data augmentation techniques are already standard in the image field, and data augmentation is achieved by techniques such as flipping, rotating, mirroring, and Gaussian white noise on images. However, in the field of NLP, there are four ways of data augmentation: synonym substitution, random insertion, random swapping, and random deletion. In this paper, we use random swapping, i.e., random replacement of entities in a sentence with other entities of the same type. We extract entities in English, German, and Dutch translated into Chinese as replacement entities.

4 Experiment

4.1 Dataset Introduction

SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recog- nition,Track 9 - Chinese provide 15,300 training data, 800 validation sets, and at least 150,000 final test data(Malmasi et al., 2022a). Six types of entities are included: people, places, organizations, products, companies, and creative works.The sentence and character statistics of the training set, development set, and test set are shown in Table 1.

4.2 Evaluation Metrics

This task takes strict macro F1 as the evaluation metric. The macro F1 evaluation metric looks at each entity category equally compared to micro F1, and for the strict F1 evaluation metric, it is considered correct only when both entity boundaries and entity types agree with the standard answer.

Dataset	Type	Train	Dev	Test
Chinese	Sentence	15.3K	0.8K	153K
	Char	382.1K	20K	1835K

Table 1: Statistics of dataset.

4.3 Pre-Processing

Text Expansion: Inevitably there are many combinations of number strings, English and Chinese in Chinese datasets, which are usually done in Chinese ner based on a single character. To maintain uniformity, we expand the English and number strings into a single English letter and a single number.

Labeling Scheme: The annotation scheme is a method for marking character sequences in sequence annotation tasks, by which the type and location of entities in a sentence can be uniquely determined. Moreover, the annotation scheme affects the named entity recognition performance. We use the BIOES annotation scheme which has better performance compared to the BIO annotation scheme.

4.4 Model Parameters

The basic model structure is RoBERTa-wwm-ext-large+BiLSTM-CRF. The batch size is set to 64, the BERT learning rate is set to 1e-5, the BiLSTM+CRF learning rate is set to 1e-3, the training epoch is set to 50, AdamW is used as the optimizer, and a dropout of 0.3 is used.

For the dev set, we used the basic model of RoBERTa-wwm-ext-large+BiLSTM+CRF with two training methods: Child-Tuning and adversarial training.

For the test set, the training data from 10 additional languages were first translated into Chinese using the back-translation approach to do continue pre-training, and then a further pre-training model with 7 epochs and 15 epochs was obtained by changing the training duration.

For the model after continued pre-training, we used data augmentation to extract entities from other languages and translate them into Chinese, and divided the Chinese training set of 15,300 data into five parts, each containing 3,600 training data,

Type	CW	PER	LOC	GRP	CORP	PROD
Num	8732	14851	7610	7250	5220	4354

Table 2: Number of entities for data augmentation.

Model	Precision	Recal	F1
roberta-large	0.8810	0.8510	0.8648
Roberta+fgm	0.8840	0.8600	0.8710
Roberta+child-tuning-F	0.8775	0.8553	0.8656
Roberta+child-tuning-D	0.8722	0.8629	0.8668
Roberta-large+child-tuning-F+fgm	0.8900	0.8424	0.8629
Roberta-large+child-tuning-D+fgm	0.8929	0.8491	0.8686

Table 3: Results on the Chinese dev set.

	(1)	(2)	(3)	(4)	(5)	(6)
7epoch-micro	+	-	+	+	-	+
15epoch-micro	+	-	+	-	+	+
7epoch-macro	+	+	-	-	-	+
15epoch-macro	+	+	-	-	-	+
Data Augmentation	+	+	+	+	+	-
F1	0.7844	0.7835	0.7823	0.7792	0.7812	0.7970

Table 4: Results on the Chinese test set.

	魔	鬼	军	团	博	物	馆
True Table	B-CW	I-CW	I-CW	I-CW	O	O	O
Baseline	O	O	O	O	O	O	O
Other Method	B-GRP	I-GRP	I-GRP	I-GRP	O	O	O
Our Method	B-CW	I-CW	I-CW	I-CW	O	O	O
F1	0.7844	0.7835	0.7823	0.7792	0.7812	0.7970	

Table 5: Case Study.

and replaced the entities in each part with the translated entities, and added the development set to each data augmentation to get the new five-fold data. Since the total number of entities contained in the other 10 languages is large, we selected three languages with high Baseline scores, English, German and Dutch, to do the data augmentation. Since there are more entities appearing in the three languages, we need to do some filtering by downsampling method.We use the down-sampling method, retaining once for entities that appear once and twice for those that appear twice or more. The final number of entities obtained is shown in Table 2.

4.5 Experimental Results and Analysis

We have conducted a large number of experiments locally and the results are shown in Table 3.

The experimental results of the final test set are shown in Table 4. "+" represents the final results using the fifty-fold cross-validation results of the model to participate in hard voting, and "-" represents not using.

Looking at the results of the local experiments and the test set, we found the following two problems:

- **Why does the use of adversarial training and Child-Tuning in local experiments show a drop in scores?** According to our previous experience, adversarial training and Child-Tuning are effective for improving system performance. We believe this is because local experiments use the development set as the test set for validation, and the number of GRP tags in the Chinese development set is very small, resulting in a strong influence of GRP tags on the results and the score drop phenomenon.
- **Why does the score drop when using data augmentation in a test set?** After using the data augmentation method, the score showed a decreasing trend, which we believe is due to the overfitting phenomenon of adding too many entities, resulting in an imbalance between the number of entities and the number of sentences. Later, we'll experiment with changing the amount of data enhanced entities to see if we can improve the model's performance.

5 Case Study

Our model undergoes the above approach and the above challenges are effectively improved. It is able to identify ambiguous and complex entities in shorter contexts.The case study is shown in Table 5. As you can see, our method can easily predict entities like creative work compared to Baseline and other methods.

6 Conclusion and Future Work

To address the challenges of complex entities, ambiguous entities and emerging entities problem, we propose adversarial training and Child-Tuning training methods, context-independent data augmentation strategies, and a progressive domain-adaptive pre-training mechanism to improve the performance of the named entity recognition system.

In the future, we will focus our efforts on strategies and methods to enhance the use of data and hope to make a good progress.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No. 2020AAA0106400), the National Natural Science

Foundation of China (No.62176257，61976211). This work is also supported by the CCF-Tencent Open Research Fund and the Youth Innovation Promotion Association CAS.

References

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv e-prints*, pages arXiv–1408.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. *arXiv preprint arXiv:2004.13922*.

Jacob Devlin and Kristina Toutanova Ming-Wei Chang, Kenton Lee. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv*.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. *arXiv preprint arXiv:1809.01479*.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Charles Sutton, Andrew McCallum, et al. 2012. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373.

Ming Tu, Visar Berisha, Martin Woolf, Jae-sun Seo, and Yu Cao. 2016. Ranking the parameters of deep neural networks using the fisher information. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2647–2651. IEEE.

Kai Xu, Zhanfan Zhou, Tianyong Hao, and Wenyin Liu. 2017. A bidirectional lstm and conditional random fields approach to medical named entity recognition. In *International Conference on Advanced Intelligent Systems and Informatics*, pages 355–365. Springer.

TEAM-Atreides at SemEval-2022 Task 11: On leveraging data augmentation and ensemble to recognize complex Named Entities in Bangla

Nazia Tasnim* and **Istiak Shihab***
Department of CSE, SUST
nimzia.blu@gmail.com
istiak@protonmail.ch

Asif Shahriyar Sushmit
Bengali.Ai
sushmit@bengali.ai

Steven Bethard
School of Information
University of Arizona
Tucson, AZ 85721
bethard@arizona.edu

Farig Sadeque
Department of CSE
BRAC University
Dhaka 1212
farig.sadeque@bracu.ac.bd

Abstract

Biological and healthcare domains, artistic works, and organization names can all have nested, overlapping, discontinuous entity mentions that may be syntactically or semantically ambiguous in practice. Traditional sequence tagging algorithms are unable to recognize these complex mentions because they violate the assumptions upon which sequence tagging schemes are founded. In this paper, we describe our contribution to SemEval 2022 Task 11 on identifying such complex named entities. We leveraged an ensemble of ELECTRA-based models exclusively pretrained on the Bangla language with ELECTRA-based monolingual models pretrained on English to achieve competitive performance. Besides providing a system description, we also present the outcomes of our experiments on architectural decisions, dataset augmentations and post-competition findings.

1 Introduction and Related Works

The task of identifying and classifying entities in text is known as named entity recognition (NER). Some named entities are easy to distinguish in English since each of their words is capitalized; e.g. "The capital of Bangladesh is Dhaka". In this sentence, both "Bangladesh" and "Dhaka" are capitalized named entities. But there are other entity mentions that are not simple nouns and are more difficult to recognize. In the SemEval Task 11: MultiCoNER Multilingual Complex Named Entity Recognition (Malmasi et al., 2022b), the organizers concentrated on the more unusual Named Entities, which can be difficult to identify accurately from the text.

NER tasks have received much attention from the research community due to its crucial role in different NLP problems like information retrieval (Etzioni et al., 2005), Question Answering (Banko et al., 2002) (Toral et al., 2005), Relation extraction, Entity linking (Limsopatham and Collier, 2016) and searching (Pasca, 2004). However, there is such a conceptual difference between an ordinary named entity and a complex named entity that traditional tagging strategies cannot be used to recognize these mentions (Brown et al., 1992). Complex NERs can be any language element (single word, abbreviations, imperative clauses, questions) of ambiguous (Multi-type or Overlapping) and non-regular forms (Nested or Discontinuous or Overlapping) (Ashwini and Choi, 2014). What makes the task more challenging is, Complex NER is part of the open-domain with ever expanding and emerging entity sets and categories.

In recent days, Transformer-based models (Devlin et al., 2018) (Liu et al., 2019) (Yang et al., 2019) have been performing as the state-of-the-art (Yamada et al., 2020) (Yan et al., 2019) models in different NER benchmark datasets. Although, Augenstein and colleagues, demonstrate in their paper that these powerful models are only good at picking up the conventional NERs from well formed texts (Augenstein et al., 2017), while for complex NERs we still need to integrate external knowledge sources. A recent paper on integrating external sources or Gazetteer features in combination with contextual information, has shown that this can indeed improve performance on complex NER tasks (Meng et al., 2021). Gazetteer-based solutions also show good performance improvements in extracting NERs from both normal and code-mixed webqueries (Fetahu et al., 2021).

*These authors contributed equally

In tasks like NER, Bangla NLP has not made significant progress. Many linguistic issues arise while training models on Bangla because it is a rich language in terms of both usability and vocabulary (Ekbal and Bandyopadhyay, 2009). In Bangla, there are few markers for tags, such as capitalization (Karim et al., 2019). The same words can have a variety of meanings and types of entities. In addition, because Bangla is a somewhat free word order language, words can exist in any place inside a phrase without changing their meaning (Ekbal et al., 2008). Affixes that are added to the root word to cause complex inflections can modify the meaning and type of the word as well (Ekbal and Bandyopadhyay, 2009). Despite these issues, transfomer models have been used with considerable success for NER tasks in Bangla (Bhattacharjee et al., 2021) (Ashrafi et al., 2020).

In this work, we demonstrate our approaches in tackling the concerns raised in the SemEval Task 11, as well as the obstacles posed by the Bangla language's intrinsic complexity. In our proposed architecture, we used a variety of methodologies, primarily focusing on transfer-learning with state-of-the-art deep learning architectures. In particular, we submitted the results obtained from monolingual ELECTRA models, while we also ran experiments with non-contextual word embeddings and multilingual language models.

2 Dataset Description

According to the organizers, the data were gathered from Wikipedia and Microsoft Orcas, which included both statements and queries (Malmasi et al., 2022a). The train set contains about 100 domain adaption instances, whereas the test set has significantly more out-of-domain data to measure out-of-domain performance. The test dataset is a large file of 130k+ sentences, with a preset training dataset of 15300 Bangla sentences and a development dataset of 800 sentences. Other important statistics about the dataset is presented in ??. The distribution of NER classes in the training set is shown in figure 1.

To perform the experiments, we augmented our datasets in several stages. At first we token-wise translated a portion of our non-Bangla dataset to Bangla using google translate API[1]. In the first stage, we combined translated Hindi and Farsi dataset with our Bangla dataset, as all three languages come from the Indo-Iranian (Wikipedia contributors, 2022) family. Bangla contains borrowed words from Farsi and it has the same sentence structures as Hindi. In the next step, we combined subsets of translated sentences from all the non-Bangla dataset. This process is repeated for English as well. However, for English, we only combined English, Hindi and Bangla datasets. A summary of our augmented datasets is given in table 2.

[1]https://cloud.google.com/translate

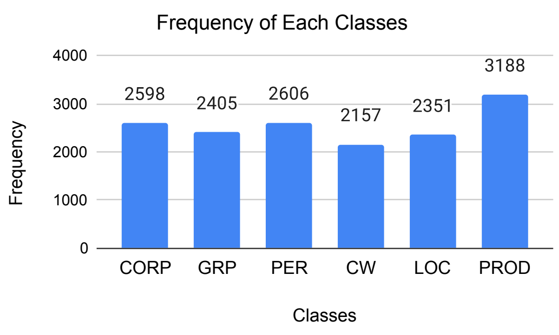


Figure 1: Frequency of each NER Classes

Type	Frequency
Train	15300
Dev	800
Test	133119
Single Word Tokens	4824
Multi Word Tokens	10481

Table 1: Dataset Statistics

3 System Description

The system we proposed for complex Bangla Named Entity Recognition is an ensemble of ELECTRA based models trained on the augmented datasets mentioned in table 2 and a combination of hyperparameters shown in table 3. The representation of each token is fed into our sequence tagging algorithms, which generate a label for each token. The tag of one token is determined by the attributes of that token in context as well as the tag of the token before it. To execute joint inference, these local decisions are connected together.

The implementation of our mono-lingual ELECTRA-based systems can broadly be categorized based on the decision of using non-contextual embeddings (word2vec) with a contextual pre-trained weight (Bhattacharjee et al., 2021). We defined the vanilla token classification system which is largely based on the huggingface token classifi-

Language	Dataset Version	Dataset Constituents	Train Set (Sentences)
Bangla	D1	Bangla	15300
	D2	Bangla + Hindi(tr.) + Farsi(tr.)	21673
	D3	Bangla + All(tr.)	82552
English	D4	Bangla(tr.)	15300
	D5	Bangla(tr.) + Hindi(tr.)	30600
	D6	Bangla(tr.) + Hindi(tr.) + English	45900

Table 2: Default and Agumented Dataset Summary

cation scripts [2], as *S1*. The more advanced NER system incorporating non-contextual embedding and optionally, character CNN (Chiu and Nichols, 2016) and CRF (Qin et al., 2008) is defined as *S2*. Finally, we developed a majority voting based ensemble scheme, *S3*, to obtain our final prediction for each token.

3.1 S1 : Vanilla ELECTRA-based token classification

The input to *S1* is first normalized using a specific normalization pipeline developed for Bangla mentioned in the (Hasan et al., 2020) paper. The normalized data is then tokenized and aligned with labels. *S1* has 12 hidden layers, each with 12 attention heads. A standard training loop, with the hyperparameters mentioned in table 3 is used in different combinations. Since the original huggingface script does not include an early stopping mechanism, we wrote a custom callback based on evaluation loss and a patience of 5. High-level overview of *S1* is shown in figure 2.

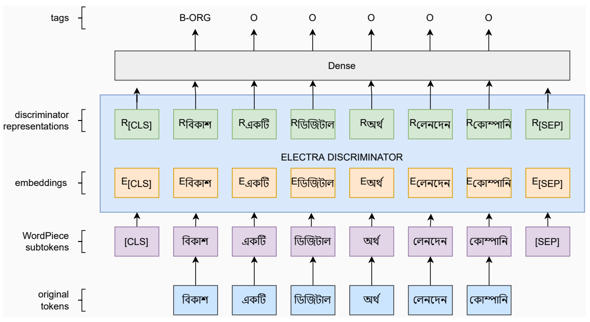


Figure 2: System Overview of *S1*

[2]https://github.com/huggingface/transformers/blob/master/examples/pytorch/token-classification/run_ner.py

System	Settings	
S1	**Tokenizer**	csebuetnlp/banglabert
	Dropout	0.1
	Batch Size	[4,8,16]
	Epoch	[10,20,30]
	Patience	5
	Learning Rate	1.00e-5
	Weight Decay	0.01
S2	**Tokenizer**	csebuetnlp/banglabert
	Dropout	[0.0, 0.1, 0.2]
	LSTM layer	[2, 4]
	Batch Size	[8,16]
	Epoch	[30,40,60,100]
	Patience	[5,7,10]
	Use Character CNN	[True, False]
	Char CNN Kernel Size	[3,6,9]
	Learning Rate	[1e-05, 5e-o5]
	Weight Decay	0.01
	Use CRF Layer	[True, False]
S1.A	**Tokenizer**	google/electra-base-discriminator
	Dropout	0.1
	Batch Size	64
	Epoch	20
	Patience	5
	Learning Rate	1e-4, 1e-5

Table 3: Hyperparameter Settings for *S1 S1.a and S2*

3.1.1 S1.a : Vanilla ELECTRA-based token classification on ENGLISH translated data

As a preprocessing step for this approach, the input dataset was tokenized and translated to english using Google Translate API. The translated input set is then used with the standard huggingface base Electra model with different combination of hyperparameters, as presented in table 3. We experimented with several token-translated language here with early stopping mechanism at patience of 5. The overall architecture is similar to *S1*.

3.2 S2: Advanced NER system

For this system, character and word level features were first extracted and combined with word2vec and ELECTRA embeddings. To generate the final embedding these extracted input features passed through a combination of layers including non-contextual embedding layer, contextual pretrained

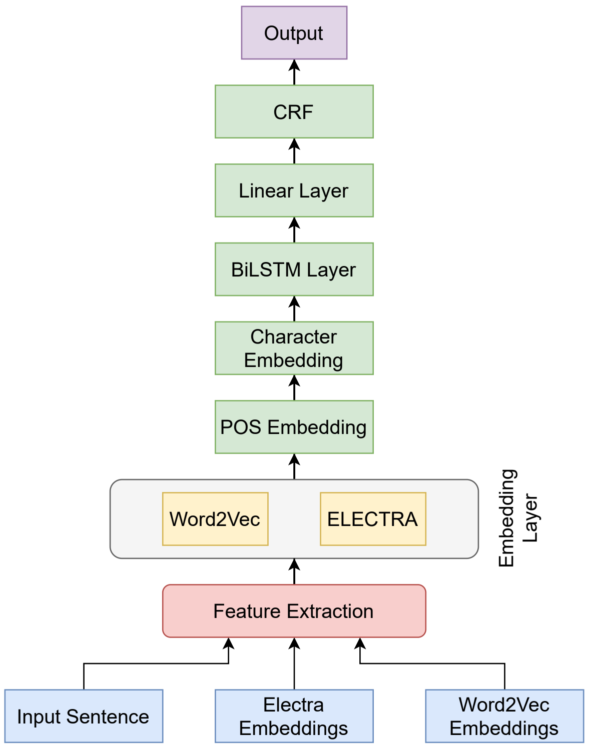


Figure 3: System Overview of *S2*

embedding layer, character embedding layer, parts-of-speech (POS) embedding layer, BiLSTM layer and an additional multi-headed attention(MHA) layer. This is projected through a linear layer and optionally goes through a CRF decoding layer to produce the final predictions. This system also included an early stopping mechanism based on evaluation f1 score. An overview of *S2* is presented in figure 3.

3.3 S3 : Majority Voting Ensemble

The basic concept behind this type of classification is that the final output class is chosen based on the most votes. This ensemble technique has previously been used to overcome the constraints of a single classifier, as presented by the authors in (Siddiqua et al., 2016). Before majority voting, we performed a thresholding on the prediction score for each token from each of the 8 models trained using a variety of augmented datasets, pretrained weights, and hyperparameters. We only considered a token label for majority voting if it had a prediction score over 50%. Then, we counted the number of times the distilled labels appeared in the set. A label was added to the final list of labels if it appeared in the majority of the models. Overview of the *S3* is shown in 4.

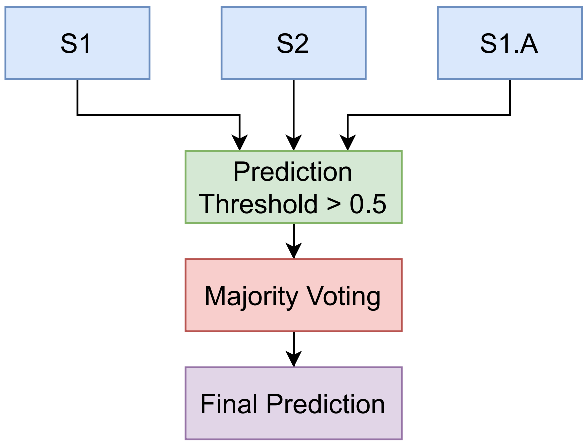


Figure 4: System Overview of *S3*

4 Experimental Setup

As we have previously discussed in section 2, we augmented our training data in multiple steps which extended the dataset several times compared to original. We split each version of these dataset into a 70%-30% ratio during training. The default dev set containing 800 sentences is used for the final validation, in choosing the best performing model during test phase. We employed accuracy, precision, recall, and F1 score as evaluation metrics, with the macro averaged F1 score as the primary and official metric, as per the benchmark of SemEval 2022 Task 11: MultiCoNER (Malmasi et al., 2022b).

We defined each of our best performing model configurations in table 4. While training both *S1* and *S2* we utilized all versions of the Bangla augmented data. Additionally, to train *S1.a* we used all versions of the English translated dataset. In table 3 we have provided the range of hyperparameters used for each of our systems. The performance of these individual models is also demonstrated in table 5. However, in case of the English models, we have only presented the configuration and prediction score for the best performing model. It should be noted that, these models were submitted for evaluation after competition deadline.

5 Results

We made 4 submissions during the test phase, by applying majority voting scheme on various combinations of model predictions. The performance of the final ensemble outputs are presented in 6. As we can observe, the final ensembles of all models performs the highest and it is ranked 8th among all

Model Versions	
M1	S1 + D1 + MHA
M2	S1 + D2
M3	S1 + D4
M4	S2 + D1 + CRF
M5	S2 + D2 + CRF
M6	S2 + D4 + CRF + MHA
M7	S2 + D4 + character CNN
M8	S1 + D6

Table 4: Individual Model Configurations

Attempt	**Precision**	**Recall**	**macro F1**
M1	0.4873	0.3783	0.402
M2	0.6121	0.6053	0.6072
M3	0.5437	0.5135	0.5248
M4	0.5184	0.487	0.4971
M5	0.5433	0.5253	0.526
M6	0.5605	0.5376	0.5431
M7	0.5514	0.5464	0.5472
M8	0.5357	0.5316	0.5333

Table 5: Individual Model Performance Summary

the other teams in Track 11. From the table 7, it is visible that our ensemble model does not perform very well in comparison with the top 3 models and in fact, has a difference of over 20% with the best performing model.

Models	**Precision**	**Recall**	**macro F1**
M1 - M3	0.5924	0.566	0.5768
M4 - M7	0.5926	0.5449	0.5597
M1 - M7	0.5972	0.5578	0.5717
All Models	0.6209	0.5825	0.5975

Table 6: System Ensemble Summary

Team Name	**Score**
USTC-NELSLIP (1st)	0.8424
DAMO-NLP(2nd)	0.8351
NetEase.AI (3rd)	0.7088
Sliced (7th)	0.6305
Team Atreides (8th)	0.5975
brotherhood (9th)	0.5863

Table 7: Leaderboard for Track-11

6 Discussion and Future Directions

From section 5 we see that, there's hardly any difference among the variations of the *S2* models, while major fluctuations can be observed among the variations of *S1* models. Furthermore, separately grouped ensembles of *S1* and *S2* performs almost identically, with the combined ensemble of *S1* and *S2*. However, the performance improves upon including the predictions from *S1.a* models, which are trained on English translated datasets. Despite this, the final best model is clearly overfitting because it had over 80% score on the development dataset, while performing significantly worse (approximately 60%) during the test phase of the competition. This outcome may be attributed to several factors, including the choice of hyperparameters, dataset augmentations and splitting process, early stopping criteria etc. As per the rules of the competition, we only experimented with mono-lingual models to obtain our results. However, we ran the baseline XLM-RoBERTa model which achieves an f1-score of approximately 68% on the development dataset. There are many scopes of expanding this work. For starters, we would like to refine our data augmentation pipeline to generate more well-formed instances. We would explore and compare the performance of cross-lingual and mono-lingual models. We also believe that, the dataset requires further analysis and should receive both quantitative and qualitative error analysis. In addition, we want to do elaborate ablation studies on the components of our systems. In this paper, we have majorly focused on transfer learning and so, in the future, we want to compare the performance of simpler statistical and shallow models with these deep models. Another thing we don't mention empirically in this paper is the class-wise performance of each of our models. From general observation, we find that all the models perform the worst in identifying CW (creative works) tags, while simpler tags like PER (person) and LOC (location) was the easiest to tag. In future, we look forward to investigate more into the reasons behind this behaviors. Finally, we only exploited a simple majority voting based ensemble scheme during this competition. For our future directions, we would also experiment on fusioning the layers of our models to develop a more sophisticated and informed ensembling scheme.

References

Imranul Ashrafi, Muntasir Mohammad, Arani Shawkat Mauree, Galib Md Azraf Nijhum, Redwanul Karim, Nabeel Mohammed, and Sifat Momen. 2020. Banner: a cost-sensitive contextualized model for bangla named entity recognition. *IEEE Access*, 8:58206–58226.

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Michele Banko, Eric Brill, Susan Dumais, and Jimmy Lin. 2002. Askmsr: Question answering using the worldwide web. In *Proceedings of 2002 AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases*, pages 7–9.

Abhik Bhattacharjee, Tahmid Hasan, Kazi Samin, Md Saiful Islam, M Sohel Rahman, Anindya Iqbal, and Rifat Shahriyar. 2021. Banglabert: Combating embedding barrier in multilingual models for low-resource language understanding. *arXiv preprint arXiv:2101.00204*.

Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the association for computational linguistics*, 4:357–370.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Asif Ekbal and Sivaji Bandyopadhyay. 2009. Named entity recognition in bengali: A multi-engine approach. *Northern European Journal of Language Technology*, 1:26–58.

Asif Ekbal, Rejwanul Haque, and Sivaji Bandyopadhyay. 2008. Named entity recognition in bengali: A conditional random field approach. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence*, 165(1):91–134.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Tahmid Hasan, Abhik Bhattacharjee, Kazi Samin, Masum Hasan, Madhusudan Basak, M Sohel Rahman, and Rifat Shahriyar. 2020. Not low-resource anymore: Aligner ensembling, batch filtering, and new datasets for bengali-english machine translation. *arXiv preprint arXiv:2009.09359*.

Redwanul Karim, MA Islam, Sazid Rahman Simanto, Saif Ahmed Chowdhury, Kalyan Roy, Adnan Al Neon, Md Hasan, Adnan Firoze, Rashedur M Rahman, et al. 2019. A step towards information extraction: Named entity recognition in bangla using deep learning. *Journal of Intelligent & Fuzzy Systems*, 37(6):7401–7413.

Nut Limsopatham and Nigel Collier. 2016. Normalising medical concepts in social media texts by learning semantic representation. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 1014–1023.

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

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Marius Pasca. 2004. Acquisition of categorized named entities for web search. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 137–145.

Tao Qin, Tie-Yan Liu, Xu-Dong Zhang, De-Sheng Wang, and Hang Li. 2008. Global ranking using continuous conditional random fields. *Advances in neural information processing systems*, 21.

Umme Aymun Siddiqua, Tanveer Ahsan, and Abu Nowshed Chy. 2016. Combining a rule-based classifier with ensemble of feature sets and machine learning techniques for sentiment analysis on microblog. In *2016 19th international conference on computer and information technology (ICCIT)*, pages 304–309. IEEE.

Antonio Toral, Elisa Noguera, Fernando Llopis, and Rafael Munoz. 2005. Improving question answering using named entity recognition. In *International Conference on Application of Natural Language to Information Systems*, pages 181–191. Springer.

Wikipedia contributors. 2022. Indo-iranian languages — Wikipedia, the free encyclopedia. [Online; accessed 28-February-2022].

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. Tener: adapting transformer encoder for named entity recognition. *arXiv preprint arXiv:1911.04474*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

KDDIE at SemEval-2022 Task 11: Using DeBERTa for Named Entity Recognition

Caleb Martin, Huichen Yang, and **William Hsu**
Computer Science of Kansas State University
Manhattan, Kansas 66502
{calebjm288, huichen, bhsu}@ksu.edu

Abstract

In this work, we introduce our system to the SemEval 2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER) competition. Our team (KDDIE) attempted the sub-task of Named Entity Recognition (NER) for the language of English in the challenge and reported our results. For this task, we use transfer learning method: fine-tuning the pre-trained language models (PLMs) on the competition dataset. Our two approaches are the BERT-based PLMs and PLMs with additional layer such as Condition Random Field. We report our finding and results in this report.

1 Introduction

In today's world there is an ever-growing supply of unstructured data, a lot of it in the form of free text. Named Entity Recognition (NER) is a process that seeks to gather information from free text, by extracting and labeling the named entities. SemEval 2022 Task 11 is a competition where NER systems are trained and then they compete against each other for the best scores (Malmasi et al. (2022b)). The task is split into 13 tracks: one for each of 11 different languages, a multi-lingual track, and a mixed language track (Malmasi et al. (2022b)). In this paper we will be focusing on track 1, which is monolingual English NER. For this SemEval Task participants were asked to have their system both identify named entities and then label them with one of the six provided labels. The six labels to be used were person, location, group, corporation, product, and creative work (Malmasi et al. (2022b)). Figure 1 shows an example of these words being tagged with their appropriate labels.

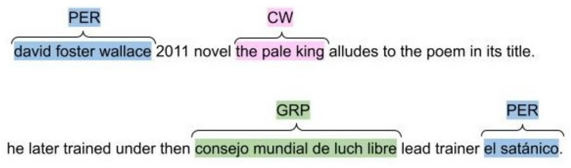

PER
david foster wallace 2011 novel the pale king alludes to the poem in its title.
CW
GRP
he later trained under then consejo mundial de luch libre lead trainer el satánico.
PER

Figure 1: Example of labeling named entities in a sentence using the SemEval 2022 Task 11 tagging scheme. These two sentences are taken directly from the SemEval 2022 Task 11 training set (Malmasi et al. (2022a)).

For this paper our strategy involved trying various transformer models from the HuggingFace library (Wolf et al. (2020)) and training them on the training data provided. We then tried adding a conditional random field layer to some different models to see if that would improve the scores that we received. With all the models we trained we also fine-tuned many different training parameters to obtain the best scores possible. We tried using different number of epochs, different learning rates, different batch sizes, and changing many additional parameters.

In the research we have done leading up to this paper we have learned many things. One is that adding a CRF layer to a BERT model (Devlin et al. (2018)) helps its performance, but adding a CRF layer to a DeBERTa model (He et al. (2021)) doesn't help and actually hurts its performance for NER. We also learned through our testing that a DeBERTa model (He et al. (2021)) is one of the best transformer models for NER, specifically on the SemEval 2022 Task 11 dataset (Malmasi et al. (2022a)).

2 Related Work

There are many challenges that can make NER extremely difficult. In Meng et al. (2021) they explain that named entity recognition is especially difficult in situations with low-context or in scenarios where the named entities are exceptionally complex and unique. Also as stated in Li et al. (2020) NER requires well annotated data and lots of it. This a major obstacle because annotating data can be extremely time consuming and expensive.

Some older approaches to NER include rule-based systems and unsupervised learning systems, however both of these fall short of the performance of modern feature-based supervised methods, which often use transformer models (Li et al. (2020)). According to Li et al. (2020) using pre-trained transformer models with other possible layers and then fine tuning these transformers is becoming the standard for NER. Within this paper we will be using pre-trained transformer models to help us achieve the best results.

Another related work (Souza et al. (2019)) shows the use of a BERT transformer model (Devlin et al. (2018)) for named entity recognition. They also added a conditional random field (CRF) layer on top the BERT model which ended up improving results. This paper (Souza et al. (2019)) shows that both BERT is good for NER and that adding a CRF layer can also help a transformer model perform better.

3 System Overview

For this paper our main strategy will be fine-tuning some large feature-based transformer models on the training data provided. We used the HuggingFace library (Wolf et al. (2020)) to download and train the transformer models. To train our models we used the HuggingFace default optimizer called AdamW. For the most part, we left the AdamW optimizer parameters at the default values. Also, for all of the models we used a linear learning rate scheduler with zero warm up steps.

3.1 BERT

For this competition we experimented with several different systems and methodologies. The first and most basic was to train a BERT model on the data provided (Devlin et al. (2018)). To do this we took a BERT-large-uncased model and trained it for 5 epochs at a learning rate of 2e-5 on the competition training data. With this model we received a precision of 0.870, a recall of 0.811, and a F1-score of 0.839.

3.2 BERT-CRF

Next, we tried improving the score of the BERT model (Devlin et al. (2018)), by adding a conditional random field (CRF) layer after the BERT model. Following the work in (Yang and Hsu (2021)), using the CRF layer should help the model learn the specific parameters of the task, and thus increase the score. We trained this model for 6 epochs at a learning rate of 2e-5 and with a weight decay of 0.01. With this model we received a precision of 0.851, a recall of 0.862, and a F1-score of 0.856.

3.3 DeBERTa-Large

The third model was created using a DeBERTa pre-trained language model. DeBERTa is a BERT (Devlin et al. (2018)) based transformer model that uses disentangled attention and enhanced decoding to improve performance (He et al. (2021)). Within DeBERTa they used a two-vector approach where they split the position encoding and the token encoding into two separate vectors. This allowed the attention layers to be disentangled and learn from each encoding vector separately. They also used enhanced decoding where they give the model both the relative word positions within the sentence and their absolute positions. These improvements allow DeBERTa to outperform BERT in many scenarios (He et al. (2021)). It achieves state of the art scores in many Natural Language Processing tasks including NER (He et al. (2021)).

For this paper we took a DeBERTa model and trained it on the SemEval 2022 Task 11 training data. We got the best results when training the DeBERTa large model for 5 epochs with a learning rate of 2e-5. With this model we received a precision of 0.870, a recall of 0.872, and a F1-score of 0.871.

3.4 DeBERTa-XLarge

This model is similar to the DeBERTa Large except with more parameters. This model has twice the number of layers and parameters. We also got the best results when training the DeBERTa xlarge model for 5 epochs with a learning rate of 2e-5. With this model we received a precision of 0.868, a recall of 0.876, and a F1-score of 0.872.

3.5 DeBERTa-CRF

For this model we decided to follow in the ideas of the BERT-CRF model (Yang and Hsu (2021)) and try to add a CRF layer to our DeBERTa large model (He et al. (2021)). We thought since the CRF layer improved the score of the BERT model that it might do the same for a DeBERTa model. We made a DeBERTa-CRF model and trained it on the task training data for 5 epochs with a starting learning rate of 5e-5 and a weight decay of 0.01. Unfortunately, this model performed quite poorly with a final score precision of 0.820, a recall of

0.830, and a F1-score of 0.825 on the validation dataset (Malmasi et al. (2022a)).

4 Experimental Setup

The data that the organizers provided was in text files in CoNLL format (Malmasi et al. (2022a)). For the English language there was 15300 training examples and 800 validation examples (Malmasi et al. (2022a)). All of the examples provided were in BIO format. BIO is a scheme where all words at the beginning of a named entity are labeled with a B, all the words inside an entity are labeled with a I, and all words outside an entity are labeled with an O. In total that means there are 13 possible tags: O, B/I-PER, B/I-LOC, B/I-GRP, B/I-CORP, B/I-PROD, and B/I-CW (Malmasi et al. (2022b)). The following is the meaning of the abbreviations: PER is person, LOC is location, GRP is group, CORP is corporation, PROD is product, and CW is creative work.

To process that data file first we split the text into each example and then split each example into a list of tokens and labels. Then we mapped each label to a specific number to represent it, 0-12. Finally, from these lists of lists we created a Hugging Face dataset (Wolf et al. (2020)). We left the data sets in the default train/eval splits of 15300 training examples and 800 validation examples.

To evaluate our models, we used the metrics macro precision, recall, and f1-score. Precision deals with how accurate the model is when it does predict a label. Recall corresponds to how good the model is at predicting labels compared to the total number of actual entities. Macro f1-score is the harmonic mean of these two, and gives the best single number representation of how well the model is performing. To calculate these, we used a python library called seqeval (Nakayama (2018)). To use seqeval we simply provide it a list of the predicted values and a list of the ground truth values. The equations seqeval (Nakayama (2018)) uses are shown below with tp being true positives, fp being false positives, and fn being false negatives:

$$Macro\ Precision = \frac{tp}{tp + fp}$$

$$Macro\ Recall = \frac{tp}{tp + fn}$$

$$Macro\ F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

5 Results

Based on the results shown in Table 1 the DeBERTa-XLarge model (He et al. (2021)) performed the best. The BERT-CRF model was significantly better than the BERT model (Devlin et al. (2018)), but it still wasn't better than the DeBERTa models. As expected, the DeBERTa-XLarge model slightly outperformed the DeBERTa-Large model since it had more parameters. The DeBERTa-CRF model performed much worse than we had hoped and was worse than even the base BERT model.

Model	Precision	Recall	F1
BERT	**0.870**	0.811	0.839
BERT-CRF	0.851	0.862	0.856
DeBERTa-Large	**0.870**	0.872	0.871
DeBERTa-XLarge	0.868	**0.876**	**0.872**
DeBERTa-CRF	0.820	0.830	0.825

Table 1: Summary of the scores of all the models tested in this paper. All the scores are from testing on the SemEval 2022 Task 11 validation data set (Malmasi et al. (2022a)).

Interestingly the BERT base model (Devlin et al. (2018)) was tied for the highest precision score even though its recall and F1 scores were relatively low. This means that when it did make a prediction the BERT model was the most accurate at labeling that entity. Out of all the models tried the DeBERTa-XLarge model (He et al. (2021)) ended up as the best scoring model overall and had the highest recall score.

For the official results we used our best model which was the DeBERTa-XLarge model (He et al. (2021)) and made predictions on the provided test data set (Malmasi et al. (2022a)). According to those official SemEval 2022 Task 11 metrics our model had a macro F1 score of 0.717 on the test data. This score put us in 16th place in the competition results.

5.1 Category Results

Table 2 displays the results of our best model, the DeBERTa-XLarge model, on each tag category. Person and location both have high scores with creative work and product being much lower. This makes sense because person and location had the most instances in the training data. Intuitively it also makes sense because person and location are generally simpler entities to identify.

Model	Precision	Recall	F1
Corporation	0.82	0.84	0.83
Creative Work	0.71	0.74	0.73
Group	0.80	0.88	0.84
Location	0.88	0.93	0.91
Person	**0.94**	**0.96**	**0.95**
Product	0.78	0.81	0.80

Table 2: The results of our best model the DeBERTa-XLarge model on each possible label. These scores are on from evaluating on the SemEval 2022 Task 11 validation data set (Malmasi et al. (2022a)).

5.2 Error Analysis

Figure 2 shows a confusion matrix for out best model, the DeBERTa-XLarge model. It shows a couple of key areas where our model is struggling. The matrix shows that while our model is doing well most of the time, there are a couple of labels it commonly confuses. For example it struggles distinguishing between group and corporation. As seen in the matrix when our model predicts B-CORP it is right 88.1% of the time and its most common error is B-GRP which it mistook for B-CORP 4.1% of the time. Logically this makes sense because those labels are quite semantically similar. Interestingly for the entities the model is the worst at predicting (product and creative work), the most common mistake is labeling them PROD or CW when in reality it isn't an entity and is O.

Another surprising error is the number of I or inside entities the model struggles with. It was expected that the model would figure out that the for example an inside entity like I-PER can not follow a start entity of a different type like B-LOC. In an ideal case the model shouldn't mistake an I entity with another I entity, it should just match the B entity. However there are cases in the output of this happening. This would be a good problem to fix in future work.

6 Conclusion

We tried many different types of models for this SemEval 2022 Task 11 competition. We tried training each of these models on the training data, a BERT model, a BERT-CRF model, a DeBERTa-Large model, a DeBERTa-XLarge model, and lastly a DeBERTa-CRF model. In the end the fine-tuned DeBERTa-XLarge model achieved the highest F1-score. We also found out that adding a CRF layer to a DeBERTa pre-trained model didn't help its performance at all.

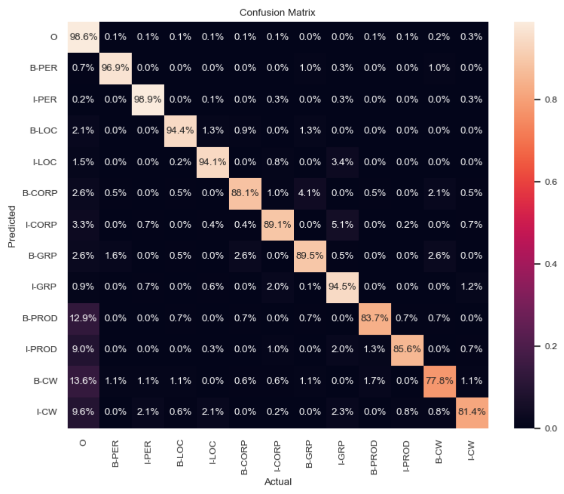


Figure 2: This figure shows a confusion matrix for the different entities that the DeBERTa-XLarge model had to identify. These scores are on from evaluating on the SemEval 2022 Task 11 validation data set (Malmasi et al. (2022a)).

For any future work or improvements on our current work, we would like to try adding some other layers to the DeBERTa model and work on fixing some commmon errors our model has. Since the DeBERTa model seems to be the best at NER, we would like to modify it in some way or modify the data in some way to help it perform better. With further adjustments and experimentation, we believe that the performance of our DeBERTa model could improve.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Fábio Souza, Rodrigo Frassetto Nogueira, and Roberto de Alencar Lotufo. 2019. Portuguese named entity recognition using BERT-CRF. *CoRR*, abs/1909.10649.

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

Huichen Yang and William Hsu. 2021. Named entity recognition from synthesis procedural text in materials science domain with attention-based approach. In *SDU@ AAAI*.

silpa_nlp at SemEval-2022 Tasks 11: Transformer based NER models for Hindi and Bangla languages

Pawankumar Jawale[*] **Sumit Singh** [*] **Uma Shanker Tiwary**
Indian Institute of Information Technology, Allahabad, UP, India
{pawankumar.jawale, sumitrsch}@gmail.com
ust@iiita.ac.in

Abstract

We present Transformer based pretrained models, which are fine-tuned for Named Entity Recognition (NER) task. Our team participated in SemEval-2022 Task 11 MultiCoNER: Multilingual Complex Named Entity Recognition task for Hindi and Bangla. Result comparison of six models (mBERT, IndicBERT, MuRIL (Base), MuRIL (Large), XLM-RoBERTa (Base) and XLM-RoBERTa (Large)) has been performed. It is found that among these models MuRIL (Large) model performs better for both the Hindi and Bangla languages. Its F1-Scores for Hindi and Bangla are 0.69 and 0.59 respectively.

1 Introduction

Named Entity Recognition (NER) is one of the hot topic in natural language processing (NLP). NER is a task of identification of named entities from given sentence and their classification into predefined classes like Person, Location, Organisation, Corporation etc. For below sentence:

राम दिल्ली में गूगल में काम करता है।,
राम is Person, दिल्ली is Location and गूगल is Corporation.

The application of NER can be found in other NLP tasks such as text summarization (Toda and Kataoka, 2005), information retrieval, machine translation (Babych and Hartley, 2003), question-answering (Molla Aliod et al., 2009). The researchers have come up with many approaches for NER task such as Rule-based (Krupka and IsoQuest, 2005), feature-based Supervised approach (Liao and Veeramachaneni, 2009), Unsupervised approach and Deep learning based approach (Li et al., 2020) and Transformer based approach (Vaswani et al., 2017).

The Transformer models are good at capturing features from lengthy sentences compared to recurrent neural networks (Vaswani et al., 2017). RoBERTa model (Liu et al., 2019) performed good for NER task for rich resource languages like English.

For Hindi and Bangla languages, we applied XLM-RoBERTa, which is a multilingual version of RoBERTa pre-trained in 100 languages (including Hindi and Bangla). We also applied IndicBERT (Kakwani et al., 2020) and mBERT (Devlin et al., 2018). At last, we applied MuRIL (Khanuja et al., 2021; Sharma et al., 2022) which is specifically pre-trained in the text of 17 Indic languages and it gave better result than above models for the NER task.

This paper consists of a total of six sections apart from the introduction. Section 2 briefly defines the problem definition and task provided by organizers. Section 3 discusses the work done till now on the NER task. Section 4 mentions the dataset being used in this paper. Section 5 describes the general Transformer architecture for the NER task, along with prepossessing and post-processing. Section 6 discusses the results obtained by used models on both Hindi and Bangla languages and the error analysis. Finally, Section 7 concludes this paperwork.

2 Problem Definition

The organisers (Malmasi et al., 2022b) have arranged 13 tasks according to language. They have provided a separate dataset for each task. Each dataset is comprised of training, development and testing. Respective named entity tags were provided in the training and development dataset. Only tokens were provided in the testing dataset. Participants were required to train the models using the training and development dataset and predict NER tags on the testing dataset. We have worked on Hindi and Bangla tasks.

[*] Authors equally contributed to this work.

3 Related Work

Several works have been done on NER that can be categorized under two broad categories: traditional and deep learning methods.

3.1 Traditional NER approaches

In this approach, feature engineering is carried out by the researchers (Li et al., 2020). Under this category comes the rule-based, feature-based supervised learning, and unsupervised learning approaches.

In the rule-based method, the hand-crafted semantic and syntactic features are provided to recognize the entities (Krupka and IsoQuest, 2005; Aone et al., 1998). These rules-based systems can not be extended to other domains because they depend on domain-specific rules (Appelt et al., 1995).

In the feature-based supervised approach, feature engineering plays a critical role. Features such as word-level features (Liao and Veeramachaneni, 2009; Settles, 2004) and document-level features (Ravin and Wacholder, 1997; Zhu et al., 2005) are used. These features are then passed through supervised models: HMM (Eddy, 1998), Decision trees (Quinlan, 1986), SVM (Hearst et al., 1998) and CRF (Lafferty et al., 2001) for the classification in the labeled corpus.

In the unsupervised approach, the lexical patterns and statistical features are computed, which helps in the clustering (Collins and Singer, 1999). The clustering approach is applied as the data is not labeled in these cases. They extract named entities by making clusters depending on the context similarity (Nadeau et al., 2006).

3.2 Deep Learning NER approaches

As compared to the traditional NER approach, this approach does not explicitly need features. These models automatically extract the hidden features, due to which the accuracies of these models are high compared to the traditional NER approaches. This approach involves the work done using multilevel perceptrons, CNN (Wu et al., 2015), and BiLSTM (Wei et al., 2016; Lin et al., 2017). Recently the Transformer-based models have gained significant advancement in this field (Wolf et al., 2020). The Transformer-based models are good at capturing features in lengthy sentences as compared to recurrent neural networks. The Transformer (Vaswani et al., 2017) is equipped with parallel training and made up of a pair of an encoder and a decoder (to get sequence to sequence prediction). For NER task encoder is used.

In paper (Devlin et al., 2019), the authors have performed NER task on CoNLL-2003 dataset (Tjong Kim Sang and De Meulder, 2003). The authors applied both variants of BERT (Large and Base). The Large variant achieved an F1-score of 92.8 on the test set, whereas the base variant achieved F1-score of 92.4 on the test set. The authors have used BERT as word embedding and fed this to the BiLSTM. XLM-RoBERTa (Conneau et al., 2020) outperform the mBERT (Devlin et al., 2019) and XLM (CONNEAU and Lample, 2019) and show strong improvements over low-resource languages.

4 Data

The dataset (Malmasi et al., 2022a) for Hindi and Bangla contains six different NER entities, namely Location (LOC), Person (PER), Production (PROD), Group (GRP), Corporation (CORP) and Creative Work (CW). The dataset is in standard CONLL format, which uses BIO (Beginning-Inside-Outside) tagging. The dataset provided was of three types, namely training, development and testing. The training and development data contains tokens with tags, whereas testing data contains only tokens. For both Hindi and Bangla tracks, there were 15300 samples in training and 800 samples in development. In the testing dataset, for Hindi and Bangla track there were 141565 (with 933273 total tokens) and 133119

Tag	Training	Development
B-LOC	2614	131
B-PER	2418	133
B-PROD	3077	169
B-GRP	2843	148
B-CORP	2700	134
B-CW	2304	113
I-LOC	1604	77
I-PER	2836	166
I-PROD	2295	107
I-GRP	5821	297
I-CORP	2917	138
I-CW	3592	151
O	209545	10882
Total	244566	12646

Table 1: Entity distribution for Hindi track

(with 693886 total tokens) samples, respectively. Tables 1 and 2 shows the number of each entity in the training and development dataset for Hindi and Bangla, respectively.

Tag	Training	Development
B-LOC	2351	101
B-PER	2606	144
B-PROD	3188	190
B-GRP	2405	118
B-CORP	2598	127
B-CW	2157	120
I-LOC	1453	61
I-PER	3132	180
I-PROD	1964	129
I-GRP	4248	226
I-CORP	2701	122
I-CW	2844	161
O	160250	8654
Total	191897	10333

Table 2: Entity distribution for Bangla track

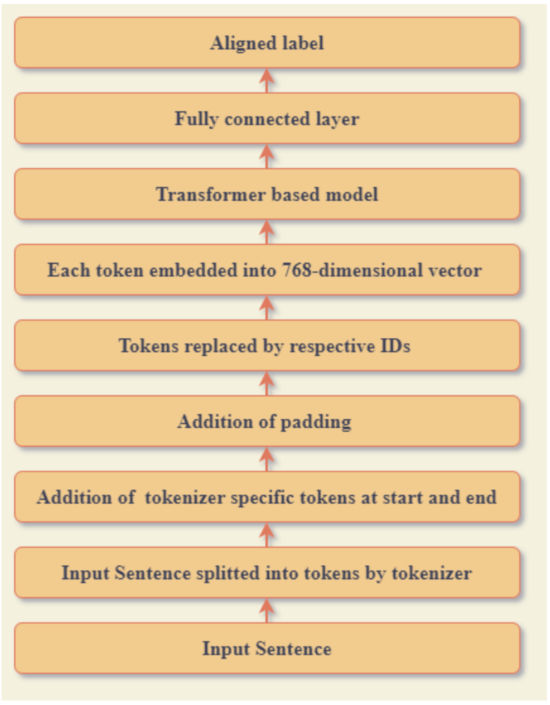
5 Methodology

This work fine tuned 6 Transformer (Vaswani et al., 2017) based pre-trained model for the task. IndicBERT (Kakwani et al., 2020) is Albert based model which is pre-trained on 11 Indic languages, including Hindi and Bangla. We also fine-tuned XLM-RoBERTa (Base) and XLM-RoBERTa (Large) (Conneau et al., 2020), which is pre-trained on text in 100 languages. Other models are mBERT (Devlin et al., 2018) which is pre-trained on text in 104 languages and MuRIL Base and MuRIL Large (Khanuja et al., 2021) which is pre-trained on text in 17 languages with explicitly augmented monolingual text corpora with translated and transliterated document pairs. All the corpora describe above include Hindi and Bangla languages.

Figure 1 shows the architecture of this work, which is divided into 3 sections: Preprocessing, Fine tuning and Post processing.

5.1 Preprocessing

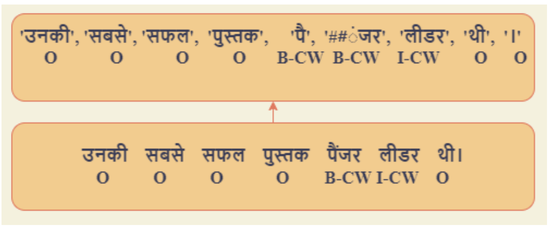
XLM-RoBERTa (Conneau et al., 2020) model and IndicBERT (Kakwani et al., 2020) uses SentencePiece tokeniser (Kudo and Richardson, 2018), which is language independent subword tokeniser and detokeniser. mBERT, MuRIL Base and MuRIL Large model uses WordPiece tokeniser



Aligned label
Fully connected layer
Transformer based model
Each token embedded into 768-dimensional vector
Tokens replaced by respective IDs
Addition of padding
Addition of tokenizer specific tokens at start and end
Input Sentence splitted into tokens by tokenizer
Input Sentence

Figure 1: Generalized transformer-based model

(Wu et al., 2016). As all models use subword tokeniser, any token may get divided into more than one subword. Therefore an alignment of the label is required for that token. Each subword is assigned with the same label as the tokenised word. In figure 2, the token पैंजर is divided by tokeniser into two subwords: 'पै' and 'ंजर', both subwords gets B-CW as their label and the token थी। is divided by tokeniser into two subwords: 'थी' and '।', both subwords gets O as their label. Tokenised sen-



'उनकी', 'सबसे', 'सफल', 'पुस्तक', 'पै', '##ंजर', 'लीडर', 'थी', '।'
O O O O B-CW B-CW I-CW O O
उनकी सबसे सफल पुस्तक पैंजर लीडर थी।
O O O O B-CW I-CW O

Figure 2: Label alignment

tences are added with special tokens along with padding tokens, and thereafter, all the tokens replaced with their ID values for feeding into the models.

5.2 Fine tuning

Architecture in Figure 1 shows that a fully connected layer added on final output hidden vector of the model. This layer takes word embedding corresponding to each token generated by the model (base models generate word embedding of 768 dimension and large models generate word embedding of 1024 dimension) and maps each embedding to the output layer of size (13,), which is the number of unique labels of our task. Further, we calculate loss using the cross-entropy loss function. This model is optimized with Adamw (Loshchilov and Hutter, 2019) and L2 weight decay of 0.01. It is fine-tuned with the dynamic learning rate with linear learning rate scheduler with max learning rate 4e-5, and also, batch size varies from 8 to 64 for different models subject to optimization and a dropout of 0.1 on all layers applied. Maximum token length is taken between 84 to 128 depending on the maximum length of tokenised sentences, which helps in faster training. Number of epochs for training were 30 for all the models in this experiment. We chose best model based on calculated F1-score on valid data. This work predicts the sequence of labels by the argmax of the final layer for each tokens. All models along with the fully-connected layer implemented by XXXForTokenClassification (Wolf et al., 2020), where XXX refers the corresponding model.

5.3 Post processing

After the generation of labels from the model, labels are realigned according to the detokenised sentence. This is reverse of label alignment, discussed in the Preprocessing section. The labels of all the tokens which are first tokens of their original word, are taken as generated labels.

Model	Precision (%)	Recall (%)	F1-Score (%)
M1	47.99	45.77	46.42
M2	51.05	48.08	48.97
M3	62.59	61.49	61.81
M4	**70.06**	**69.07**	**69.08**
M5	47.31	45.98	46.01
M6	51.90	47.90	49.55

Table 3: Results of each model on Hindi test data (M1: mBERT, M2: IndicBERT, M3: MuRIL Base, M4: MuRIL Large, M5: XLM-RoBERTa Base M5: XLM-RoBERTa Large)

6 Results and Analysis

Table 3 and 4 shows the macro average of Precision, Recall and F1-score of each model on testing dataset for Hindi and Bangla respectively.

Model	Precision (%)	Recall (%)	F1-Score (%)
M1	45.28	41.54	42.47
M2	43.40	37.48	38.55
M3	56.98	56.73	56.71
M4	**60.25**	**59.27**	**59.52**
M5	34.75	32.26	33.37
M6	38.74	33.07	35.45

Table 4: Results of each model on Bangla test data (M1: mBERT, M2: IndicBERT, M3: MuRIL Base, M4: MuRIL Large, M5: XLM-RoBERTa Base M5: XLM-RoBERTa Large)

Tables 6 and 7 present the Entity-wise F1 score for Hindi and Bangla testing dataset corresponding to each NER model. It has been found that MuRIL (Large) is showing the highest F1 score for each entity. It has also been observed that the F1 score for CW (Creative work) is the least among all the entities. It indicates that predicting CW is the most difficult for the model.

Sentence	अब तक का सबसे बड़ा बालिका बधू (1976 फ़िल्म)
Gold annotation	[O, O, O, O, O, B-CW, I-CW, I-CW, I-CW]
mBERT	[O, O, O, O, O, O, B-CW, I-CW, I-CW]
IndicBERT	[O, O, O, O, O, O, B-CW, I-CW, I-CW]
MuRIL	[O, O, O, O, O, B-CW, I-CW, I-CW, I-CW]
XLM-RoBERTa Large	[O, O, O, O, O, O, B-CW, I-CW, I-CW]

Table 5: Comparative analysis of a sentence from test corpus

Finally, Table 5 presents the comparative results obtained using different transformer models. Here, the MuRIL output is close to Ground annotation compared to other models.

7 Conclusion

Results show that large models are better than their corresponding base models. MuRIL (Large) model is the best among all six models described above and the second-best model is MuRIL (Base).

Entity	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**
LOC	51.13	52.44	61.52	67.43	49.06	5077
PER	51.50	58.11	71.09	77.86	51.76	5589
PROD	40.57	47.78	59.54	69.01	38.93	4302
GRP	46.74	49.92	63.78	71.48	50.37	5357
CW	**38.83**	**31.64**	**51.49**	**56.95**	**33.97**	**3942**
CORP	49.77	53.95	63.46	71.78	51.96	5466
Avg.	**46.42**	**48.97**	**61.81**	**69.08**	**46.01**	**49.55**

Table 6: Entity-wise F1-score of each model for Hindi dataset (M1: mBERT, M2: IndicBERT, M3: MuRIL Base, M4: MuRIL Large, M5: XLM-RoBERTa Base, M6: XLM-RoBERTa Large)

Entity	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**
LOC	48.91	43.38	54.56	55.93	37.99	38.03
PER	56.35	56.71	74.98	78.21	45.10	48.28
PROD	39.28	40.66	55.03	63.54	37.60	37.53
GRP	35.79	29.62	53.77	**48.03**	23.45	26.75
CW	**30.44**	**22.79**	**40.78**	48.38	**18.11**	**20.97**
CORP	44.09	38.17	61.14	63.04	38.00	41.14
Avg.	**42.47**	**38.55**	**56.71**	**59.52**	**33.37**	**35.45**

Table 7: Entity-wise F1-score of each model for Bangla dataset (M1: mBERT, M2: IndicBERT, M3: MuRIL Base, M4: MuRIL Large, M5: XLM-RoBERTa Base M5: XLM-RoBERTa Large)

Predicting labels corresponding to Creative Work (CW) is most challenging for all the models and predicting labels corresponding to Person (PER) is easier than predicting other labels.

References

Chinatsu Aone, Lauren Halverson, Tom Hampton, and Mila Ramos-Santacruz. 1998. Sra: Description of the ie2 system used for muc-7. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29-May 1, 1998.*

Douglas Appelt, Jerry R Hobbs, John Bear, David Israel, Megumi Kameyama, Andrew Kehler, David Martin, Karen Myers, and Mabry Tyson. 1995. Sri international fastus systemmuc-6 test results and analysis. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995.*

Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT Workshop on MT and Other Language Technology Tools, Improving MT through Other Language Technology Tools: Resources and Tools for Building MT*, EAMT '03, page 18, USA. Association for Computational Linguistics.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *1999 Joint SIGDAT conference on empirical methods in natural language processing and very large corpora.*

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Alexis CONNEAU and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Sean R. Eddy. 1998. Profile hidden markov models. *Bioinformatics (Oxford, England)*, 14(9):755–763.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In *Findings of EMNLP*.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages.

GR Krupka and K IsoQuest. 2005. Description of the nerowl extractor system as used for muc-7. In *Proc. 7th Message Understanding Conf*, pages 21–28.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition.

Wenhui Liao and Sriharsha Veeramachaneni. 2009. A simple semi-supervised algorithm for named entity recognition. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, pages 58–65.

Bill Yuchen Lin, Frank F Xu, Zhiyi Luo, and Kenny Zhu. 2017. Multi-channel bilstm-crf model for emerging named entity recognition in social media. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 160–165.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Diego Molla Aliod, Menno Zaanen, and Daniel Smith. 2009. Named entity recognition for question answering. pages 51–58.

David Nadeau, Peter D Turney, and Stan Matwin. 2006. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Conference of the Canadian society for computational studies of intelligence*, pages 266–277. Springer.

J Quinlan. 1986. Induction of decision trees. mach. learn.

Yael Ravin and Nina Wacholder. 1997. *Extracting names from natural-language text*. Citeseer.

Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications (NLPBA/BioNLP)*, pages 107–110.

Richa Sharma, Sudha Morwal, and Basant Agarwal. 2022. Named entity recognition using neural language model and crf for hindi language. *Computer Speech Language*, 74:101356.

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

Hiroyuki Toda and Ryoji Kataoka. 2005. A search result clustering method using informatively named entities. In *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management*, WIDM '05, page 8186, New York, NY, USA. Association for Computing Machinery.

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

Qikang Wei, Tao Chen, Ruifeng Xu, Yulan He, and Lin Gui. 2016. Disease named entity recognition by combining conditional random fields and bidirectional recurrent neural networks. *Database*, 2016.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Yonghui Wu, Min Jiang, Jianbo Lei, and Hua Xu. 2015. Named entity recognition in chinese clinical text using deep neural network. *Studies in health technology and informatics*, 216:624.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Jianhan Zhu, Victoria Uren, and Enrico Motta. 2005. Espotter: Adaptive named entity recognition for web browsing. In *Biennial Conference on Professional Knowledge Management/Wissensmanagement*, pages 518–529. Springer.

DS4DH at SemEval-2022 Task 11: Multilingual Named Entity Recognition Using an Ensemble of Transformer-based Language Models

Hossein Rouhizadeh
University of Geneva, Switzerland
hossein.rouhizadeh@unige.ch

Douglas Teodoro
University of Geneva, Switzerland
douglas.teodoro@unige.ch

Abstract

In this paper, we describe our proposed method for the SemEval 2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). The goal of this task is to locate and classify named entities in unstructured short complex texts in 11 different languages. After training a variety of contextual language models on the NER dataset, we used an ensemble strategy based on a majority vote to finalize our model. We evaluated our proposed approach on the multilingual NER dataset at SemEval-2022. The ensemble model provided consistent improvements against the individual models on the multilingual track, achieving a macro F1 performance of 65.2%. However, our results were significantly outperformed by the top ranking systems, achieving thus a baseline performance.

1 Introduction

Named entity recognition (NER) is the process of identifying pre-defined categories of named entities, such as people, places, organizations, from unstructured text. NER usually serves as an important first component in various natural language processing (NLP) tasks, such as question answering (Mollá et al., 2006), information retrieval (Guo et al., 2009) and machine translation (Babych and Hartley, 2003). Thus, the performance of the NER system can influence the quality of many downstream NLP applications. Despite the high performance achieved by the current NER systems, they still face some critical challenges(Augenstein et al., 2017). NER models are typically trained on a well-formed news text containing a variety of entities within a relatively long context. In addition, most of the existing NER datasets usually include a large number of common entities between train set and test set. As a result, the performance of the models drops dramatically in the real world applications as they must deal with unseen entities and noisy texts. Furthermore, previous studies on NER have mostly focused on English and as a result, many other languages specially low-resource ones, such as Turkish, Korean, and Persian, have not been as well studied (Rouhizadeh et al., 2021a,b). In this context, SemEval-2022 proposes the task of Multilingual Complex Named Entity Recognition (MultiCoNER) (Malmasi et al., 2022b), which is concerned with detecting semantically ambiguous and complex entities in short and low-contextual settings for 11 languages (i.e. English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla). In this paper, we present a multilingual NER method based on ensemble of deep neural language models. We first trained multiple NER models on the official training dataset and then utilized an ensemble strategy based on a majority of votes from the top-3 best-performing models. Based on the macro-average F1-score of 65.2, achieved by our model, we placed 20th in the multilingual track of the competition.
The rest of the paper is organized as follows. Section 2 reviews published work related to the NER task. Section 3 and section 4 explain our proposed NER system and the experimental setup respectively. The results and detailed analysis of the model performance are discussed in section 5 and the conclusion and future work are reported in section 6.

2 Related Work

Over the last decade, deep learning approaches have significantly improved the results of different NER tasks (Baevski et al., 2019; Akbik et al., 2018). The most recent works on NER utilize pre-trained language models like BERT in a supervised setting (Yamada et al., 2020; Wang et al., 2020; Schneider et al., 2020; Shaffer, 2021). These models use pre-trained language models that have been trained on a large monolingual or multilingual corpus to fine-tune NER models. Meng et al. (2021) introduced a number of current challenges of developed NER datasets and systems. The challenges include the presence of long-tail entities, i.e., entities with large distribution and millions of values, emerging entities, i.e., domains with growing entities, or complex entities, i.e., linguistically complex entities such as gerunds and full clauses, in the context of the systems' inputs. In addition, as discussed in Jayarao et al. (2018) the context of search queries and questions usually include a short amount of words which could be problematic for NER systems. To overcome the above issues, Meng et al. (2021) created three new NER datasets, including short sentences, questions, and search queries, and a novel NER system which uses a contextual gazetteer representation (CGR) encoder and a mixture of experts (MoE) gating network to feed a CRF layer for final predictions. Fetahu et al. (2021) also tackled the challenge of the code-mixed queries in which entities and non-entity query terms coexist simultaneously. They developed a large-scale NER dataset in six languages with four different scripts as well as a novel multi-lingual NER method for code-mixed queries which integrates external knowledge into the multilingual setting.

3 Method

Our multilingual NER system takes sentences in 11 different languages and automatically identifies and classifies named entities within each sentence. For each sentence, the system utilizes three different BERT-like models (fined-tuned on the multilingual NER dataset) to perform entity prediction independently. Next, for each entity, the label with the majority of votes will be chosen as the final prediction. In the following, we provide details on different NER models we used in our pipeline and our ensemble strategy for label prediction in section 3.1 and section 3.2, respectively.

3.1 Training NER Models

To build our NER model, we first fine-tune a number of pre-trained multilingual transformer-based models, i.e., Multilingual-BERT (Pires et al., 2019), XLM-RoBERTa-base, XLM-RoBERTa-Large (Conneau et al., 2019) and Distilbert-Multilingual (Sanh et al., 2019), on the official training dataset (see section 4.1 for more details about the dataset). We fine-tune each particular model by adding (1): a fully connected neural network (FCNN) layer or (2): a conditional random fields (CRF) layer (Lafferty et al., 2001) on the top of the transformer architecture. Transformer-based models usually use the byte-pair encoding for the tokenization. In other words, each token might be divided into more than one sub-token. To deal with this, during training, among the subtokens labels of a given word, the label of the fisrt sub-token has been considered as the label of the word. We also use the BERT-like models to train a simple BiLSTM model with an additional linear classifier on the dataset [1] Following Reimers and Gurevych (2019), we calculate the vector representation for each context word by taking the average of the layer output embeddings of the pre-trained language model and feed them to a BiLSTM neural network as input[2].

As the next step, we select three of the best-performing NER models and use an ensemble strategy (discussed in section 3.2) to finalize our model.

3.2 Ensemble of the NER Models

Having trained multiple NER models, we use an ensemble strategy based on a majority vote to assign the predictions (Copara et al., 2020b,a; Knafou et al., 2020; Naderi et al., 2021). More in detail, for a given sentence S, three NER models infer their predictions independently. Thus, we will have three labeled instances of S associated with several entity labels. Next, for each identified entity, we choose the label that gets the majority of votes (at least two votes) as the final prediction. Note that as we use three different NER models in our pipeline, three different labels might be assigned to a given entity. In such cases, we choose the predicted label of the best-performing model (evaluated on the dev set) as the final prediction.

4 Experimental Setup

This section discusses the dataset we used to conduct our experiments, followed by the parameters we used to train the models.

4.1 Data

Our experiments were conducted using the multilingual dataset provided by the SemEval-2022 Task 11 organizers (Malmasi et al., 2022a). The dataset consists of entity annotated sentences from eleven dif-

[1]We used the code provided by Adelani et al. (2021) to perform BiLSTM experiments.

[2]We only report the results when we feed the BiLSTM with XLM-RoBERTa-large as it performed best compared to the other models

Entity	Train		Dev		Test	
Person	35091	18.4%	8862	18.6%	2342	18.7%
Location	43052	22.6%	10978	23.1%	2932	23.4%
Group	26373	13.8%	6473	13.6%	1638	13.0%
Creative Work	30817	16.2%	7556	15.9%	2015	16.1%
Production	28170	14.8%	6949	14.6%	1848	14.7%
Corporation	26315	13.8%	6575	13.8%	1738	13.8%
All	189818	100%	47393	100%	12513	100%

Table 1: General statistics of the dataset including the number and the distribution of each entity.

Entity / Model	m-BERT	XLM-RoBERTa-base	XLM-RoBERTa-large	m-DistillBERT	BiLSTM	Ensemble
Person	69.2 | 70.8	88.8 | 89.2	90.1 |90.8	83.0 | 82.1	74.3	**91.3**
Location	69.4 | 69.9	86.9 | 87.6	88.0 | 89.3	83.0 | 79.9	75.7	**89.9**
Group	60.7 | 71.1	80.3 | 81.7	84.2 | 85.5	74.0 | 73.4	61.3	**86.2**
Creative Work	58.3 | 59.1	75.0 | 77.4	80.7 | **82.3**	67.0 | 73.2	51.1	81.7
Production	55.0 | 56.6	74.8 | 76.1	79.6 | 80.6	67.0 | 63.5	54.6	**80.6**
Corporation	69.1 | 69.4	82.7 | 83.9	85.5 | 87.1	76.0 | 75.2	61.5	**88.1**
All	63.8 | 64.9	82.5 | 84.0	84.7 | 85.8	75.7 | 75.2	64.2	**86.3**

Table 2: The F1 performance of different multilingual NER models. Each cell include the results when we used a FFCN (the number of the left side) or a CRF layer (the number of the right side) in the model.

ferent languages: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla. The six entity types of the dataset are Person, Location, Production, Corporation, Group, and Creative Work. The organizers provided the competitors with NER-tagged training and development sets, and then released an unlabeled test set for the final prediction. To fine-tune our hyper-parameters and evaluate our models in the development phase, we divided the training set into two parts - 0.80% for the train set and 0.20% for the dev set - and used the official dev set (i.e., provided by the organizers) to test the models and analyse our model results as the labels of the official test set are not released. The number and distribution of occurrences of each entity in the training (train and dev) and test (official dev) datasets are reported in Table 1, where we can notice a relatively good class distribution among the training examples.

4.2 Parameters

In our experiments, we fine-tuned different multilingual pre-trained language models including *bert-base-multilingual-uncased*, *XLM-Roberta-base*, *XLM-Roberta-large*, *distilbert-base-multilingual-cased*, and also trained a simple BiLSTM model on the dataset. We trained each particular model for 6 epochs using Adam optimizer (Kingma and Ba, 2014), a batch size of 16, the learning rate of 2e-5, and the maximum sequence length of 256 tokens. We computed the F1 performance of the model on each epoch and finally saved the parameters of the epoch with the best performance to perform NER on the test set.

5 Results and Discussion

5.1 Results

In Table 2, we show the macro-averaged F1 performance of the NER models on the different entities of the unofficial test dataset. We use the three best performing models identified in the dev set, i.e., *XLM-RoBERTa-large + CRF*, *XLM-RoBERTa-base + CRF* and *XLM-RoBERTa-large + FCNN*, to create our ensemble strategy. As shown in Table 2, the ensemble model outperforms the other single transformer-based models, improving the F1-score of the top-performer models by around 1% point. The results also indicate that the models fine-tuned on the XLM-RoBERTa (both large and base) outperform the other models by a wide margin. In addition, a comparison between the results of each particular model with and without CRF on the test set shows that adding a CRF layer to the models could be helpful as it improves the model performance in most cases. The results show that all models perform best in inferring $Person$ and $Location$ entities. This can be due to the large number of instances of both entities in the training set. In Table 1, it is shown that the number of oc-

Sentence Length	$1 \leq N \leq 5$	$6 \leq N \leq 10$	$11 \leq N \leq 15$	$16 \leq N \leq 20$	$N > 20$	All
Number of Sentences	85	1988	2517	1964	2246	8800
Ratio of the sentences	0.1%	22.5%	28.6%	22.3%	25.5%	100%

Table 3: Number and ratio of sentences with different length (in words) in the test set.

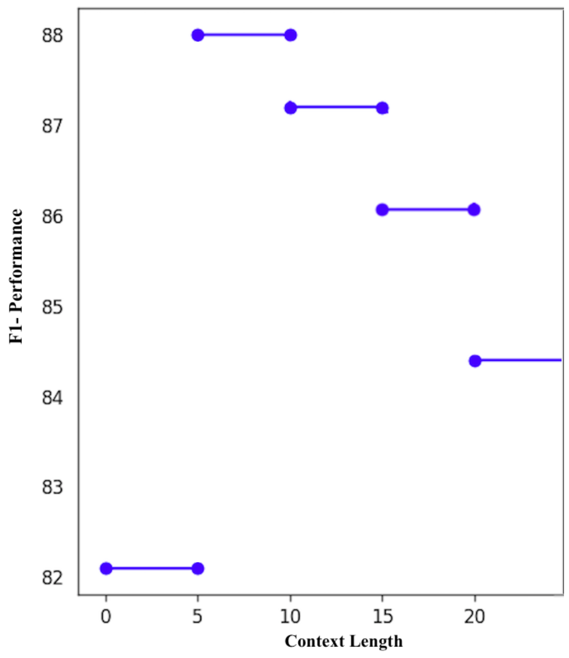


Figure 1: Performance of our ensemble model according to the sentence length (in words).

currences of these entities in the dataset is greater than the other ones. The BiLSTM model also performs significantly worse than the fine-tuned XLM-RoBERTa-large models, despite using the same word vectors.

5.2 Discussion

Effect of the context length One of the most important factors affecting the performance of the NER systems is the context length (Meng et al., 2021). To analyze the effect of the input context on our NER system, we divided the (unofficial) test set into 5 different groups: (1): sentences with five or fewer words, (2): sentences with a context length of at least 6 and less than 11, (3) sentences including at least 10 and less than 15 context words, (4) sentences containing between 15 and 20 words, and (5) sentences containing more than 20 context words. The number and ratio of sentences in each group is reported in Table 3. Figure 1 shows the performance of the ensemble NER model on the different groups of sentences. As it can be seen, the model has the worse performance when the sentences contain 5 or less words. Surprisingly, the model performs best in the second group (sentences containing between 5 and 10 words) showing the strength of the model even in the short the sentences.

6 Conclusion

In this paper, we presented our multilingual NER method that uses an ensemble of different fine-tuned models to identify the named entities in the unstructured texts. Using a variety of multilingual pre-trained language models, we first fine-tuned several NER models and then applied a vote-based ensemble strategy to make the final prediction. Our submission achieved an overall F1 score of 65.2, ranking 20th in the multilingual track of task 11 of SemEval-2022. Our next step would be to examine other possible types of ensemble strategies as it has shown to be effective in the performance of the NER models.

References

David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, Shamsuddeen H. Muhammad, Chris Chinenye Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Aremu Anuoluwapo, Catherine Gitau, Derguene Mbaye, Jesujoba Alabi, Seid Muhie Yimam, Tajuddeen Rabiu Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, Verrah Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, Chiamaka Chukwuneke, Nkiruka Odu, Eric Peter Wairagala, Samuel Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane MBOUP, Dibora Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima DIOP, Abdoulaye Diallo, Adewale Akinfaderin, Tendai Marengereke, and Salomey Osei. 2021. MasakhaNER: Named Entity Recognition for African Languages. *Transactions of the Association for Computational Linguistics*, 9:1116–1131.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018.

Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics*, pages 1638–1649.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other language technology tools, Improving MT through other language technology tools, Resource and tools for building MT at EACL 2003*.

Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. Cloze-driven pretraining of self-attention networks. *arXiv preprint arXiv:1903.07785*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jenny Copara, Julien Knafou, Nona Naderi, Claudia Moro, Patrick Ruch, and Douglas Teodoro. 2020a. Contextualized french language models for biomedical named entity recognition. In *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Atelier DÉfi Fouille de Textes*, pages 36–48.

Jenny Copara, Nona Naderi, Julien Knafou, Patrick Ruch, and Douglas Teodoro. 2020b. Named entity recognition in chemical patents using ensemble of contextual language models. *arXiv preprint arXiv:2007.12569*.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 267–274.

Pratik Jayarao, Chirag Jain, and Aman Srivastava. 2018. Exploring the importance of context and embeddings in neural ner models for task-oriented dialogue systems. *arXiv preprint arXiv:1812.02370*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Julien Knafou, Nona Naderi, Jenny Copara, Douglas Teodoro, and Patrick Ruch. 2020. Bitem at wnut 2020 shared task-1: Named entity recognition over wet lab protocols using an ensemble of contextual language models. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 305–313.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Diego Mollá, Menno Van Zaanen, and Daniel Smith. 2006. Named entity recognition for question answering. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 51–58.

Nona Naderi, Julien Knafou, Jenny Copara, Patrick Ruch, and Douglas Teodoro. 2021. Ensemble of deep masked language models for effective named entity recognition in health and life science corpora. *Frontiers in research metrics and analytics*, 6.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Hossein Rouhizadeh, Mehrnoush Shamsfard, Mahdi Dehghan, and Masoud Rouhizadeh. 2021a. Persian semcor: A bag of word sense annotated corpus for the persian language. In *Proceedings of the 11th Global Wordnet Conference*, pages 147–156.

Hossein Rouhizadeh, Mehrnoush Shamsfard, Vahideh Tajalli, and Masoud Rouhziadeh. 2021b. Persian-wsd-corpus: A sense annotated corpus for persian all-words word sense disambiguation. *arXiv preprint arXiv:2107.01540*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Elisa Terumi Rubel Schneider, Joao Vitor Andrioli de Souza, Julien Knafou, Lucas Emanuel Silva e Oliveira, Jenny Copara, Yohan Bonescki Gumiel, Lucas Ferro Antunes de Oliveira, Emerson Cabrera Paraiso, Douglas Teodoro, and Cláudia Maria Cabral Moro Barra. 2020. Biobertpt-a portuguese neural language model for clinical named entity recognition. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 65–72.

Kyle Shaffer. 2021. Language clustering for multilingual named entity recognition. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 40–45.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*.

CSECU-DSG at SemEval-2022 Task 11: Identifying the Multilingual Complex Named Entity in Text Using Stacked Embeddings and Transformer based Approach

Abdul Aziz, Md. Akram Hossain, and **Abu Nowshed Chy**
Department of Computer Science and Engineering
University of Chittagong, Chattogram-4331, Bangladesh
{aziz.abdul.cu, akram.hossain.cse.cu}@gmail.com,
and nowshed@cu.ac.bd

Abstract

Recognizing complex and ambiguous named entities (NEs) is one of the formidable tasks in the NLP domain. However, the diversity of linguistic constituents, syntactic structure, semantic ambiguity as well as differences from traditional NEs make it challenging to identify the complex NEs. To address these challenges, SemEval-2022 Task 11 introduced a shared task MultiCoNER focusing on complex named entity recognition in multilingual settings. This paper presents our participation in this task where we propose two different approaches including a BiLSTM-CRF model with stacked-embedding strategy and a transformer-based approach. Our proposed method achieved competitive performance among the participants' methods in a few languages.

1 Introduction

Named entity recognition (NER) is a popular sequence labeling task in the natural language processing (NLP) arena. It has numerous applications in several computational linguistic tasks including designing efficient search systems, data mining, and document indexing. However, prior studies mostly focused on identifying traditional named entities (NEs) recognition e.g. person names, locations, and organizations names (Murthy et al., 2018).

The ever-growing generation of unstructured social media data contains a huge amount of complex (Meng et al., 2021) and ambiguous(Fetahu et al., 2021) named entities. This is because social media data is severely induced by noise as well as the linguistic constituent, syntactic and semantic ambiguity exists in this data source. Besides, the social media data mostly have multilingual data. Therefore it poses new challenges to the traditional named entity recognizer system (Aguilar et al., 2019; Ashwini and Choi, 2014). To address the challenges of recognizing such complex and ambiguous named entities in multilingual settings, (Malmasi et al., 2022b) introduced a shared task at SemEval-2022 named as MultiCoNER. The task is composed of three category of tracks including multi-lingual, mono-lingual, and code-mixed tracks. A multilingual dataset (Malmasi et al., 2022a) containing data from 11 languages is used to assess the participants' system. To illustrate a clear view of the task definition, we articulate two examples from English languages and corresponding labels in Table 1.

English
Text#1: Adaptation of seinen series by kenichi sonoda. **Tag:** [O, O, B-CW, O, O, B-PER, I-PER, O]
Text#2: It was designed by kohn pedersen fox. **Tag:** [O, O, O, O, B-CORP, I-CORP, I-CORP, O]

Table 1: Data sample.

We articulate the rest of the contents as follows: Section 2 describes our proposed approach whereas, in Section 3, we present our experimental setup and conduct performance analysis against the various settings and participants' methods. Finally, we conclude our work with some future directions in Section 4.

2 Proposed Framework

In this section, we describe our proposed approach for the MultiCoNER shared task. Our goal is to identify the complex and ambiguous named entities in multilingual settings. The task is articulated into multi-lingual, mono-lingual, and code-mixed tracks. To address the task challenges, we employ two different approaches including a BiLSTM-CRF

**The first two authors have equal contributions.

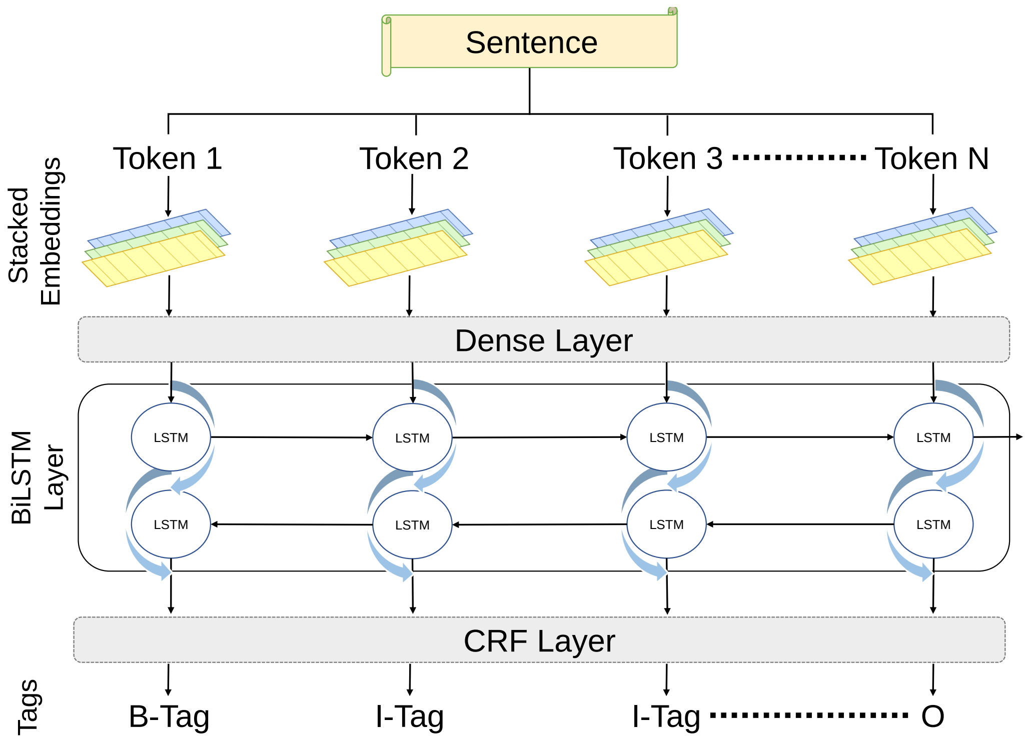


Figure 1: Overview of our proposed BiLSTM-CRF based framework.

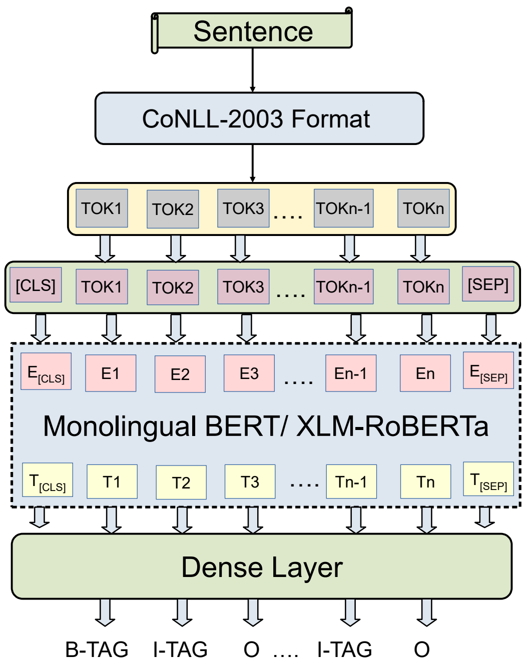


Figure 2: Overview of our proposed transformer-based NER framework.

based approach and a transformer-based approach. The overview diagrams of these approaches are depicted in Figure 1 and Figure 2, respectively. In the BiLSTM-CRF model, we employ the Flair's (Akbik et al., 2019) implementation of stacked embedding technique for effective word representation. Whereas in our transformer-based approach, we employ the available monolingual BERT for each of the languages. However, for some of the languages and for the multilingual and code-mixed settings, we employ the XLM-RoBERTa.

2.1 BiLSTM-CRF with Stacked Embedding

The BiLSTM-CRF model is well-known for the named entity recognition (NER) task. For the training purpose, we use the sentence represented as CoNLL -U format containing BIO tag for each token. A token tagged as O means it is not part of an entity, B-X denotes the first token of an X entity, I-X denotes the token is within the X type entity having multiple tokens. The tokens are then sent to the embedding layer. We employ Flair's (Akbik et al., 2019) implementation of stacked embedding strategy that concatenates embedding vectors of different models together for the effective representation of tokens. To choose the optimal ones in the stacking, we explore various embedding models. The list of embedding models used in our stacked-embedding approach is presented in Table 2. The embedding vectors are concatenated and send to the BiLSTM encoder to distill the contextual dimension of each token. The BiLSTM encoder is followed by a linear-chain conditional random fields (CRF) classifier that generate predictions with the

Track	Word Embeddings Used in Stack Embedding Model (BiLSTM-CRF based System)	Used Transformer Model (Transformer based System)
English (en)	Globe, News-forward, News-backward (Akbik et al., 2018)	bert-base-uncased (Devlin et al., 2019)
Spanish (es)	FastText (es), es-forward, es-backward	dccuchile/bert-base-spanish-wwm-cased (Cañete et al., 2020)
Dutch (nl)	FastText (nl), nl-forward, nl-backward	GroNLP/bert-base-dutch-cased (de Vries et al., 2019)
Russian (ru)	FastText (ru), Byte Pair Embedding (ru), Character Embedding	DeepPavlov/rubert-base-cased-sentence
Turkish (tr)	FastText (tr), Byte Pair Embedding (tr)	dbmdz/bert-base-turkish-128k-cased
Korean (ko)	FastText (ko), Byte Pair Embedding (ko), Character Embedding	klue/bert-base (Park et al., 2021)
Farsi (fa)	FastText (fa), fa-forward, fa-backward	HooshvareLab/bert-fa-base-uncased (Farahani et al., 2020)
German (de)	FastText (de), de-forward, de-backward	-
Chinese (zh)	FastText (zh), Byte Pair Embedding (zh), character Embedding	bert-base-chinese
Hindi (hi)	FastText (hi), hi-forward, hi-backward	xlm-roberta-base
Bangla (bn)	Byte Pair Embedding (bn), Character Embedding	xlm-roberta-base
Multi-linguall (multi)	Byte Pair Embedding (multi), multi-Forward, multi-Backward	xlm-roberta-base
Code-mixed (mix)	-	xlm-roberta-base

Table 2: Used word embeddings and transformer models in our CSECU-DSG system.

BIO tagging scheme.

2.2 Transformer based System

In our transformer-based approach, we employ the monolingual BERT model for each of the languages that are available in the Huggingface repository (Wolf et al., 2019). To choose the optimal transformers, we explore various embedding models. However, for some of the languages, XLM-RoBERTa performed better compared to the monolingual BERT. In such cases and also for the code-mixed and multilingual data, we employ the XLM-RoBERTa model.

The Facebook AI launched the XLM-RoBERTa as an upgrade to their initial XLM-100 model (Conneau et al., 2020). It is a scaled cross-lingual sentence encoder. Using self-supervised training approaches, it offers state-of-the-art performances in cross-lingual understanding where a model is taught in one language and then applied to multiple languages with no additional training data. This model showed increased performance on numerous

NLP applications. Utilizing just monolingual data, XLM-RoBERTa was trained with a multilingual masked language model (MLM) objective.

With users publishing content in over 160 languages on Facebook, XLM-RoBERTa is a major leap toward the goal of offering the greatest possible experience on this platform for everybody, regardless of their native language. XLM-RoBERTa creates the possibility for a one-model-for-many-languages approach rather than a single model per language. There are two versions of XLM-RoBERTa. The base version of XLM-RoBERTa contains 250M parameters, whereas the large version has 560M. The vocabulary in both versions is 250K. In our framework, we use the XLM-RoBERTa-base version to extract the effective transfer learning features. The list of transformer models used in our transformer-based approach is presented in Table 2.

3 Experiment and Evaluation

3.1 Dataset Description

The organizers of the SemEval-2022 MultiCoNER shared task 11 (Malmasi et al., 2022b,a) provided a benchmark dataset to evaluate the performance of the participants' systems. The dataset comprises data of 11 languages along with a code-mixed dataset. The dataset statistics are summarized in Table 3.

Track	Training	validation	Test
BN-Bangla	15,300	800	133,119
DE-German	15,300	800	217,824
EN-English	15,300	800	217,818
ES-Spanish	15,300	800	217,887
FA-Farsi	15,300	800	165,702
HI-Hindi	15,300	800	141,565
KO-Korean	15,300	800	178,249
NL-Dutch	15,300	800	217,337
RU-Russian	15,300	800	217,501
TR-Turkish	15,300	800	136,935
ZH-Chinese	15,300	800	151,661
MULTI-Multilingual	168,300	8,800	471,911
MIX-Code_mixed	1500	500	100000
Total	338,100	18,100	2,567,509

Table 3: The statistics of the datasets used in different lingual track.

3.2 Experimental Settings

We now describe the the set of parameters that we have used to design our proposed CSECU-DSG systems for the MultiCoNER task. We employ two different approaches including a BiLSTM-CRF based approach and a transformer-based approach. We employ Flair's (Akbik et al., 2019) implementation of the BiLSTM-CRF approach and Huggingface (Wolf et al., 2019) implementation of the monolingual and multilingual transformer models with fine-tuning. We finetune these models with the provided training data. We also tune several hyper-parameters to obtain the optimal performances. Our hyper-parameters search space is presented in Table 4 and default settings are used for the others.

Hyper-parameters	Search Space
Training batch size	$\{8, 16, 32\}$
Learning rate	$\{0.1, 1e-3, 1e-5, ..., 3e-6\}$
Number of epochs	$\{3, 5, ..., 100\}$
BiLSTM output size	$\{64, 128, 256, 512\}$

Table 4: The hyper-parameters search space.

3.3 Evaluation Measures

To assess the performance of the participants' systems, SemEval-2022 MultiCoNER shared task 11 (Malmasi et al., 2022b) used different strategies and metrics. Since the evaluation file contains instances from all 11 languages and two other language settings including multilingual and code-mixed, the macro averaged F1 score of all these languages is used as the primary evaluation metric to rank the participants' systems. However, the organizers reported the results based on precision and recall evaluation measures too.

3.4 Results and Analysis

In this section, we analyze the performance of our proposed approaches in the MultiCoNER shared task. We have employed two different approaches including a BiLSTM-CRF based approach and a transformer-based approach. The dataset comprises of 11 different languages and two other language settings including multilingual and code-mixed. The overall performance of the system is estimated considering the macro average F1 score obtains in each languages dataset. Considering this,

https://huggingface.co/xlm-roberta-base

https://huggingface.co/models

Track	BiLSTM-CRF based System			Transformers based System		
	F1-macro	Precision	Recall	F1-macro	Precision	Recall
EN	0.6403	0.6871	0.6033	**0.6924**	**0.6872**	**0.6981**
ES	**0.6562**	**0.6894**	**0.6313**	0.6138	0.6069	0.6232
NL	**0.6794**	**0.7174**	**0.650**	0.5981	0.5985	0.6026
RU	0.6177	**0.6971**	0.5578	**0.6308**	0.626	**0.639**
TR	**0.553**	**0.6457**	0.4925	0.538	0.5285	**0.558**
KO	0.6128	**0.681**	0.5616	**0.6205**	0.6227	**0.6214**
FA	0.5454	**0.5941**	0.5094	**0.5581**	0.558	**0.5617**
DE	**0.7249**	**0.7493**	**0.7047**	-	-	-
ZH	0.387	0.5461	0.3372	**0.6722**	**0.6855**	**0.6761**
HI	**0.5768**	**0.6146**	0.5477	0.5563	0.5638	**0.5551**
BN	0.428	0.4858	0.395	**0.5055**	**0.5221**	**0.4942**
MULTI	0.3505	0.4926	0.3065	**0.644**	**0.6479**	**0.652**
MIX	-	-	-	**0.6403**	**0.6423**	**0.6436**

Table 5: CSECU-DSG results of both systems on all tracks.

Model Type	Parameter	Track Name												
		ES	NL	TR	DE	HI	EN	RU	KO	FA	ZH	BN	MULTI	MIX
BiLSTM-CRF based system	learning rate	0.1	0.1	0.1	0.1	0.1	-	-	-	-	-	-	-	-
	epoch	50	100	100	100	100	-	-	-	-	-	-	-	-
	batch size	32	32	32	32	32	-	-	-	-	-	-	-	-
	hidden size	256	128	256	128	256	-	-	-	-	-	-	-	-
Transformer based system	learning rate	-	-	-	-	-	3e-5	3e-5	3e-5	3e-5	3e-5	3e-5	3e-5	3e-5
	epoch	-	-	-	-	-	7	10	10	7	10	10	6	25
	batch size	-	-	-	-	-	16	16	16	16	16	16	16	16

Table 6: Experimental settings best performing system of all track.

we analyze the performance of our proposed systems, based on each language. The corresponding results are reported in Table 5.

Results showed that the transformer-based approach obtained better performances in most of the languages compared to the BiLSTM-CRF with stacked embedding approach in terms of primary evaluation measure F1-macro. However, in terms of precision BiLSTM-CRF performed better and in terms of recall transformer-based system performed better in most of the languages dataset. Considering the diverse performances in these two approaches the optimal parameter settings of the best-performing ones are articulated in Table 6 and default settings used for the other parameters.

However, comparative performance analysis in most of the languages dataset showed that our proposed system did not perform well to address the task challenges. However, the best performing results of our proposed CSECU-DSG system in the MultiCoNER shared task for the Chinese(ZH) and Hindi (HI) languages along with other top performing and competitive participants systems are articulated in Table 7. Following the benchmark of the MultiCoNER shared task, participants' systems are ranked based on the primary evaluation measures F1-macro. For these two languages, our proposed CSECU-DSG system obtained comparatively better performances.

4 Conclusion and Future Directions

In this paper, we presented our proposed systems to address the challenges of the MultiCoNER shared task. We employed a BiLSTM-CRF with a stacked embedding-based approach and a transformer-based approach. Experimental results demonstrate

Team (Rank)	F1-macro
Chinese (ZH)	
CSECU-DSG (9th)	0.6722
Performance of other participants'	
USTC-NELSLIP (1st)	0.8169
OPDAI (3rd)	0.7954
QTrade AI (8th)	0.7400
RACAI (16th)	0.6270
MarSan_AI (19th)	0.5664
Hindi (HI)	
CSECU-DSG (11th)	0.5768
Performance of other participants'	
DAMO-NLP (1st)	0.8623
RACAI (3rd)	0.6808
YNUNLP (8th)	0.6339
silpa_nlp (14th)	0.5149
Enigma (17th)	0.4862

Table 7: Comparative results with other selected participants on Chinese and Hindi track.

that transformer-based approach performed better compared to the other approach in most of the languages dataset.

In the future, we have a plan to incorporate the task-specific features and technologies to address the challenges properly. We also have a plan to explore the existing NER technologies and fuse them in a unified architecture to overcome the limitations of the current approaches.

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2019. A multi-task approach for named entity recognition in social media data. *arXiv preprint arXiv:1906.04135*.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT Model. arXiv:1912.09582.

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

Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, and Mohammad Manthouri. 2020. Parsbert: Transformer-based model for persian language understanding. *ArXiv*, abs/2005.12515.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Rudra Murthy, Mitesh M Khapra, and Pushpak Bhattacharyya. 2018. Improving ner tagging performance in low-resource languages via multilingual learning. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 18(2):1–20.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Lucy Park, Alice Oh, Jungwoo Ha, and Kyunghyun Cho. 2021. Klue: Korean language understanding evaluation.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. HuggingFace's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

CMNEROne at SemEval-2022 Task 11: Code-Mixed Named Entity Recognition by leveraging multilingual data

Suman Dowlagar
LTRC
IIIT-Hyderabad
suman.dowlagar
@research.iiit.ac.in

Radhika Mamidi
LTRC
IIIT-Hyderabad
radhika.mamidi
@iiit.ac.in

Abstract

Identifying named entities is, in general, a practical and challenging task in the field of Natural Language Processing. Named Entity Recognition on the code-mixed text is further challenging due to the linguistic complexity resulting from the nature of the mixing. This paper addresses the submission of team CMNEROne to the SEMEVAL 2022 shared task 11 MultiCoNER. The Code-mixed NER task aimed to identify named entities on the code-mixed dataset. Our work consists of Named Entity Recognition (NER) on the code-mixed dataset by leveraging the multilingual data. We achieved a weighted average F1 score of 0.7044, i.e., 6% greater than the baseline.

1 Introduction

Named entity recognition (NER) is a fundamental task in NLP. It aims to identify and classify entities in a text into predefined types. It is an essential tool for information retrieval, question answering, (Banerjee et al., 2019) and text summarization tasks (Patil et al., 2016; Li et al., 2020). However, except for some resource-rich monolingual languages, NER annotated data for most other languages are still very limited (Kruengkrai et al., 2020). Moreover, it is usually time-consuming to annotate such data, particularly for low-resource languages such as multilingual and code-mixed (Liu et al., 2021). Therefore, transfer learning and leveraging the other datasets for multilingual and code-mixed NER has attracted growing interest recently, especially with the influx of deep learning methods.

This paper presents the system description for named entity recognition on the code-mixed dataset. Code-mixing is defined as using two or more languages in a single sentence or utterance (Dowlagar and Mamidi, 2021). The use of code-mixed language is prevalent in most multilingual societies. Due to linguistic complexity arising from mixing two languages, the processing of code-mixed sentences is a challenging task (Bali et al., 2014). So, the models that are trained on monolingual and multilingual datasets typically fail to handle code-mixed inputs (Khanuja et al., 2020). Therefore, to encourage research on code-mixing, the speech and NLP communities are organizing several shared tasks. The shared tasks have concentrated on language identification, POS-tagging, sentiment analysis, hate speech detection, and several datasets exist for these as well. Similarly, SEMEVAL 2022's Task 11 sub-task 13 was devoted to identifying named entities in code-mixed languages (Malmasi et al., 2022b). This task aims to classify the given tokens in the code-mixed sentences as persons, corporation, location, and others. An example is shown in Table 1.

Sentence	*hameM*	this	magic	moment
Languge	Hi	En	En	En
NER tags	O	B-CW	I-CW	I-CW

Table 1: An example of the code-mixed NER annotated sentence. The Hindi words are converted from utf to wx format and are italicized

The lack of annotated data is a crucial issue for code-mixed datasets. Lack of data poses a problem of data overfitting and poor entity recognition. The language models trained on such low resource datasets cannot generalize the training data, thus performing low on the test datasets. Several previous studies have used monolingual data as training signals for transfer learning, and these data can also be used in the form of pre-training. Thus, we used a similar approach of including the multilingual data along with the code-mixed dataset.

We used the multilingual pre-trained BERT model as our model for code-mixed NER. The model uses code mixed training data along with the multilingual training and mulilingual validation data.

sIriyala	*naMbara*	*xvArA*	kleding	in	de	oudheid	*kI*	*pahacAna*	*kareM*
O	O	O	B-PROD	I-PROD	I-PROD	I-PROD	O	O	O
what	city	is	dig	me	out	in?			
O	0	O	B-CW	I-CW	I-CW	O			
AmAra	das	testament	*mUlya*	*kawa?*					
0	B-CW	I-CW	O	O					

Table 2: An example of the code-mixed NER annotated sentence. The multilingual words are converted from utf to wx format and are italicized

We have analyzed that Bi-LSTM with CRF models has shown an improved accuracy on the token classification tasks such as POS tagging, language identification, and NER. The ensemble of BERT or XLM-RoBERTa with Bi-LSTM and CRF would have shown a further improvement in the code-mixed NER. Also, using language identification as a downstream task with the current method might have improved the NER's accuracy.

The paper is organized as follows. Section 2 provides related work on Named Entity Recognition on CM social media text. Section 3 provides information on the task and examples of datasets. Section 4 describes the proposed work. Section 5 presents the experimental setup and Section 6 project the performance of the model. Section 7 concludes our work.

2 Related Work

Code-mixed NER has attracted a lot of attention in the NLP community this decade. This section lists the latest works on code-mixed named entity recognition.

Priyadharshini et al. (2020); Winata et al. (2019) generated multilingual meta representations from pre-trained monolingual word embeddings. The model learned to construct the best word representation by mixing multiple sources without explicit language identification.

Aguilar et al. (2019) presented a shared task on named entity recognition in the CALCS workshop. The language pairs used were English-Spanish (ENG-SPA) and Modern Standard Arabic Egyptian (MSA-EGY). They used Twitter data and nine entity types to establish a new dataset for code-switched NER benchmarks. The participating teams used LSTM, CNN, CRF, and word representations to recognize named entities.

Singh et al. (2018) We presented a new Code-Mixed Hinglish corpus for NER. Different machine learning classification algorithms with word, character, and lexical features are used to establish baselines. The algorithms used were Decision tree, Long Short-Term Memory (LSTM), and Conditional Random Field (CRF).

(Meng et al., 2021; Fetahu et al., 2021) presented a novel CM NER model. They proposed a gated architecture that enhances existing multilingual Transformers by dynamically infusing multilingual knowledge bases, a.k.a gazetteers. The evaluation of code-mixed queries shows that this approach efficiently utilizes gazetteers to recognize entities in code-mixed queries with an F1=68%, an absolute improvement of +31% over a non-gazetteer baseline.

(Meng et al., 2021) mentioned that including Gazetteer features could cause models to overuse or underuse them, leading to poor generalization. They proposed a new approach for gazetteer knowledge integration by including Context in Gazetteer Representation using encoder and Mixture-of-Experts gating network models. These models overcome the feature overuse issue by learning to conditionally combine the context and gazetteer features instead of assigning them fixed weights.

3 Task Setup

The shared task detects semantically ambiguous and complex entities in short and low-context settings. Complex NEs, like the titles of creative works (movie/book/song/software names), are not simple nouns. Usually, they take imperative clauses, or they often resemble typical syntactic constituents. Such NEs are harder to recognize (Ashwini and Choi, 2014). Syntactic parsing of such complex noun phrases is hard, and most NER systems fail to identify them. Inside–outside–beginning (IOB) format (Ramshaw and Marcus, 1999) is used for annotating entities. A few examples of complex NEs and ambiguous NEs from the code-mixed dataset are given in Table 2.

So the MultiCoNER shared task encourages the models to handle such complex NEs. A huge dataset (Malmasi et al., 2022a) is released for this task (Malmasi et al., 2022b). The languages focused on in this shared task are: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla. The shared task also offered an additional track with code-mixed and multilingual datasets. In this paper, we will be concentrating on the code-mixed dataset.

4 System Overview

We finetuned the pre-trained multilingual BERT model by using the multilingual training and validation datasets for code-mixed named entity recognition. We found that the training data is insufficient for the deep learning language model to identify the named entities in the validation data correctly. The data scarcity of low-resource languages has been a significant challenge for building NLP systems since they require a large amount of data to learn a robust model. We observed that the multilingual NER training data is similar to the code-mixed dataset. Also, it is relatively large when compared to the code-mixed dataset. In our approach, the multilingual training and validation data is combined with the code-mixed training dataset. Using the combined dataset, we finetune the deep neural network model. Our method thus attempts to learn language-agnostic features by using the combined multilingual and code-mixed dataset. This finetuned model can be used to infer named entity information at a token level on a code-mixed low resource language.

We used the pre-trained mBERT (Devlin et al., 2018) model for code-mixed NER. mBERT is a transformer-based self multi-headed attention model that is pre-trained on a massive collection of multilingual data and can be finetuned for our NER task. As the model is pre-trained on a large corpus, the semantic and syntactic information is well modeled and can be directly finetuned for a specific task. BERT is a bi-directional transformer model (Vaswani et al., 2017). It analyzes the meaning of a term depending on its context given on both sides. The transformer part in the BERT works like an attention mechanism capable of learning the contextual relationships between the terms in a sentence.

5 Experimental Setup

The section presents the baselines, hyper-parameter settings, and analysis of observed results.

5.1 Baselines

The baselines used for the proposed work is:

Conditional random field (CRF) (Lafferty et al., 2001) CRF is a statistical model and is a well-known approach for handling NER problems. The CRF model considers the neighboring samples by modeling the prediction as a graphical model. It assumes that the tag for the present word (denoted as y_i) is dependent on the tag of its previous/next word (denoted as y_{i-1} or y_{i+1}).

MultiCoNER baseline (Malmasi et al., 2022b) The XLM-RoBERTa base with CRF model is used as a baseline for NER.

Pre-trained multilingual BERT (mBERT) (Devlin et al., 2018) A pre-trained multilingual BERT model with token classification without leveraging the multilingual data is used as a baseline.

5.2 Hyperparameters and libraries

For developing our model, the neural network library used is PyTorch, and the pre-trained multilingual BERT model (*bert-base-multilingual-cased*) and XLM-ROBERTa base model (*xlm-roberta-base*) is obtained from the hugging face-transformers library and is finetuned for the code-mixed NER task. The model is implemented in Kaggle Notebook with GPU processing.

The batch size of the datasets is kept as 64. The maximum length of the sentence from the training data is considered during the input data encoding/padding. Due to subword tokenization, we used the first token for predicting the tag. The optimizer used is weighted Adam with the learning rate of 2e-5 and epsilon value equal to 1e-5. The dropout is set to 0.1. The loss function used is a cross-entropy loss that is inbuilt into the transformer's BERT model. The number of epochs used for training the model is 30. The training is stopped when there is no change in validation accuracy for more than four epochs.

The CRF is obtained from pytorch-crfsuite library[1]. The previous word and its tag, the next

[1]https://github.com/scrapinghub/python-crfsuite

Model	F1-score			
	w/o multilingual data		with multilingual data	
	valid data	test data	valid data	test data
CRF	0.565	0.561	0.560	0.556
mBERT	0.627	0.612	**0.716**	**0.707**
MultiCoNER baseline	0.651	0.645	0.725	0.719

Table 3: The performance of the models on the code-mixed dataset with and without including multilingual data. Our submission for the given task is highlighted.

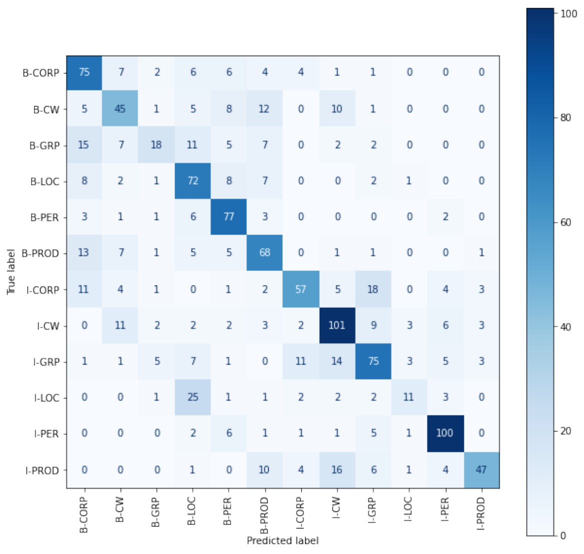


Figure 1: Confusion matrix of CM-NER baseline

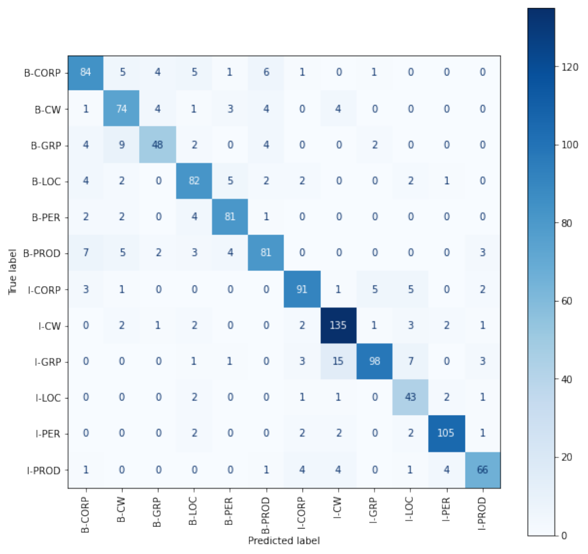


Figure 2: Confusion matrix of CM-NER by leveraging multilingual data

word, and its tag is used as the features to predict the tag of the current word.

6 Results and Analysis

Table 3 presents the f1-score of the models on the Dravidian code-mixed dataset. From the above results, it is clear that our system, i.e., leveraging the multilingual NER data in a low-resource code-mixed setting, improves the NER task compared to the baseline models. The CRF model didn't perform well on the given NER task, as this statistical model does not capture the semantics of the tokens. Even the CRF with multilingual data performed poorly on this task compared to the baseline NN models. It shows the importance of capturing semantical, syntactic, and contextual information while building the NER model on these complex datasets.

Our submission, the pre-trained mBERT by leveraging the multilingual dataset, performed better than the MultiCoNER baseline by 6%. Even the MultiCoNER baseline with the multilingual dataset performed better than our submission.

The confusion matrices with and without multilingual data of our submission on the code-mixed NER validation dataset are shown in the Figures **??** and 2. By using confusion matrices, we observed that the multilingual data given in Figure 2 helped better identify the CW, PROD, CROP, and LOC entities when compared to the baseline model.

7 Conclusion and future work

In this paper, we addressed the shared task on named entity recognition for the code-mixed dataset. As the code-mixed data is a low resource language and there are no pre-trained models, we leveraged the multilingual dataset for training the NER model. The model used for testing our method is the pre-trained multilingual BERT. We finetuned the pre-trained mBERT for the code-mixed NER task by using the code-mixed training data and multilingual training and validation datasets.

The use of meta embeddings for dealing with code-mixed datasets has recently attracted a lot of attention. It might be possible that meta embedding-based NER will work better on this code-mixed dataset. Unlike the social media data where code-mixed sentences/words are written in Roman script, the native script is used for each word, so the language identification will work better on this dataset. Using Language identification or POS tagging as a downstream task for NER on this dataset will help in improving the code-mixed NER.

References

Gustavo Aguilar, Fahad AlGhamdi, Victor Soto, Mona Diab, Julia Hirschberg, and Thamar Solorio. 2019. Named entity recognition on code-switched data: Overview of the calcs 2018 shared task. *arXiv preprint arXiv:1906.04138*.

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*.

Kalika Bali, Jatin Sharma, Monojit Choudhury, and Yogarshi Vyas. 2014. "i am borrowing ya mixing?" an analysis of english-hindi code mixing in facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 116–126.

Partha Sarathy Banerjee, Baisakhi Chakraborty, Deepak Tripathi, Hardik Gupta, and Sourabh S Kumar. 2019. A information retrieval based on question and answering and ner for unstructured information without using sql. *Wireless Personal Communications*, 108(3):1909–1931.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Suman Dowlagar and Radhika Mamidi. 2021. Offlangone@ dravidianlangtech-eacl2021: Transformers with the class balanced loss for offensive language identification in dravidian code-mixed text. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, pages 154–159.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Simran Khanuja, Sandipan Dandapat, Sunayana Sitaram, and Monojit Choudhury. 2020. A new dataset for natural language inference from code-mixed conversations. *arXiv preprint arXiv:2004.05051*.

Canasai Kruengkrai, Thien Hai Nguyen, Sharifah Mahani Aljunied, and Lidong Bing. 2020. Improving low-resource named entity recognition using joint sentence and token labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5898–5905.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Linlin Liu, Bosheng Ding, Lidong Bing, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. Mulda: A multilingual data augmentation framework for low-resource cross-lingual ner. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5834–5846.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Nita Patil, Ajay S Patil, and BV Pawar. 2016. Survey of named entity recognition systems with respect to indian and foreign languages. *International Journal of Computer Applications*, 134(16).

Ruba Priyadharshini, Bharathi Raja Chakravarthi, Mani Vegupatti, and John P McCrae. 2020. Named entity recognition for code-mixed indian corpus using meta embedding. In *2020 6th international conference on advanced computing and communication systems (ICACCS)*, pages 68–72. IEEE.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Vinay Singh, Deepanshu Vijay, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. Named entity recognition for hindi-english code-mixed social media text. In *Proceedings of the seventh named entities workshop*, pages 27–35.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Genta Indra Winata, Zhaojiang Lin, and Pascale Fung. 2019. Learning multilingual meta-embeddings for code-switching named entity recognition. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 181–186.

RACAI at SemEval-2022 Task 11: Complex named entity recognition using a lateral inhibition mechanism

Vasile Păiș
Research Institute for Artificial Intelligence ”Mihai Drăgănescu”, Romanian Academy
Bucharest, Romania
vasile@racai.ro

Abstract

This paper presents RACAI's system used for the shared task of "Multilingual Complex Named Entity Recognition (MultiCoNER)", organized as part of "The 16th International Workshop on Semantic Evaluation (SemEval 2022)". The system employs a novel layer inspired by the biological mechanism of lateral inhibition. This allowed the system to achieve good results without any additional resources apart from the provided training data. In addition to the system's architecture, results are provided as well as observations regarding the provided dataset.

1 Introduction

Named entity recognition (NER) is a well known task in natural language processing. It aims to detect spans of text associated with known entities. Initially, much work focused on detecting persons, organizations, and locations (Grishman and Sundheim, 1996; Tjong Kim Sang and De Meulder, 2003). However, this limited approach is not suitable for every domain, thus leading to research in domain-specific NER. For example, in the biomedical domain, a number of works have addressed entities such as genes, proteins, diseases (Hu and Verberne, 2020), cell types (Settles, 2004), chemicals (Gonzalez-Agirre et al., 2019; Ion et al., 2019). Similarly, in the legal domain additional classes are employed such as money value (Glaser et al., 2018), legal reference (Landthaler et al., 2016; Păiș et al., 2021), judge, and lawyer (Leitner et al., 2019).

In the context of "The 16th International Workshop on Semantic Evaluation (SemEval 2022)"[1], the task number 11 "Multilingual Complex Named Entity Recognition (MultiCoNER)"[2] (Malmasi et al., 2022b) required participants to build a NER system able to recognize complex entities in 11 languages: Bangla, Chinese, Dutch, English, Farsi, German, Hindi, Korean, Russian, Spanish, and Turkish. In addition, a multilingual track and a code-mixed track were available. The task focused on 6 entity types: person, location, group, corporation, product and creative work.

As noted by Ashwini and Choi (2014), non-traditional entities can pose a challenge for NER systems. This happens because datasets are harder to build and certain entities (such as creative works) are updated more frequently than traditional ones (persons, locations). Furthermore, traditional entities tend to occur as noun phrases, while the newly proposed entities (for the purposes of the task) may be linguistically complex (complex noun phrases, gerunds, infinitives or full clauses). An interesting result was provided by Aguilar et al. (2017), where the top system from WNUT 2017 achieved only 8% recall when dealing with creative works.

This paper describes a system for complex NER in a multilingual context, developed at the Research Institute for Artificial Intelligence of the Romanian Academy (RACAI), that participated in the MultiCoNER task. The system employs a new artificial neural network layer trying to mimic the biological process of *lateral inhibition* (Cohen, 2011). In various regions of the brain, excited neurons can reduce the activity of other neighbouring neurons. In the visual cortex this process may account for an increased perception in low-lighting conditions. Thus, intuitively the newly proposed system may better focus on subtle details present in the data and the language model.

The paper is structured as follows: Section 2 presents related work, Section 3 describes the dataset and pre-processing operations used, Section 4 describes the method used with the system architecture in Section 4.1 and performed experiments in Section 4.2. The results are given in Section 5 and finally, conclusions and future work

[1]https://semeval.github.io/SemEval2022/
[2]https://multiconer.github.io/

are available in Section 6.

2 Related work

In a survey regarding NER using deep learning models, Yadav and Bethard (2018) emphasize the importance of pre-trained word embedding representations. Dernoncourt et al. (2017), in the NeuroNER package[3], combine pre-trained word embeddings with character level embeddings passing through a neural network with a final CRF layer achieving high scores on different datasets. Other authors, using similar architectures have shown that combining multiple static word representations can further increase the overall system performance (Păiș and Mitrofan, 2021).

With the introduction of contextual word representation models, such as BERT (Devlin et al., 2019), ELMo (Peters et al., 2018), ROBERTA (Liu et al., 2019), XLNet (Yang et al., 2019), NER systems have been adapted to make use of these new models. For the majority of the contextual models, depending on the size of the artificial neural network being used, we distinguish between a base version and a large version (with a larger number of parameters). Devlin et al. (2019) used the BERT-large model for NER on the CoNLL-2003 English dataset, achieving an F1 score of 92.8%. Nguyen et al. (2020) propose a custom BERT-like model for English tweets, called BERTweet, for improving the NER performance on two datasets: WNUT-16 and WNUT-17.

Contextualized embeddings were also applied with success in domain-specific settings. Considering the ProfNER shared task (Miranda-Escalada et al., 2021), dedicated to identifying mentions of occupations in health-related social media, the best performing system made use of the BETO model (Cañete et al., 2020) trained on a large Spanish corpus.

Wang et al. (2021) propose an algorithm for automatically finding a concatenation of embeddings that improves a system's performance in different tasks, including NER. The authors considered multiple contextual and non-contextual embeddings and the proposed algorithm can identify any of their combinations. Their work builds on previous experiments that showed increased performance when manually concatenating contextual and non-contextual embeddings (Straková et al., 2019; Wang et al., 2020).

Training a contextualized embedding model requires many processing resources. This means that not all flavours are available for all languages. Multilingual models have been proposed, making use of training data in multiple languages. Such models include mBERT and XLM-RoBERTa (XLM-R) (Conneau et al., 2020). These models are known to perform especially well on low resourced languages. Considering NER, Conneau et al. (2020) show that XLM-R large performs better than mBERT, providing an average 2% increase in F1 score for Dutch, Spanish and German. Interestingly, in English the performance of XLM-R is more similar to mBERT (though still offering an improved performance of less than 1%) and less than (Akbik et al., 2018). This seems to support the idea that monolingual models, trained on a specific language or domain, can perform better than multilingual ones.

Meng et al. (2021) recognizes the importance of gazetteer resources, even in the case of state-of-the-art systems making use of contextualized embeddings. The authors propose using an encoder for obtaining Contextual Gazetteer Representations (CGRs) as a way to incorporate any number of gazetteers into a single, span-aware, dense representation. Then, the authors go one step further and propose a gated Mixture-of-Experts (MoE) method to fuse CGRs with contextual word representations from any word-level model.

Fetahu et al. (2021) employ multilingual gazetteers fused with transformer models in a MoE approach to improve the recognition of entities in code-mixed web queries. In this case the entities were written in a different language than the rest of the query, thus posing particular challenges to existing NER systems.

3 Dataset and pre-processing

The dataset (Malmasi et al., 2022a) was provided in a column-based format, with 4 columns in each file. The text was tokenized, with tokens available in the first column. Columns 2 and 3 did not contain useful information (only an underscore symbol was present). Column 4 contains the named entity annotation in BIO format. The first token (or the single token) of an entity contains the "B-" prefix. Other entity tokens (in the case of multi-token entities) start with an "I-" prefix, while non-entity tokens are denoted with "O".

For each language there was a train/dev/test split

[3]http://neuroner.com/

provided by the organizers. In the train set there were 15,300 sentences, in the dev set there were 800 sentences, while the test set was much larger. The actual number of sentences in the test set varies between languages, having as many as 217,818 sentences for English. For the multilingual track, there were 168,300 train sentences, 8,800 dev sentences and 471,911 test sentences. The smallest number of sentences was given in the code-mixed track with only 1,500 training sentences, 500 dev sentences and 100,000 test sentences.

Regardless of the language or additional multilingual and code-mixed tracks the development set is very small while the test set is much larger than the training set. In addition, the number of training sentences in the code-mixed track seems insufficient to build a model by themselves.

Predictions on the test set were required to provide annotations, in the same BIO format, for each token. Furthermore, considering the multilingual nature of the task, with very different languages (such as German and Chinese), no pre-processing was applied, other than converting the provided format to a format suitable for the developed system. For the code-mixed track, a second dataset was generated, based on the original provided dataset augmented with new sentences. These were generated by randomly replacing existing entities in the code-mixed dataset with similar entities extracted from the multilingual dataset. For each sentence containing at least one entity, a new sentence was generated, thus doubling the size of the training dataset for the code-mixed track.

4 Method

4.1 System Architecture

The proposed system employs the XLM-RoBERTa (Conneau et al., 2020) large model. Given an input sequence from the dataset it is first transformed into model-specific tokens. The sequence is also enhanced with the special tokens for sequence start (CLS), sequence end (SEP) and, if needed, padding (PAD). Then it passes through the model to obtain associated contextual embeddings. In existing systems, the word representations usually pass through a linear layer and finally a classification head, giving predictions for each input token. However, in the proposed system, there is a new layer inserted just after the XLM-RoBERTa embeddings calculation and before the linear layer. The resulting system architecture is presented in Figure 1.

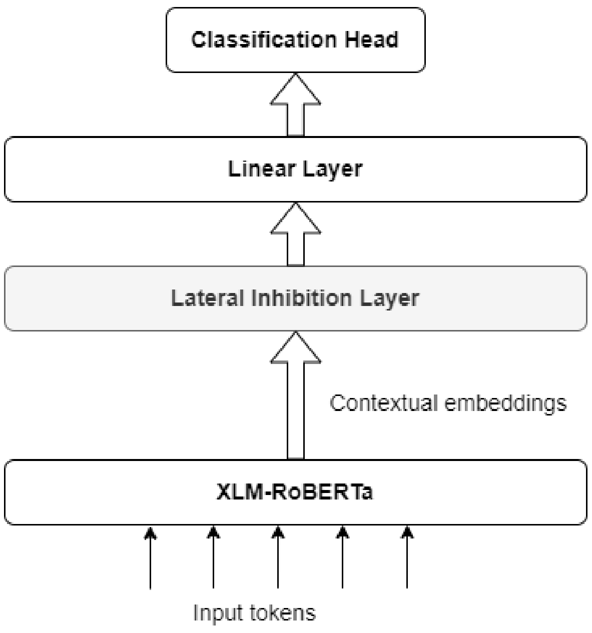


Figure 1: System architecture

The newly introduced layer follows the biological process of lateral inhibition. Thus, an embedding value is either allowed to pass unchanged to the next layer or set to zero, depending on the other values. Similar to a linear layer, a matrix of weights (W) and a bias (B) were kept. However, the diagonal values of the weights matrix were always set to zero to allow only interaction from adjacent neurons. Furthermore, the Heaviside function (see Equation 1) was applied to determine which values pass through the layer or become zero. The equation associated with the forward pass is given in Equation 2, where X is the layer's input vector, associated with a token embedding representation, $Diag$ represents the matrix diagonal, $ZeroDiag$ is the matrix with the value zero on the diagonal, and W and B represent the weights and bias.

$$\Theta(x) = \begin{cases} 1, x > 0 \\ 0, x \leq 0 \end{cases} \tag{1}$$

$$F(X) = X * Diag(\Theta(X * ZeroDiag(W) + B)) \tag{2}$$

To overcome the problem of computing a derivative for the Heaviside function, in the backwards pass the Heaviside function was approximated with the sigmoid function with a scaling parameter (Wunderlich and Pehle, 2021). This is described in Equation 3. The derivative of the sigmoid function is given in Equation 4, where $\sigma(x)$ is the same as in Equation 3. This approximation technique is also

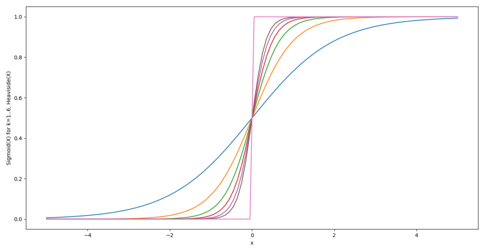


Figure 2: $\sigma(x)$ for k=1..6 and $\Theta(x)$

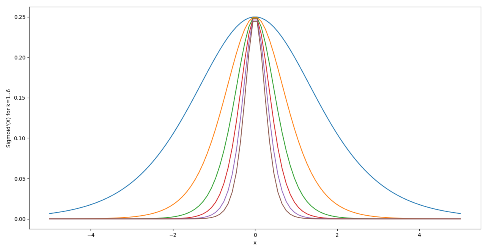


Figure 3: $\sigma'(x)$ for $k = 1..6$

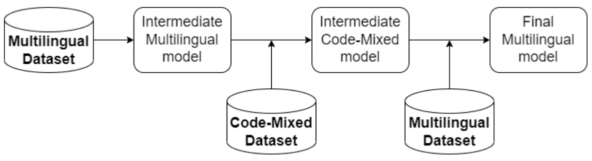


Figure 4: Training flow for creating the multilingual model

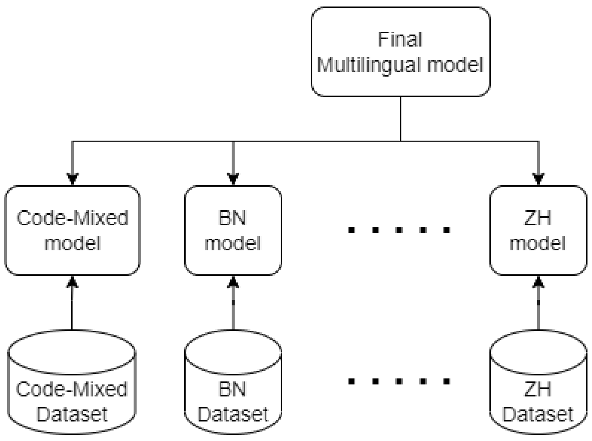


Figure 5: Language-specific training

known as surrogate gradient learning (Neftci et al., 2019). It allows the use of a non-differentiable function in the forward pass (in this case the Heaviside function) and approximates the derivative in the backwards pass by means of a different function.

$$\sigma(x) = \frac{1}{1 + e^{-kx}} \tag{3}$$

$$\sigma'(x) = k\sigma(x)\sigma(-x) \tag{4}$$

Figure 2 shows a plot for the Sigmoid function, for various values of the scaling parameter k, superimposed on a plot of the Heaviside function. Higher values for the scaling parameter give better approximations for the Heaviside function. A plot for the derivative of the Sigmoid function with $k = 1..6$ is given in Figure 3.

The entire system was then implemented in PyTorch following the system diagram given in Figure 1. The embedding vector associated with each token is processed by the lateral inhibition layer, then goes through a linear layer using ReLU as the activation function and finally through the classification head.

4.2 Experiments

Training started with the creation of a multilingual model based on the provided multilingual corpus. Training was performed for 40 epochs, with the best performing model on the dev dataset being stored. The resulting intermediate model was fine-tuned with the code mixed data, for another 40 epochs. Thus, all the available data were now included in the new intermediate model. This model was then used as the basis for further training on the multilingual dataset (another 40 epochs), producing the final multilingual model. The results from this final model were submitted in the multilingual task. This training procedure is described in Figure 4.

The final multilingual model was then used as a starting point for the other models. The architecture was the same and the model parameters were initialized with the multilingual parameters. Fine-tuning was performed for 80 epochs for all languages and in the case of the code-mixed dataset. The best performing model for each task was stored and the results were submitted to the corresponding category. The language-specific training procedure is described in Figure 5.

The models were trained using a single Nvidia Quadro RTX 5000 GPU board. The dataset was first pre-processed, as described in Section 3. Several experiments were performed to determine the best parameters. However, due to limited time and hardware resources, it was not possible to perform an exhaustive search. The participating models employed learning rates of $1e-05$ (the majority of the models) or $2e-05$ (Bangla and code-mixed).

Track	With Lateral Inhibition				Without Lateral Inhibition				Diff
	P	R	F1	Epoch	P	R	F1	Epoch	F1
Bangla	**68.08**	**65.33**	**66.28**	40	65.95	63.99	64.68	68	1.60
Chinese	**68.17**	62.05	62.70	35	67.05	**64.39**	**64.41**	56	-1.71
Dutch	78.82	**78.30**	**78.41**	63	**78.85**	77.27	77.25	37	1.16
English	76.54	75.35	75.78	66	**76.59**	**76.09**	**76.21**	53	-0.43
Farsi	**70.77**	**70.45**	**70.42**	36	70.10	68.84	69.24	21	1.18
German	80.01	78.97	79.39	37	**80.19**	**79.44**	**79.70**	71	-0.31
Hindi	69.05	67.77	68.08	42	**70.14**	**68.18**	**68.87**	65	-0.79
Korean	**72.06**	71.93	71.74	8	71.83	**72.58**	**71.99**	70	-0.16
Russian	75.86	73.83	74.60	62	**75.89**	**73.94**	**74.68**	62	-0.08
Spanish	**76.21**	**75.43**	**75.62**	4	75.88	75.25	75.36	58	0.26
Turkish	**71.81**	**70.25**	**70.42**	78	71.28	69.14	69.70	74	0.72
Multilingual	**72.25**	**72.78**	**72.10**	33	71.78	72.72	71.87	17	0.23
Code-mixed	**79.58**	**79.23**	**79.37**	63	78.93	79.02	78.95	62	0.42
Intermediate Multilingual	**71.53**	**72.34**	**71.50**	34	70.48	71.11	70.35	14	1.15
Intermediate Code-mixed	**79.28**	**79.42**	**79.31**	31	78.50	79.00	78.73	28	0.58

Table 1: Results on the test dataset for all tracks.

A dropout value of 0.1 was used for both the lateral inhibition and linear layers. For the backwards pass approximation of the Heaviside function, the sigmoid scaling parameter was set to 10.

5 Results

The models produced by the system described in the previous sections participated in all the MultiCoNER tracks. The official ranking metric was macro-averaged F1. The results are given in Table 1 for all tracks, in alphabetical order of the languages, while the multilingual and code-mixed tracks are shown at the end of the table.

The best best average F1 score was achieved in the German track (79.39%), followed closely by the code-mixed track (79.37%). All these results were obtained by using only the provided dataset, without external resources. For most of the tracks precision and recall are similar (with precision being approximately 1% higher than recall), with the exception of Chinese (precision is 6% higher than recall) and Russian (precision is 2% higher than recall).

In Table 1 is also given a comparison between the results obtained using the new lateral inhibition layer and the results obtained by the same system without the lateral inhibition layer. In 7 tracks (Bangla, Dutch, Farsi, Spanish, Turkish, Multilingual, Code-mixed) the new layer improved the overall F1 score, in 1 track (Russian) the results were roughly the same (a difference of 0.08 %), while in 5 tracks (Chinese, English, German, Hindi, Korean) the new layer actually decreased the system's performance. By looking at the size of the training corpora used in XLM-RoBERTa, as reported by Conneau et al. (2020), it seems the new layer improved the system performance in the case of languages represented by less than 54Gb of data. This seems to confirm the intuition behind the new layer, that the model is able to better focus on details present in the data and potentially filter out the noise. There are two exceptions: Farsi, trained on 111 Gb of data, and Hindi, trained on 20 Gb. In this case additional investigation should probably be performed with regard to the quality of the data used for training the model, possibly taking into account the relative language complexity (Bentz et al., 2016).

Table 1 also presents the training epochs associated with the best model. For some languages, the new layer is able to reduce the number of training epochs, but there is no clear pattern for when this happens. Interestingly however, for Spanish and Korean the new layer is able to reduce the training time to less than 10 epochs.

In the multilingual and code-mixed tracks, the model enhanced with the lateral inhibition layer

Track	LOC	PER	PROD	GRP	CW	CORP
Bangla	69.03	77.41	62.92	73.46	49.57	65.31
Chinese	72.30	64.71	67.93	45.44	58.53	67.27
Dutch	80.85	89.82	76.99	75.14	71.35	76.29
English	77.29	90.11	73.27	72.93	66.76	74.29
Farsi	74.04	79.19	70.71	72.33	58.59	67.67
German	82.19	90.24	78.66	75.28	73.39	76.57
Hindi	70.93	75.14	66.35	68.70	57.13	70.25
Korean	76.54	77.86	72.48	68.02	64.02	71.51
Russian	74.05	80.30	75.04	71.29	70.09	76.84
Spanish	76.95	89.27	70.81	71.61	68.69	76.40
Turkish	72.44	81.80	73.01	64.85	61.91	68.49
Multilingual	76.84	83.85	68.82	64.89	66.72	71.47
Code-mixed	82.29	88.29	81.09	73.13	73.99	77.42

Table 2: F1 scores on the test dataset for all tracks and for all entity types.

was able to improve both precision (by 0.47% for multilingual and 0.65% for code-mixed) and recall (by 0.06% for multilingual and 0.21% for code-mixed), leading to corresponding F1 differences. Nevertheless, the number of epochs is not reduced for these tracks. As described in Section 4.2 and illustrated in Figure 4, there were also two intermediate models (one multilingual and one code-mixed). The results for these models are also provided in Table 1. The proposed approach increased the multilingual F1 score by 0.6% for the lateral inhibition system and by 1.52% without lateral inhibition. Similarly, the code-mixed performance was increased by 0.06% (with lateral inhibition) and by 0.22% (without lateral inhibition). It seems in this case that the presence of the lateral inhibition layer reduced the overall gain obtained from training the model in multiple stages. Nevertheless, the multi-stage approach increased the final F1 score.

Table 2 presents the F1 results obtained with the lateral inhibition layer, for each track and for each entity type. It can be noticed that commonly used named entities (locations and persons) achieve the highest score in all tracks. The hardest to predict (achieving the lowest scores) are group (GRP) and creative work (CW). This observation holds for all tracks, but particularly in Chinese, the group entity obtains the lowest score (45.44%), and in Bangla, the creative works entity obtains 49.57%. It seems that these low values are due to the nature of these entities, their complexity and their evolution over time. Furthermore, by looking at the dataset structure for Chinese, the group entity type was the least represented in the training set. Moreover, by looking at the number of unique entities present in the dataset, there is a high discrepancy between unique training entities, for group and creative works, and the corresponding unique instances in the test set, especially for Chinese and Bangla.

6 Conclusion

The system described in this paper participated in the MultiCoNER shared task. It made use of a new artificial neural network layer inspired by the biological process of lateral inhibition. By means of this mechanism, the system achieved third place in the general ranking associated with 7 languages (Spanish, Dutch, Russian, Korean, Farsi, German, and Hindi). There were no additional resources employed, apart from the provided dataset (no additional text and no gazetteers). A simple data augmentation method (as described in Section 3) was applied only for the code-mixed track, while still using only the provided data.

Experiments performed after the task deadline, showed (as reported in Section 5) that not all tracks benefit from using the new layer. It is likely that languages with an increased number of tokens present when training the multilingual XLM-RoBERTa model will benefit less from employing the new layer. Nevertheless, models with the lateral inhibition layer trained for lower resourced languages, multilingual, and code-mixed datasets seem to achieve higher scores. Furthermore, improvements can be seen in both precision and recall.

The proposed lateral inhibition layer can be applied to other natural language processing tasks as well. In the future more experiments will be conducted with this layer to determine its suitability in other contexts.

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153, Copenhagen, Denmark. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Sandeep Ashwini and Jinho D. Choi. 2014. Targetable named entity recognition in social media. *CoRR*, abs/1408.0782.

Christian Bentz, Tatyana Ruzsics, Alexander Koplenig, and Tanja Samardžić. 2016. A comparison between morphological complexity measures: Typological data vs. language corpora. In *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)*, pages 142–153, Osaka, Japan. The COLING 2016 Organizing Committee.

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

Ronald A. Cohen. 2011. *Lateral Inhibition*, pages 1436–1437. Springer New York, New York, NY.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. 2017. NeuroNER: an easy-to-use program for named-entity recognition based on neural networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 97–102, Copenhagen, Denmark. Association for Computational Linguistics.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Ingo Glaser, Bernhard Waltl, and Florian Matthes. 2018. Named entity recognition, extraction, and linking in german legal contracts. In *IRIS: Internationales Rechtsinformatik Symposium*, pages 325–334.

Aitor Gonzalez-Agirre, Montserrat Marimon, Ander Intxaurrondo, Obdulia Rabal, Marta Villegas, and Martin Krallinger. 2019. PharmaCoNER: Pharmacological substances, compounds and proteins named entity recognition track. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference- 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Yuting Hu and Suzan Verberne. 2020. Named entity recognition for Chinese biomedical patents. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 627–637, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Radu Ion, Vasile Păiș, and Maria Mitrofan. 2019. RACAI's system at PharmaCoNER 2019. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, pages 90–99, Hong Kong, China. Association for Computational Linguistics.

Jörg Landthaler, Bernhard Waltl, and Florian Matthes. 2016. Unveiling references in legal texts-implicit versus explicit network structures. In *IRIS: Internationales Rechtsinformatik Symposium*, volume 8, pages 71–8.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2019. Fine-grained named entity recognition in legal documents. In *Semantic Systems. The Power of AI and Knowledge Graphs*, pages 272–287, Cham. Springer International Publishing.

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

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512, Online. Association for Computational Linguistics.

Antonio Miranda-Escalada, Eulàlia Farré-Maduell, Salvador Lima-López, Luis Gascó, Vicent Briva-Iglesias, Marvin Agüero-Torales, and Martin Krallinger. 2021. The ProfNER shared task on automatic recognition of occupation mentions in social media: systems, evaluation, guidelines, embeddings and corpora. In *Proceedings of the Sixth Social Media Mining for Health (#SMM4H) Workshop and Shared Task*, pages 13–20, Mexico City, Mexico. Association for Computational Linguistics.

Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. 2019. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63.

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

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

Vasile Păiș and Maria Mitrofan. 2021. Assessing multiple word embeddings for named entity recognition of professions and occupations in health-related social media. In *Proceedings of the Sixth Social Media Mining for Health (#SMM4H) Workshop and Shared Task*, pages 128–130, Mexico City, Mexico. Association for Computational Linguistics.

Vasile Păiș, Maria Mitrofan, Carol Luca Gasan, Vlad Coneschi, and Alexandru Ianov. 2021. Named entity recognition in the Romanian legal domain. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 9–18, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*, pages 107–110, Geneva, Switzerland. COLING.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.

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

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. More embeddings, better sequence labelers? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3992–4006, Online. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Automated concatenation of embeddings for structured prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2643–2660, Online. Association for Computational Linguistics.

Timo C. Wunderlich and Christian Pehle. 2021. Event-based backpropagation can compute exact gradients for spiking neural networks. *Scientific Reports*, 11(1):12829.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

NamedEntityRangers at SemEval-2022 Task 11: Transformer-based Approaches for Multilingual Complex Named Entity Recognition

Amina Miftahova
Innopolis University, Russia
noteisenheim@gmail.com

Alexander Pugachev
HSE University, Russia
avpugachev@edu.hse.ru

Artem Skiba
Ershov Institute of Informatics Systems, Russia
efildream@gmail.com

Ekaterina Artemova
HSE University
Huawei Noah's Ark lab, Russia
elartemova@hse.ru

Tatiana Batura
Moscow State University
Novosibirsk State University, Russia
t.batura@g.nsu.ru

Pavel Braslavski
HSE University
Ural Federal University, Russia
pbras@yandex.ru

Vladimir Ivanov
Innopolis University
Kazan Federal University, Russia
v.ivanov@innopolis.ru

Abstract

This paper presents the two submissions of NamedEntityRangers Team to the MultiCoNER Shared Task, hosted at SemEval-2022. We evaluate two state-of-the-art approaches, of which both utilize pre-trained multi-lingual language models differently. The first approach follows the token classification schema, in which each token is assigned with a tag. The second approach follows a recent template-free paradigm (Ma et al., 2021), in which an encoder-decoder model translates the input sequence of words to a special output, encoding named entities with predefined labels. We utilize RemBERT and mT5 as backbone models for these two approaches, respectively. Our results show that the oldie but goodie token classification outperforms the template-free method by a wide margin. Our code is available at: https://github.com/Abiks/MultiCoNER.

1 Introduction

This paper describes two submissions to the Multilingual Complex Named Entity Recognition (MultiCoNER) Shared Task, held by SemEval-2022 (Malmasi et al., 2022b). This shared task aims at recognizing named entities with the ambitious end goal of building systems that support up to 11 languages. Multilingual setups are complicated when the dataset mixes languages from different groups. This way, the MultiCoNER dataset comprises 11 languages from different families, and multiple scripts (Malmasi et al., 2022a). This setup hayj s become increasingly popular recently since it allows to test for transfer learning across languages within a single pre-trained model. The dataset, proposed in the shared task, has several unique features previously neglected in NER evaluation: syntactically complex entities, ambiguous entities, divers, and low-frequency (aka long-tail) entities. This makes the shared task setup closer to real-life settings, where datasets are way noisier and nonuniformly distributed.

Our solution consists of two state-of-the-art approaches adopted to the task. First, we use a mainstream NER technique, token classification. Under this approach, the model is trained to assign a label to each input token. The second approach falls into the group of prompt-based techniques, at the core of which are the capabilities of auto-regressive language models to memorize and reproduce input texts. In this case, we train an encoder-decoder model to replace entities in the input sentence with predefined labels. We utilize RemBERT (Chung et al., 2021) and mT5 (Xue et al., 2021) as backbone models for these two approaches, respectively. These two models provide state-of-the-art results for the common test-beds of cross-lingual experiments (Hu et al., 2020).

In other words, we explore the following questions: 'How do pre-trained transformer-based models perform in the Multilingual Complex Named Entity Recognition task?' and 'Which of the two approaches perform better?'.

Our results show that plain fine-tuning of the above-mentioned state-of-the-art multilingual transformer-based models can give moderate results. Results are within 10% margin of the top solution for the multilingual NER task. We analyzed the errors of both models. As expected, for two novel 'complex' entity types ('Creative Work' and 'Product') error rate is higher than for 'common' types ('Group', 'Location', 'Person'). However, some of the errors might be caused by inconsistencies in the labeling of the MultiCoNER dataset. The performance of the Template-free approach suffers from such inconsistencies more than the performance of the Token classification approach.

2 Related Work

Named Entity Recognition is one of the central tasks in Natural Language Processing, which attracts a lot of research efforts. Yang et al. (2016) encode morphology and context information via character and word embeddings. Recent studies (Ghaddar and Langlais, 2018; Jie and Lu, 2019; Liu et al., 2019; Meng et al., 2021) employ syntactic dependencies, lexical similarity, gazetteers, etc. in the word representations before feeding them to context encoding layers. The authors show that additional information may lead to improvements in NER performance. However, NER still faces multiple challenges (Li et al., 2022) such as detection of fine-grained and nested named entities (Kim and Kim, 2021; Ringland et al., 2019; Loukachevitch et al., 2021), NER in domain-specific areas (Weber et al., 2021), NER from noisy data (Derczynski et al., 2017) and code-mixed data (Fetahu et al., 2021).

Recent research in this area considers not only standard types of entities (*person, location, organization*) but also semantically ambiguous and complex entities (Hanselowski et al., 2018). For example, a system has to recognize the titles of movies, books, or songs, which may contain verbs, adverbs, prepositions, etc. Cui et al. (2021) propose a template-based method, treating NER as a language model ranking problem in a seq2seq manner. Original sentences and statement templates filled by a candidate named entity span are the source sequence and the target sequence, respectively. Ma et al. (2021) induces a language model to predict label words at entity positions during fine-tuning. This method demonstrates the effectiveness under the few-shot setting.

The SemEval-2020 shared task MultiCoNER (Malmasi et al., 2022a) focuses on a more exciting and challenging problem of building a NER system for multiple languages. The results of a recent Multilingual Named Entity Challenge in six Slavic languages (Piskorski et al., 2021) have also confirmed the complexity and significance of the task. Training competitive multilingual NER systems requires either manually labeled text collections or large automatically annotated datasets (Nothman et al., 2013).

3 Experiments

3.1 Token Classification Approach

Our baseline is to treat the NER task as a token classification problem. The base model is a pre-trained RemBERT (Chung et al., 2021) with a linear layer on top of the hidden-states output.

RemBERT is based on a multilingual BERT bidirectional transformer architecture. It uses decoupled embeddings, which allows changing the size of input and output embeddings. The input embeddings are reduced in size, thus making the fine-tuning process faster without performance loss compared to BERT.

In the first experiment on a tokenization step the original label is propagated to all of the word tokens. We fine-tuned the model on multilingual data for three epochs to predict the labels in BIO format. The batch size is 32, Adam optimizer is used with the learning rate of 10^{-5} and the scheduler decreases the learning rate by 0.1 each epoch. The metrics obtained on the development set for this approach are presented in Table 1.

In the following experiment we investigated how the performance changes if we combine the models into an ensemble. The ensemble consists of three models trained with different seeds as described above. The final predictions are made based on a hard or soft voting scheme. The models perform very similarly; the differences in evaluation scores are insignificant.

The confusion matrix in Figure 1 is built for the multilingual model. The largest ratio of erroneously assigned O labels is PROD (Product) or CW (Creative work). Almost for all individual languages, the confusion matrix is very similar to the aggregated one. The picture differs a bit for Farsi and Russian languages: the number of mislabelled entities as O is higher for each entity tag. This might be due to the difference in the language structure. For the Chinese language, 23% of the GRP (Group) entities were classified as CORP (Corporation). This behavior is unique to the Chinese

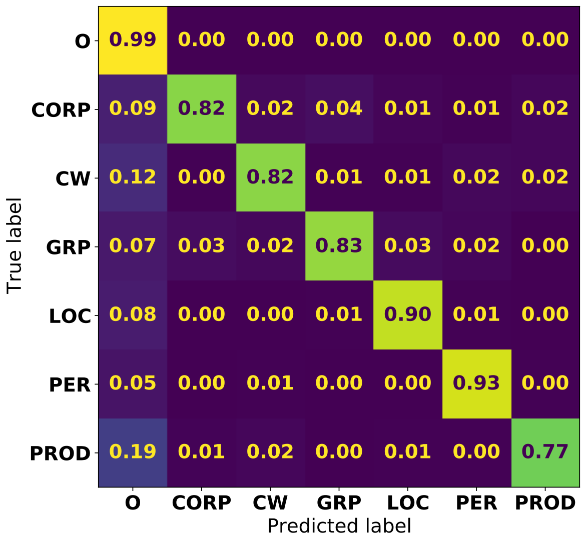


Figure 1: Confusion matrix for fine-tuned RemBERT

language.

The ensemble model does not introduce much improvement. However, training models with different seed values converge almost to the exact predictions. On the dev set, only one model's prediction out of three base models differs in 10% of the cases, and all three models have different predictions in 1% of the cases.

3.2 Template-free Approach

The second approach which we considered was Template-free (Ma et al., 2021). This approach showed decent results on CoNLL03 (Sang and Meulder, 2003) and MIT-Movie (Liu et al., 2013) datasets, so we decided to apply it to the multilingual data. The language model is trained to substitute named entity text spans with several predefined label words. For instance, given the sentence "*its headquarters are in sandy springs, united states of america*" we require the model to replace "*sandy springs*" and "*united states of america*" text spans with a predefined label word "*germany*" since these spans are considered as LOC (location) entities. In this case, the target for this example would be: "*its headquarters are in germany, germany*". After the model has substituted several text spans with the label words, we need to reconstruct these spans based on surrounding tokens to match each initial token with its predicted label. The label word for a specific entity class was chosen as the most frequent token of this class in the training data. Since we were working with multilingual data, we created a unique label words mapping for each language.

As the backbone model for the Template-free approach, we considered the mT5 pre-trained language model (Xue et al., 2021). mT5 is a multilingual variant of the T5 model that is pre-trained on a new Common Crawl-based dataset covering 101 languages. We chose two variants of the mT5 model for our experiments: mT5-Large and mT5-XL; the latter model showed better results. Due to the computational complexity, we could not run similar experiments with the mT5-XXL language model. Both models were trained with batch size equal to 8 and optimizer AdamW with learning rate $5 \cdot 10^{-5}$.

During the experiments with the Template-free approach, we encountered several challenges. Firstly, if the input phrase has two or more consecutive entity spans with a token length of more than one, it was impossible to reconstruct these spans unambiguously based on the sequence of predicted label words. In this case, we assigned the first $k-1$ predicted label words to $k-1$ input tokens, and the last label word was assigned to the rest of the tokens. Secondly, because of the punctuation issues occurring in some languages described in more detail in Section 4, it was difficult to generalize the reconstruction rules for all languages.

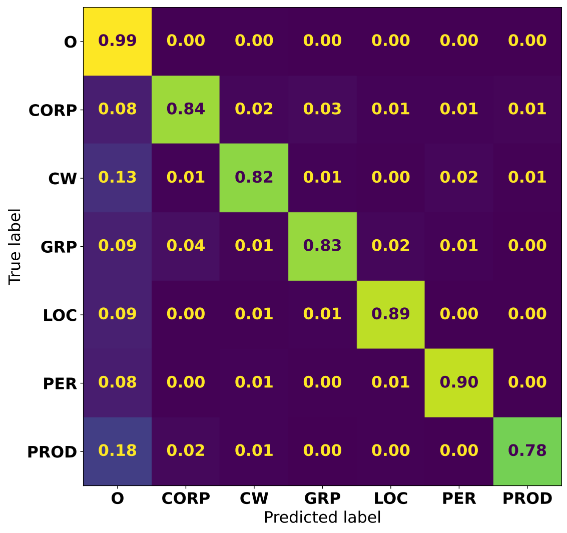


Figure 2: Confusion matrix for Template-free mT5-XL

The token-level confusion matrix for the Template-free approach on the development set is shown on Figure 2. The results for the mT5-XL fine-tuned model on the dev set is presented in Table 1.

Based on Figure 2 we can mention that the most

Table 1: Results for Token Classification and Template-free approaches on dev and test sets

Approach	Dev			Test		
	P	R	F1	P	R	F1
Token Classification	0.81	**0.84**	**0.82**	0.83	0.80	0.81
Template-free	**0.82**	0.77	0.79	0.58	0.51	0.54

common model's error is predicting the O-tag for tokens which are actually considered as parts of the entities. In other words, the model more often fails to recognize a named entity than confuses two different entities. We can also mention that in terms of languages the fine-tuned model shows the best results for English, Dutch and German and the worst results for Farsi and Russian according to F1 score. The dramatic performance downgrade on the test set compared to the dev set for the Template-free model may occur due to the significant distribution shift in the test data. For instance, the average sentence length in terms of tokens in the train and dev data is approximately equal to 16.4. However, the average length in the test data is equal to 9.6, we assume that the Template-free model could not be resistant for such changes.

4 Discussion

We provide the performance analysis of the models that we developed. The confusion matrices for both approaches demonstrate similar commonly occurring errors. Moreover, while working on experiments, we noticed several dataset issues which we suppose are worth mentioning. The problems potentially contribute to the low performance of the Template-free approach.

1. Classification decisions for tokens from PROD and CW entity types are more often confused with the O labeled tokens (Fig. 1, 2). These entities are examples of complex entities, which the competition was focused on. At the same time, the labeling for the two entity types in the provided dataset shows low consistency. For example, in the sentence[1] "*rice - long, medium, or short-grain white; also popcorn rice*" the first occurrence of "*rice*" is labelled as PROD. However, "*popcorn rice*" is not labeled as a named entity. The precise definitions of the complex entities and consistent labeling might be crucial for developing high-performing models for complex entity recognition. Another example is country names. In one sentence a country name can be considered as a named entity, but in another sentence a country name is annotated as O token. For instance, in the sentence[2] "*spain has an embassy ...*" the token "*spain*" has O label, but in the sentence "*in madrid, spain ...*" the same token is considered as a LOC (location) entity despite the fact that in both contexts this token refers to country name. Despite, we call such cases "mislabeled-as-O entities", not all of them can be considered wrong labels. As far as we have noticed, this problem is very common for the Farsi language. To be precise, we calculated the average number of mislabeled-as-O entities for each language (see Table 2). We suppose that this issue can influence the performance of both models.

Language	**Mislabeled-as-O**
BN-Bangla	0.691
DE-German	0.474
EN-English	1.002
ES-Spanish	3.061
FA-Farsi	20.234
HI-Hindi	1.995
KO-Korean	5.489
NL-Dutch	3.269
RU-Russian	1.606
TR-Turkish	3.494
ZH-Chinese	1.094

Table 2: Mean number of mislabeled-as-O entities

2. The second issue is that we noticed while working on the Template-free approach is punctuation labeling. For example, in the sentence[3] "*thomas earnshaw, inventor of ...*" the comma is presented as a separate token. However, in the sentence "*... museum of fine arts, houston ...*" the comma is part of token "*arts,*". This issue was crucial on the reconstruction entity spans step since the tokenizer always considers any punctuation token as a separate one. This issue is common for MultiCoNER data in many languages, especially in German and Bangla.

[1]The example sentence ID: 2e9d398c-a956-4419-a48d-f53790d2d237 (file en_train.conll)

[2]The example sentence IDs are df4360c4-a483-493b-bd93-87814db0104c and 475fb6b2-b9aa-4ec8-8b58-443f5e2774e8 (file en_train.conll)

[3]The example sentence IDs are be16705b-7f6e-4c28-b086-5eabf5950d29 and ea916f1b-b9a5-4959-9537-aeb875c1faf1 (file en_train.conll)

3. For the purposes of more detailed performance analysis we considered the dependence between prediction accuracy and entity length (in number of tokens). The Figure 3 shows this dependence for both Template-free and Token Classification approaches considering the dev dataset. On the one hand, the mT5-XL model, trained according to the Template free approach, performs better on longer entities compared to the RemBert model. On the other hand, the RemBert model, trained that exemplifies the Token Classification approach, shows better results on shorter entities. This could be the reason why the Token Classification method outperforms the Template free approach on the test set since the average sentence length in the test data is dramatically less than the average sentence length in the train and dev datasets.

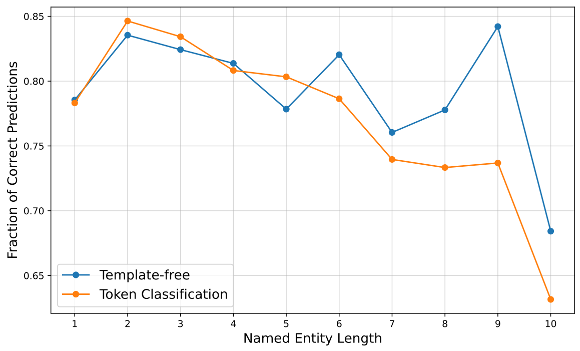


Figure 3: The dependence of prediction accuracy on the entity lengths

5 Conclusion

In the paper, we implemented and evaluated two straightforward yet different approaches to the multilingual NER subtask with complex types of entities (MultiCoNER). The first approach uses a state-of-the-art transformer-based model and fine-tuning for token classification, while the second one applies a template-free information extraction paradigm. Our results are within the 10% margin of the top solution for multilingual NER tasks and much higher than organizers' baseline performance. Therefore, we can conclude that out-of-the-box approaches generalize quite well for complex NER tasks and provide a viable alternative. The performance of the template-free approach can suffer from inconsistent annotation. The second finding is the relatively low performance of an ensemble model, but this issue needs further investigation. Evaluation of systems for multilingual NER with complex entity types is still challenging. Our study analyzed the dataset and found several issues that can be addressed in future versions of MultiCoNER Track or in similar competitions.

Acknowledgements

The project is supported by the Russian Science Foundation, grant # 20-11-20166. This research was supported in part through computational resources of HPC facilities at HSE University (Kostenetskiy et al., 2021).

References

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2021. Rethinking embedding coupling in pre-trained language models. In *International Conference on Learning Representations*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845. Association for Computational Linguistics.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Abbas Ghaddar and Phillippe Langlais. 2018. Robust lexical features for improved neural network named-entity recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1896–1907. Association for Computational Linguistics.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Zhanming Jie and Wei Lu. 2019. Dependency-guided lstm-crf for named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, page 3862–3872.

Hongjin Kim and Harksoo Kim. 2021. Fine-grained named entity recognition using a multi-stacked feature fusion and dual-stacked output in korean. *Applied Sciences*, 11(22):10795.

PS Kostenetskiy, RA Chulkevich, and VI Kozyrev. 2021. Hpc resources of the higher school of economics. In *Journal of Physics: Conference Series*, volume 1740, page 012050. IOP Publishing.

J. Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34:50–70.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and James Glass. 2013. Query understanding enhanced by hierarchical parsing structures. pages 72–77.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5301–5307. Association for Computational Linguistics.

Natalia Loukachevitch, Ekaterina Artemova, Tatiana Batura, Pavel Braslavski, Ilia Denisov, Vladimir Ivanov, Suresh Manandhar, Alexander Pugachev, and Elena Tutubalina. 2021. Nerel: A russian dataset with nested named entities, relations and events. In *Proceedings of Recent Advances in Natural Language Processing*, page 876–885.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. 2013. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151–175.

Jakub Piskorski, Bogdan Babych, Zara Kancheva, Olga Kanishcheva, Maria Lebedeva, Michał Marcińczuk, Preslav Nakov, Petya Osenova, Lidia Pivovarova, Senja Pollak, Pavel Přibáň, Ivaylo Radev, Marko Robnik-Sikonja, Vasyl Starko, Josef Steinberger, and Roman Yangarber. 2021. Slav-NER: the 3rd cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, pages 122–133. Association for Computational Linguistics.

Nicky Ringland, Xiang Dai, Ben Hachey, Sarvnaz Karimi, Cecile Paris, and James R Curran. 2019. NNE: A dataset for nested named entity recognition in english newswire. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5176–5181.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050.

Leon Weber, Mario Sänger, Jannes Münchmeyer, Maryam Habibi, Ulf Leser, and Alan Akbik. 2021. Hunflair: an easy-to-use tool for state-of-the-art biomedical named entity recognition. *Bioinformatics*, 37(17):2792–2794.

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

Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2016. Multi-task cross-lingual sequence tagging from scratch. *arXiv preprint arXiv:1603.06270*.

Raccoons at SemEval-2022 Task 11: Leveraging Concatenated Word embeddings for Named Entity Recognition

Atharvan Dogra
Thapar University, Patiala, India
atharvandogra007@gmail.com

Prabsimran Kaur
Thapar University, Patiala, India
pkaur_be18@thapar.edu

Guneet Singh Kohli
Thapar University, Patiala, India
guneetsk99@gmail.com

Jatin Bedi
Thapar University, Patiala, India
jatin.bedi@thapar.edu

Abstract

Named Entity Recognition (NER), an essential subtask in NLP that identifies text belonging to predefined semantics such as a person, location, organization, drug, time, clinical procedure, biological protein, etc. NER plays a vital role in various fields such as information extraction, question answering, and machine translation. This paper describes our participating system run to the Named entity recognition and classification shared task SemEval-2022. The task is motivated towards detecting semantically ambiguous and complex entities in short and low-context settings. Our team focused on improving entity recognition by improving the word embeddings. We concatenated the word representations from State-of-the-art language models and passed them to find the best representation through a reinforcement trainer. Our results highlight the improvements achieved by various embedding concatenations.

1 Introduction

Named entity recognition and classification is an essential subtask in natural language processing (NLP). The task of identifying named entities like a person, location, organization, drug, time, clinical procedure, biological protein, etc. in a text document is key to many applications, including information extraction and retrieval, machine translation(Al-Onaizan and Knight, 2002; Steinberger et al., 2013), topic modeling(Newman et al., 2006), text summarization(Schiffman et al., 2002), and question answering. It can also be used in domain-specific entity types such as disease, symptoms, and treatments. Recently many researchers have identified the name identification issue in a variety of languages and have made efforts to apply named entity recognition.

Complex and ambiguous Named entities, like the title of creative works such as books, songs, and movies, are not simple nouns and are thus harder to recognize(Ashwini and Choi, 2014). Unlike the traditional NEs these works take the form of linguistic constituents such as "Remember me when we are parted," which is an imperative clause and does not look like the traditional NEs (Locations, Person name, organizations). This ambiguity in syntax makes it difficult to identify them based on their context. There can also occur instances where these titles are semantically ambiguous; for example, "the girl on the train" can be a preposition or the name of a book. Such entities are growing at a faster rate as compared to the traditional categories. Therefore, processing these NEs has always been a challenging task in NLP in practical and open-domain settings. However, it has received enough engagement from the research community.

Despite the high score produced by Neural models on benchmark datasets like CoNLL03/OntoNotes, it has been noticed (Augenstein et al., 2017) that these models perform significantly low on complex or unseen data. This happens because the scores were driven by the presence of easy entities, well-formed text, and memorization due to entity overlap between train/test sets. Various researchers have noted that the majority of the errors in their downstream tasks have occurred due to the failure of NER systems to recognize complex entities. Various researchers using NER on downstream tasks have noted that a significant proportion of their errors are due to NER systems failing to recognize complex entities.

This paper presents our system for Shared Task on "MultiCoNER Multilingual Complex Named Entity Recognition @ SemEval 2022" (Malmasi et al., 2022b), (Meng et al., 2021), (Fetahu et al., 2021). The task focused on detecting complex and ambiguous entities in short and low-context settings. Our team focused on improving entity recognition by improving the word representations, following (Wang et al., 2021; Yamada et al., 2020). We concatenated the word embeddings from State-of-the-art language models and passed them to find the best representation through a reinforcement trainer.

2 Related Work

Name entity recognition has always been a challenging task that requires massive prior knowledge resources for good performance (Ratinov and Roth, 2009; Nadeau and Sekine, 2007). Research in various domains using a variety of approaches has been done. Early methods included heuristics and handcrafted rules, lexicons, orthographic features, and ontologies. These systems involved recognizing entities based on pattern matching with input documents(Rau, 1991; Collins and Singer, 1999). A higher degree of accuracy may be achieved using the rule-based system. However, it is time-consuming and challenging to port the developed rules from one domain to another.

Later machine learning methods were used (Borthwick, 1999; McCallum and Li, 2003; Takeuchi and Collier, 2002) . This learning technique involved supervised and unsupervised approaches. The supervised being a dominant technique for named entity recognition (Nadeau and Sekine, 2007). However, this method required massive high-quality annotated data. There also exist some hybrid models that make use of different rule-based and/or learning methods for enhanced performance (Srihari, 2000; Rocktäschel et al., 2012). These systems make the best use of good features and methods for improved performance.

Recently neural network systems (Collobert et al., 2011) have also become a popular method. These systems with minimal feature engineering have become appealing because they do not require sources such as lexicons or ontologies, thus making them domain independent. Most of the neural architectures proposed are based on recurrent neural networks (RNN).

3 Dataset Description

The English dataset provided by SemEval 2022 (Malmasi et al., 2022a) comprises 233,917 instances of data. The training sample included 15300 annotated statements, the dev set comprised 800 annotated sentences, and the test set included 217817 test data (as shown in the table). The sentences in the data are allocated a respective id. Each sentence is further divided into tokens which are separated by a newline. Each token is assigned a label according to the standard BIO tagging followed by the named entity tag in the specified format. *<token>_<BIO_tag>-<NE>*

In the BIO tagging, B and I tags are followed by <NE> tag, while O tags have no following tag. As presented in the fig, an instance of the dataset, statements with the 'O' tag represents a non-named entity. The statements with 'B' tags represent the beginning of the named entity, and 'I' tags represent the continuation of the same-named entity. The tokens are assigned one of the six named entities Person (abbreviated as PER), Location (abbreviated as LOC), Group (abbreviated as GRP), Corporation (abbreviated as CORP), Product (abbreviated as PROD), Creative Work (abbreviated as CW).

4 Methodology

The method used focuses on automating the process of finding better concatenation of embeddings for improved performance. In this approach, a task model and controller module repeatedly interact. To achieve high accuracy, the controller searches for a better embedding concatenation from the given set of pre-trained embeddings. The task model, on the other hand, produces a task output. The architecture works on a reward basis. The controller is rewarded every time the task model is trained over the task data after an embedding is generated. The controller receives a reward for updating its parameter and sampling new embedding concatenations. The general architecture of our approach is shown in Figure 1.

4.1 Design of Task Model

For the task model, a sequence-structure approach is used. Considering the input sentence to be *m* and the structured output to be *n*, the probability distribution *P(n|m)* can be calculated as:

$$P(n|m) = \frac{exp(Score(m,n))}{\sum_{n' \in N(m)} exp(Score(m,n'))} \quad (1)$$

where *N(m)* represents all possible output structures given the input sentence *n*. The BiLSTM-CRF model (Ma and Hovy, 2016; Lample et al., 2016) was used for sequence-structured outputs.

$$P^{seq}(n|m) = BiLSTM - CRF(E, n) \quad (2)$$

where $E = [e_1;\ e_2;\ ...\ e_n], E\ \in \mathbb{R}^{d \times w}$ is matrix of word representation for the input sentence *m* comprising of *w* words, The hidden size of the concatenation of embeddings is represented by *d*.

E_i is the word representation of the i^{th} word, which is a concatenation of L types of word embeddings:

$$e_i^l = embed_i^l(m); \;\; e_i = [e_i^1;\; e_i^2;\; ...\; e_i^L] \qquad (3)$$

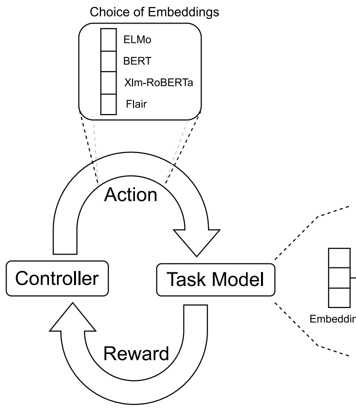


Figure 1: Architecture of our approach. It shows the task model, returning of reward function and the embeddings being concatenated

4.2 Design of Search Space

A set of neural networks can be used to represent neural architecture search space (Elsken et al., 2017). In this method, we have represented the embedding candidates as nodes. The sentence n is passed as the input to the nodes, and the resulting output is the embeddings e^l. Since the embeddings are concatenated as word representations, there is no connection between the nodes in search space, resulting in a significantly reduced search space.

A variety of embeddings are available for the extraction of word representation, for instance, finetuned XLM-RoBERTa-large embeddings.However, due to the restriction on the use of a multilingual model for a single language track, we finetuned the BERT embeddings on the provided training data and concatenated the last four layers as word features applying mean pooling operation over them (Devlin et al., 2019).

Unlike (Kondratyuk and Straka, 2019), applying a weighted sum of all twelve layers did not significantly differ the empirical results. There was no significant difference between the XLM-RoBERTa CONLL-03 English embeddings and the Finetuned BERT-Large embeddings for the submitted predictions. Thus we used regular embeddings to reduce search space. Subsequently, each embedding only has a specific operation resulting in a search space that contains 2^L-1 possible combinations of nodes.

Except for the character embeddings, all the other weights of the pre-trained embedding candidates are fixed. The parameters of task models are shared at each iteration of the search in accordance with Neural Architecture Search (NAS). All the nodes in the graph are kept in the search space. Each node performs an operation to indicate whether the embedding is masked out. This is done to avoid deciding which node is to be kept, making weight sharing difficult. For this binary task.

For this task, a binary vector $o = [o_1, \ldots, o_l, \ldots, o_L]$ is used as a mask for those embeddings which are not selected:

$$e_i = [e_i^1 o_1;\; e_i^2 o_2;\; \ldots\; ; e_i^l o_l; \ldots ; e_i^L o_L] \qquad (4)$$

Where o_l represents a binary variable. Input E is applied to the linear layer in the BiLSTM layer hence the multiplication operation in E, *i.e.* multiplying mask with embedding, is equivalent to directly concatenating the selected embeddings:

$$W^T e_i = \sum W_l^T e_i^l o_l \qquad (5)$$

Also, the unused embedding candidates and the corresponding weights in W are removed for a light task-model after finding the best concatenation.

4.3 Searching in the Space

The controller is responsible for generating the embedding mask using parameters which are generated using $\theta = [\theta_1; \theta_2; ...; \theta_L]$. The probability distribution of selecting a concatenation o is $P^{ctrl}(q;\theta) = \Pi_{l=1}^{L} P_l^{ctrl}(o_l;\theta_l)$. Element o_l of $\mathbf{o}$ is sampled using Bernoulli distribution which is given by:

$$P_l^{ctrl}(o_l;\theta_l) = \begin{cases} \sigma(\theta_l), & o_l = 1. \\ 1 - P_l^{ctrl}(o_l = 1;\theta_l), & o_l = 0. \end{cases} \qquad (6)$$

where σ is the sigmoid function. Given the mask, the task model is trained until convergence and returns an accuracy R on the dev set. The reinforcement algorithm is used for optimization. The accuracy R is used as a reward signal for training the controller, whose aim is to maximize the reward $J(\theta) = \mathbb{E}_{P^{ctrl}(o;\theta)}[R]$ utilizing the policy of gradient method (Williams, 1992). The reward function that we have used:

$$r^t = \sum_{i=1}^{t-1} (R_t - R_i)\gamma^{Hamm(o^t,o^i)-1}|o^t - o^i| \qquad (7)$$

The final gradient comes out as:

$$\nabla_\theta J(\theta) \approx \sum_{l=1}^{L} \nabla_\theta .\, log P_l^{ctrl}(o_l^t;\theta_l)\, r_l^t \qquad (8)$$

This reward function evaluates how each embedding candidate contributes to the accuracy change. The binary vector $|o^t - o^i|$ represents the change in embeddings. o^t represents a binary vector at current iteration t, and o_i represents the previous time step i. The Hamming distance $Hamm(o^t, o^i)$ is used to measure the contribution of embedding candidates in the accuracy. As hamming distance increases, the contribution becomes less significant.

4.4 Training the controller

The corresponding validation scores and concatenations were stored in a dictionary D to train the controller. Firstly the task model was trained with all the embedding candidates concatenated. Then the concatenation o^t was sampled based on Equation (6). The task model was trained using the equation (4) after which the model is evaluated on a development set to return an accuracy R_t. Then the gradient is computed for the controller following Equation (8) using the concatenation o_t, accuracy R_t and D. Based on the computed gradient, the parameters of the controller are updated. Finally, o^t and R_t are added to D. If a concatenation o^t is in dictionary, we compare its accuracy with the value in the dictionary and keep the higher one. Selecting o^{t-1} (i.e., previous concatenation) and zero vector is avoided.

5 Experimental Setup

Data was provided in form of sentences which were broken down into individual words, or tokens, each lying in a newline and their corresponding tags in front of them. All the tokens are first passed through a function to encode them into embeddings, which are initial representations for the tokens. These tokens are pushed through the RNN language model which forms the task model and iteratively returns a reward to the controller.

Table 1: Major Hyperparameters

Parameters	Value
BiLSTM size	800
BiLSTM layer	1
Optimizer	SGD
Learning rate	[0.1, 7.8125e-4]
Epochs	150
Episodes	20

This model first contains a dropout layer, then an encoder layer is added which contains the embedding. The next layer is an LSTM layer with a hidden size of 800. Finally, the representations are fed into a linear-chain CRF layer to predict the final label sequence, where a linear layer is applied for the representations to score each entity label. The above deep learning model has been built using PyTorch[1] along with the transformer embeddings that have been used.

The learning rate was set at 0.1 at the beginning with a patience level set to 5, i.e. the learning rate was halved if there was no improvement in monitored metrics for the patience, that is the validation loss. The batch size was set to 64 and a maximum of 150 epochs were allowed for 20 episodes.

6 Results

The table below shows the micro and macro F1 scores of prediction on the development set. As the tags of the test set are unavailable to us, the model's performance was judged on dev set only and the results of the best 3 models were ensembled in the end, for submission. The best performing model was achieved in a model, where the following embeddings were concatenated: Finetuned BERT-large-uncased, Finetuned RoBERTa-large, ELMo original, FastCharacterEmbeddings, Glove, FastWordEmbeddings-english, Flair news-en embedding. The model was trained on with the following hyperparameters setting 150 epochs, 20 episodes, SGD optimizer, 800 hidden units of BiLSTM-CRF, starting with a learning rate of 0.1 and a batch size of 64.

Table 2: Results with concatenated BERT embeddings

Concatenations	Micro-F1	Macro-F1
TransformerWordEmbedding: BERT-large-uncased ELMoEmbedding: original || FastCharacterEmbeddings FastWordEmbedding-0: glove || FastWordEmbeddings-1: en FlairEmbedding: news-forward	**0.8867**	**0.8745**
TransformerWordEmbedding: BERT-large-cased ELMoEmbedding: original || FastCharacterEmbeddings FastWordEmbedding-0: glove || FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8596	0.8470
TransformerWordEmbedding: BERT-base-cased ELMoEmbedding: original || FastCharacterEmbeddings FastWordEmbedding-0: glove || FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8597	0.847
TransformerWordEmbedding: BERT-base-uncased ELMoEmbedding: original || FastCharacterEmbeddings FastWordEmbedding-0: glove || FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8861	0.8693

[1]https://pytorch.org/

Table 3: Results with concatenated Xlm-RoBERTa embeddings

height **Concatenations**	**Micro-F1**	**Macro-F1**
TransformerWordEmbedding: Xlm-RoBERTa-large-finetuned-conll03-english (finetuned on current dataset) ELMoEmbedding: original ‖ FastCharacterEmbeddings FastWordEmbedding-0: glove ‖ FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.868	0.8536
TransformerWordEmbedding: Xlm-RoBERTa-large-finetuned-conll03-english ELMoEmbedding: original ‖ FastCharacterEmbeddings FastWordEmbedding-0: glove ‖ FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8563	0.8423
TransformerWordEmbedding-0: Xlm-RoBERTa-large-finetuned-conll03-english TransformerWordEmbedding-1: RoBERTa-large ELMoEmbedding: original ‖ FastCharacterEmbeddings FastWordEmbedding-0: glove ‖ FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8728	0.8592
TransformerWordEmbedding-0: Xlm-RoBERTa-large-finetuned-conll03-english TransformerWordEmbedding-1: BERT-large-uncased ELMoEmbedding: original ‖ FastCharacterEmbeddings FastWordEmbedding-0: glove ‖ FastWordEmbeddings-1: en FlairEmbedding: news-forward	**0.8997**	**0.8881**
TransformerWordEmbedding: Xlm-RoBERTa-base ELMoEmbedding: original ‖ FastCharacterEmbeddings FastWordEmbedding-0: glove ‖ FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8465	0.8332
TransformerWordEmbedding: Xlm-RoBERTa-large ELMoEmbedding: original ‖ FastCharacterEmbeddings FastWordEmbedding-0: glove ‖ FastWordEmbeddings-1: en FlairEmbedding: news-forward	0.8596	0.8456

These results were ensembled with two other models which were developed as follows:

- BERT-base-uncased, BERT-large-uncased, and RoBERTa-large with the rest of the embeddings as it is.
- Finetuned BERT-large-uncased with the rest of the embeddings as it is.

7 Result Analysis

Experiments were conducted to analyze the performance of systems by using different combinations of embeddings. The best concatenation of embeddings can be easily seen in the model containing BERT-large-uncase transformer embedding and ELMo original, FastCharacterEmbeddings, GloVe FastWord: english, and Flair: news-forward with a Macro-F1 score of 0.8745 on the dev set. This model generated the best results after ensembling the results with the models containing BERT-base-uncased embeddings and BERT-large-uncased finetuned on the current dataset, which generated a Macro-F1 score of **0.7418**[2] on the test dataset.

Table 4: Comparison in baseline and our best model

Models	**Micro-F1 Score**[3]
Baseline	0.715
Our Best	0.8867

One clear distinction in the result can be seen among the BERT uncased and cased models. In the BERT-uncased models, the text is set to lowercase before the WordPiece tokenization step, hence case is irrelevant in it, while in BERT-cased models, the case of words is also considered. In our dataset, the tokens were already lowercase.

For comparison's sake, we have experimented with RoBERTa embeddings as well and they have produced slightly better results than BERT embeddings alone. One difference in the functioning of BERT and RoBERTa was noticed during experimenting with their tokenizers. The BERT tokenizer sometimes separated hyphen-separated entities and treated them as individual entities while RoBERTa tokenizer treated them as a combined single entity and generated a representation for it.

For comparison, we have also experimented with XLMR embeddings during result compilation to show the difference (or similarity) between results. After removing a few of the other embeddings like ELMo, Flair, and Glove, we have also collected results from the above model. All the results can be referred to in Table 3.

8 Conclusion and Future work

In this paper, we present our approach to SemEval-2022 Task 11:MultiCoNER Multilingual Complex Named Entity Recognition. Our best submission gave us an F_1 score of **0.7418**, placing us 10^{th} on the Evaluation Phase Leaderboard. Future work includes experimenting with multilingual data and embeddings and different optimizers.

[2]This is the final Macro-F1 score on the test set

[3]Micro-F1 score was the only metric available for baseline

References

Yaser Al-Onaizan and Kevin Knight. 2002. Translating named entities using monolingual and bilingual resources. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 400–408.

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications. *arXiv preprint arXiv:1704.02853*.

Andrew Eliot Borthwick. 1999. *A maximum entropy approach to named entity recognition*. New York University.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *1999 Joint SIGDAT conference on empirical methods in natural language processing and very large corpora*.

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

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

Thomas Elsken, Jan-Hendrik Metzen, and Frank Hutter. 2017. Simple and efficient architecture search for convolutional neural networks. *arXiv preprint arXiv:1711.04528*.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

David Newman, Chaitanya Chemudugunta, and Padhraic Smyth. 2006. Statistical entity-topic models. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 680–686.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155.

Lisa F Rau. 1991. Extracting company names from text. In *Proceedings the Seventh IEEE Conference on Artificial Intelligence Application*, pages 29–30. IEEE Computer Society.

Tim Rocktäschel, Michael Weidlich, and Ulf Leser. 2012. Chemspot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640.

Barry Schiffman, Ani Nenkova, and Kathleen McKeown. 2002. Experiments in multidocument summarization.

Rohini K Srihari. 2000. A hybrid approach for named entity and sub-type tagging. In *Sixth Applied Natural Language Processing Conference*, pages 247–254.

Ralf Steinberger, Bruno Pouliquen, Mijail Kabadjov, and Erik Van der Goot. 2013. Jrc-names: A freely available, highly multilingual named entity resource. *arXiv preprint arXiv:1309.6162*.

Koichi Takeuchi and Nigel Collier. 2002. Use of support vector machines in extended named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Automated concatenation of embeddings for structured prediction.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

SeqL at SemEval-2022 Task 11: An Ensemble of Transformer Based Models for Complex Named Entity Recognition Task

Fadi Hassan[1], **Wondimagegnhue Tufa**[2], **Guillem Collell**[1],
Piek Vossen[2], **Lisa Beinborn**[2], **Adrian Flanagan**[1], and **Kuan Eeik Tan**[1]

[1]Helsinki Research Center, Europe Cloud Service Competence Center
Huawei Technologies Oy (Finland) Co. Ltd., Helsinki, Finland
{firstname.lastname}@huawei.com
[2]Faculty of Humanities, Vrije Universiteit Amsterdam
{w.t.tufa,p.t.j.m.vossen,l.beinborn}@vu.nl

Abstract

In this paper, we present a system for detecting complex named entities in multilingual and code-mix settings. We discuss the results obtained in task 11 (MultiCoNER) of the SemEval 2022 competition. The model is an ensemble of various transformer-based language models combined with a Conditional Random Field (CRF) layer. Our model ranks fourth in track 12 (multilingual track) and fifth in track 13 (code-mixed track). We describe the details of our model implementation and discuss the effect of different aggregation methods. Finally, we conduct additional analyses to understand the performance differences between languages.

1 Introduction

Named Entity Recognition (NER) is the task of identifying proper names in a text and categorizing them into predefined entity types such as person (PER), location (LOC), or creative work (CW). For example, given a sentence *"Michael Jeffrey Jordan was born in Brooklyn, New York"* the goal is to label entities correctly with their corresponding category using the BIO-scheme:

"Michael [B-PER], Jeffrey [I-PER], Jordan [I-PER] was born in Brooklyn [B-LOC], New [I-LOC] York [I-LOC]"

Existing systems are often trained on standard news text and strongly rely on surface form features such as capitalization and punctuation. These approaches do not scale well to user-generated content because of the increased variation in language and context in an ever-expanding domain (Meng et al., 2021; Fetahu et al., 2021; Augenstein et al., 2017).

Examples of challenging scenarios for named entity recognition are: (a) Entities in very short text inputs such as search queries with limited or no context (Meng et al., 2021) (b) Structurally complex entities such as movie or book titles ranging from complex noun phrases to full clauses (c) Recognizing named entities in dynamically evolving contexts in which novel entities emerge (Augenstein et al., 2017; Aguilar et al., 2019).

Although many named entities are shared between languages, named entity recognition systems usually rely on language-specific cues (e.g. capitalization of nouns in German, compounding phenomena in agglutinative languages such as Korean, Japanese and Turkish (Agerri and Rigau, 2016)). Such kind of detection cues do not scale well to other languages. As a consequence, most models need to be fine-tuned for each language separately on manually annotated high-quality training data which is a costly process.

Task 12 of SemEval 2022 provides a test bench for more robust systems which can detect complex named entities in 11 languages (Malmasi et al., 2022b). The dataset intentionally contains semantically ambiguous entities with limited contexts. We focus on the multilingual tracks of the competition which requires the prediction of named entities in all 11 languages by a single model (track 12). As an additional challenge, the model is also evaluated on code-mixed data (track 13).

We summarize the main finding of our analysis as follows:

- In Sec 4.2, we show that the choice of the tagging scheme affects model performance. We observe that BILOU Tagging is more effective than BIO Tagging in our experiment albeit the total number of training is reduced when annotation is changed to BILOU. We hypothesize that this is due to explicit distinction between single and multi-token entities in BILOU which might help model performance.

- In Sec 6.1, we compare the performance differences across languages and find a surprisingly large difference between German and Russian (10-point difference in f1). This is contrary to what is expected since the size of the training data and the label distribution across languages are comparable. We hypothesize that linguistic factors such as script and typology or factors relating to the pre-trained phase contribute to this difference.

- In Sec 6.3, we further expand on this hypothesis and experiment with a zero-shot model to examine patterns of transfer between pairs of languages. We observe higher transfer between English, Dutch and German. These are also the languages for which the model yields the highest scores individually.

In the rest of the paper, we discuss related work, experimental setup and model training, and extensive error analysis.

2 Related Work

Named entity recognition can be modeled as a sequence labeling problem. Deep-learning based approaches learn suitable representations in an end-to-end fashion and outperform rule-based and handcrafted feature-based approaches (Akbik et al., 2018; Wang et al., 2020).

(Huang et al., 2015) proposed a BiLSTM-CRF architecture for sequence tagging which is used by most state-of-the-art models. These models combine long short-term memory layers in a bidirectional fashion to use both past and future input and predict named entity sequences using a conditional random field layer. Significant performance gains have been obtained by initializing the model with pre-trained contextual embedding models such as BERT (Devlin et al., 2019), Flair (Akbik et al., 2018) and LUKE (Yamada et al., 2020).

Subsequent works explore some of the limitation of using a vanilla transformer. (Guo et al., 2019) show that a transformer architecture is less effective for modeling sequence labeling that strongly relies on left and right context and long-range dependencies which is the case for named entity recognition. LUKE (Yamada et al., 2020) is the state of the art in the CoNLL-03 NER dataset. It is pre-trained by contextualized representations based on bi-directional transformers on entity-annotated corpus of words and entities.

The complexity of the task and its multilingual nature are the main factors in choosing our modeling approach. The task complexity entails that our approach should rely on token context and in capturing relationships between labels since the surface form cues (e.g capitalization) are normalized in the training data. The multilingual aspect entails choosing a crosslingual pre-trained model which can handle the target languages. We choose `XLM-RoBERTa-large` (Conneau et al., 2019) and `Microsoft/infoxlm-large` (Chi et al., 2020). `XLM-RoBERTa-large` model is a cross-lingual version of RoBERTa. XLM-RoBERTa has outperformed cross-lingual BERT and it is the state of the art on many cross-lingual tasks including Named Entity Recognition. `Microsoft/infoxlm-large` is similarly a multilingual pre-trained model for over 100 languages with a new cross-lingual pre-training task named cross-lingual contrast (XLCO).

A comparison of the two pre-trained models shows both to be competitive on tasks such as cross-lingual natural language inference (XNLI) and `Microsoft/infoxlm-large` to be significantly better on cross-lingual question answering (MLQA) and cross-lingual sentence retrieval on the Tatoeba dataset. We provide a direct comparison of these two models for named entity recognition (which was previously missing in the literature) and explore an ensemble of the two models.

3 Data analysis

We use the training dataset provided as part of SemEval 2022 Task 11 MultiCoNER: Track 12 (Multilingual) and Track 13 (Code-mixed) (Malmasi et al., 2022a). The dataset consists of training and development data in 11 languages annotated with six named entity types. Table 1 provides statistical characteristics of the dataset. We observe that overall 18% of the tokens are labeled as a named entity which is comparable to other datasets. In terms of entity types, Person (PER), Group (GRP) and Creative Works (CW) occur more frequently across languages. We notice that the absolute number of entity tokens is twice as high for Chinese as for the other languages. This can be explained by the character-level tokenization of the Chinese texts. As a consequence, 98% of all Chinese entities are multi-token entities compared to 55% for Korean and 85% for English. Figure 1 visualizes the entity density across languages showing a large difference for Chinese but only small variations for the other languages. This difference may be smoothed by subtoken representations of the language models. The Chinese characters can not be broken any further, whereas the other language tokens can.

	Multilingual	Code-Mixed
NER Entity(#)	6	6
Language (#)	11	N/A
Sentences (#)	168.3 K	1.5 K
Tokens (#)	2750.9 K	17.5 K
Part of Entity (%)	18	30
Outside of Entity (%)	82	70

Table 1: Summary of training data statistics

4 System Description

The system that we proposed for both track 12 and track 13 is based on an ensemble of two pre-trained transformer models (PTMs). The first one is the `XLM-RoBERTa-large` model (Liu et al., 2020; Conneau et al., 2019) which is a cross-lingual version of RoBERTa.The second one is `Microsoft/infoxlm-large` (Chi et al., 2020) which is also multilingual pre-trained model that supports over 100 languages and includes a new cross-lingual pre-training.

4.1 Fine-tuning

Both the selected pre-trained model takes as input a sequence of tokens and encodes them to the embedding space. During fine-tuning. These embeddings are passed to a dense layer that predicts class scores. On top of the class scores, we used a CRF layer.

4.2 Tagging Schemes

Several NER tagging schemes have been used in the literature. However, choosing the ideal scheme is a complex problem (Konkol and Konopík, 2015). The two most popular NER tagging schemes are BIO and BILOU. In BIO, sometimes referred to as IOB (Sang and Buchholz, 2000), a different tag is assigned to each word in the text depending on whether it is the beginning $(B - y)$, inside $(I - y)$, or outside (O) a named entity phrase y. In case of BILOU, in addition to the previous $(B - y)$, $(I - y)$ and (O) tags, words at the end of an entity phrase get an end tag $(E - y)$ and single-token entities get a unit-length tag $(U - y)$. BILOU annotations increase the amount of information related to the boundaries of named entities compared to BIO but reduce the amount of training cases per tag.

4.3 Ensemble

In our experiments on the development set, we get the best performance using an ensemble of seven models. Four of them are based on `XLM-RoBERTa-large`, and the other three are based on `Microsoft/infoxlm-large`. Furthermore, to make the set of models more diverse, we used a different random seed to initialize their weights and while we kept the same set of hyper-parameters as defined in Table 2. Finally, we used two different ensemble techniques, explained in more detail in the following sections. We provide additional information about the ensemble models in Appendix A.

4.4 Voting and Score Fusion

A hard voting ensemble involves summing the votes for crisp (discretized) class labels from our models and predicting the class with the most votes. While *soft voting* is an ensemble that involves summing the predicted probabilities for class labels and predicting the class label with the largest sum probability. To consider the context of the labels, we employed a CRF layer on top of the aggregated scores. See Figure 2.

5 Experiment and Result

All our models were implemented with PyTorch (Paszke et al., 2019), on top of the pre-trained transformer models provided by HuggingFace (Wolf et al., 2019). For the PTMs, the output of the last attention layer was used as input for the classifier layer. The CRF classifier was implemented using the AllenNLP library (Gardner et al., 2017). Adam optimizer (Kingma and Ba, 2014) was used to update model parameters. Finally, cosine annealing decay with $T\_max = 20$ and $eta\_min = 1.0e-8$ was applied for the learning rate after (early_stopping_patience / 2) consecutive epochs without improving (Loshchilov and Hutter, 2016).

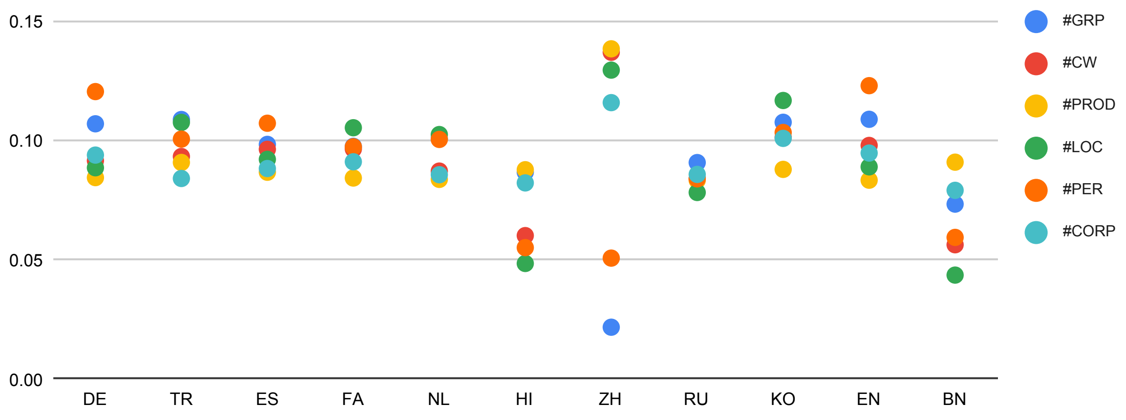


Figure 1: Named Entities Phrase Density

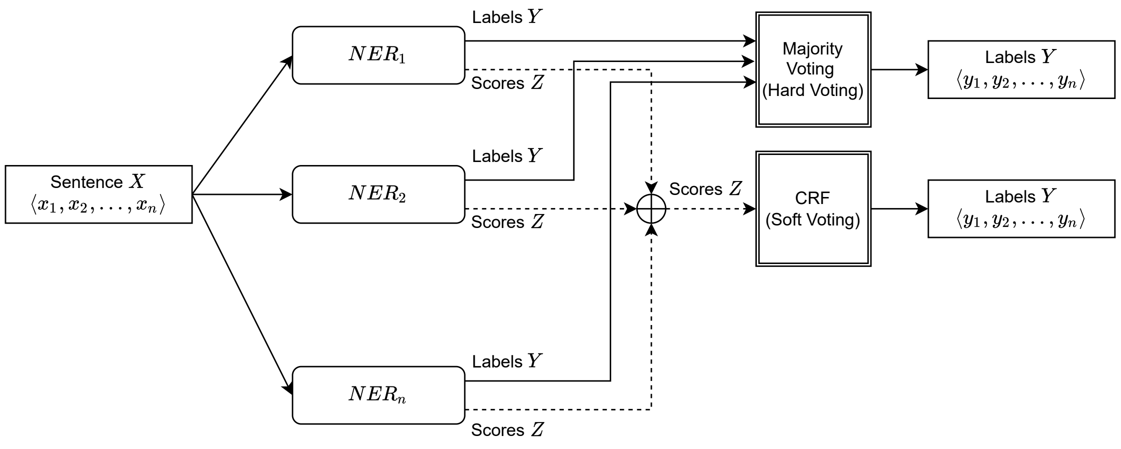

Sentence X $\langle x_1, x_2, \dots, x_n \rangle$
NER_1
NER_2
NER_n
Labels Y
Scores Z
Majority Voting (Hard Voting)
CRF (Soft Voting)
Labels Y $\langle y_1, y_2, \dots, y_n \rangle$

Figure 2: Voting

Hyperparameter	value
Attention output	Last layer
Curriculum learning	Sorting by # of tokens
Padding	Batch padding
Tagging scheme	BILOU
Max sequence length	128
Batch size	32
Learning rate	5.0e-6
Learning rate decay	cosine annealing
Early stopping patience	6
Early stopping metric	Macro span-F1
Optimizer	Adam
Loss	Viterbi

Table 2: Optimizer and hyperparameters used to fine-tune our model

5.1 Model Training and Evaluation

We train our model using only the official training data. We used the development data to evaluate the performance of the models. Models were evaluated at the end of every epoch. Early stopping and cosine annealing decay were determined using the macro average span-F1 score. In the evaluation phase, task organizers provided an unlabelled test dataset. We used the pre-trained model to make the predictions without retraining and uploaded it to Codalab. The ensemble of models with score fusion provides the best results on the development and test datasets. See tables 3 and 4.

Hyperparameter selection impact During the training phase, we tested several combinations of the hyperparameters, and we used a greedy approach to select the best individual hyperparameters. Table 2 shows the most important parameters that have a significant impact on the performance

Track →	Track 12 (Multilingual)			Track 13 (Code-Mixed)		
Model ↓	P	R	F1	P	R	F1
Baseline	64.5	65.6	64.2	60.0	61.7	59.0
Best infoxlm-large	86.4	87.1	86.8	76.9	76.5	76.7
Best XLM-R-large	86.4	86.4	86.4	78.2	76.8	77.3
Ensemble voting	87.7	**87.4**	87.6	79.4	78.1	78.7
Ensemble fusion	**88.6**	87.0	**87.8**	**81.8**	**78.5**	**80.0**

Table 3: NER results on the development datasets in span-level precision (P), recall (R) and F1 in %.

of our models.

A crucial first step was the application of curriculum learning which reduced the training time by 50%. That was crucial since training our model on such a big dataset takes about one hour per epoch.

The tagging scheme is one of the most critical parameters that impact the performance of our model. For example, using BILOU scheme improved the performance about 1.5% on span-F1 score compared with the BIO scheme.

Max sequence length and batch size played a primary role in the training speed and performance. At the same time, a small value for the learning rate prevented the model from over-fitting rapidly. Finally, learning rate decay made the model convergence smoother before the early stopping occurrence.

PTM Impact As shown in Table 3, `Microsoft/infoxlm-large` and `XLM-RoBERTa-large` have almost the same performance. To the best of our knowledge, they share the same structure and are pre-trained on the same data. However, their pre-training objective functions differ. On the other hand, model size impact can be clearly seen by comparing the above large models with the `baseline` model which is based on `XLM-RoBERTa-base`.

Ensemble Impact The *score fusion* (Sect. 4.3) ensemble outperforms the individual models and the *vote-based* ensemble on almost all metrics. This improvement was due to the soft score aggregation, which gives the model better control to select the correct class than the crisp *vote-based* class aggregation.

CRF Impact Applying CRF on model scores gives better results than using argmax only. However, it was a bit hard to apply it in the score fusion ensemble model. In this type of ensemble, we aggregated the scores, not the output of the CRF layer. There were two options to solve this problem, i) take the CRF layer of one of the ensemble models and use it directly on top of the aggregated scores without fine-tuning, or ii) fine-tune a new CRF layer on the aggregated scores. We tried both solutions, and our finding was that the second option gives better performance, about 0.1% improvement in span-F1 score compared with the first option. See last row in Table 3.

6 Analysis and Conclusion

In this section, we analyze our model output for the multilingual task and the code-mixed task on the development dataset because the gold labels for the test data were not released. Table 5 shows the performance averaged over six entity types ranked by language.

We see that the model performance varies strongly between languages. The best result is obtained for German and is 10 percentage points higher than the lowest result which is obtained for Russian. When we compare the different entity types, we observe the highest variance for Chinese.

The large differences are surprising as the training data size is equal for all languages and the entity types are roughly evenly distributed (with the exception of Chinese). We therefore analyse these differences further below.

6.1 Variation Across Language

We first clustered languages into three groups based on their model performance: Group-1 has a score of 0.9 or higher and includes German, Dutch and English. Group-2 has a score between 0.85 and 0.9 and includes Turkish, Chinese, Spanish, Korean, Hindi and Bangla. Group-3 has a score lower than 0.85 and includes Farsi and Russian. With a single language model and a comparable

Track → Model ↓	Track 12 (Multilingual) P	R	F1	Track 13 (Code-Mixed) P	R	F1
Ensemble voting	74.64	75.84	74.92	79.72	79.58	79.57
Ensemble fusion	75.96	75.78	75.49	81.10	79.72	80.29

Table 4: NER results on the test datasets in span-level precision (P), recall (R) and F1 in %.

Language	DE	NL	EN	TR	ZH	ES	KO	HI	BN	FA	RU
Macro-F1	92.28	91.43	90.26	88.71	88.11	87.62	86.47	86.19	86.11	84.54	82.92

Table 5: Macro-F1 (in %) averaged over NEs (evaluated on development dataset)

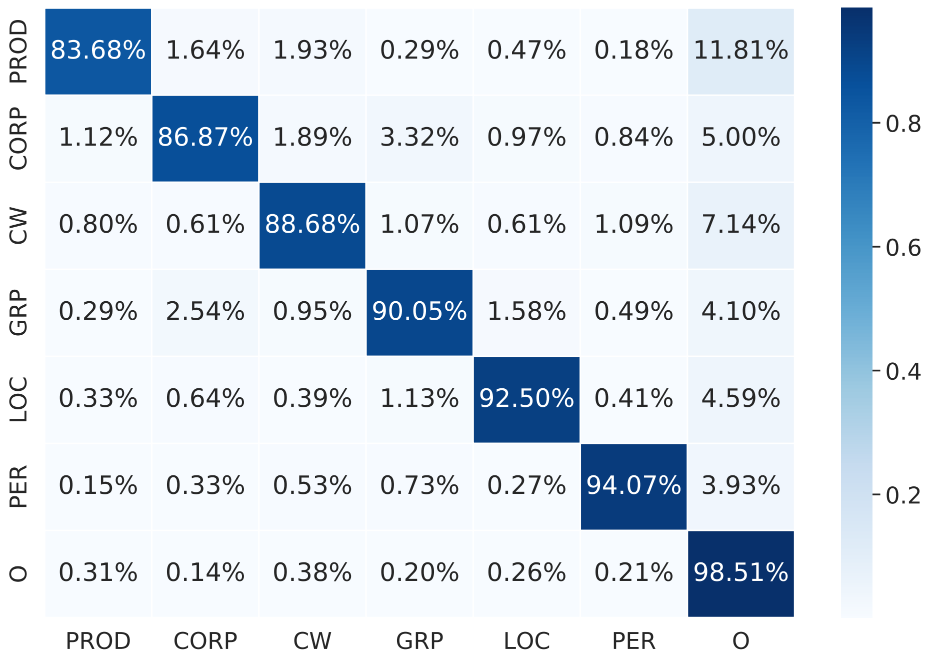


Figure 3: Confusion Matrices on the development datasets (Multilingual + Code-Mixed)

fine-tuning dataset across languages, the disparity in the result could possibly be attributed to
(a) Representational quality: differences in language representations in XLM-R, as some languages are represented better than others
(b) Typological properties: typological difference between languages as languages that are related tend to take advantage of transfer during fine-tuning
(c) Script characteristics: languages with similar scripts tend to take advantage of shared subtokens during fine-tuning.

We observe that all group-1 languages are commonly categorized as high-resource languages for which `XLM-RoBERTa-base` performs well on downstream tasks (Conneau et al., 2019; Joshi et al., 2020). Liu et al. (2020); Hu et al. (2020) give evidence for this effect in downstream tasks.

In terms of typological properties, although group-1 and group-3 languages are members of the same language family, they differ in their scripts which to some extent negatively influence transfer at least during the fine-tuning phase (Muller et al., 2021).

6.2 Confusion Matrix

Figure 3, shows a confusion matrix for coarse-grained entity types. From these results, we observe a stronger ambiguity between the O label and the entity types than across entity types. A possible explanation can be lexical overlap in the training data. The highest confusion can be observed between Creative Work (CW) and the Out Label (O). CW often consists of titles of movies and other creative work that include words that also occur in regular expressions annotated as O.

Frequency Analysis As a follow-up, we carried out a more-detailed frequency analysis of the tokens that are annotated as entity tokens and O. Frequency analysis on the token shows a long tail distribution with more than 90% of the errors occurring only once. Table 6 shows the most frequently misclassified tokens, which are the English and Dutch definite determiner and the Chinese symbol for *Sri Lanka*.

Token	Frequency
De	36
the	22
斯	22

Table 6: Tokens which are misclassified most frequently

From the misclassified tokens, we observed substantial overlap between false negative and false positive tokens. Among these, determiners (articles such as 'a', 'the', 'de') and words that stand for the type of products are common ("movie", "series", "municipal"). These words are typically expected at the border of named entity expressions. There may be two explanations for these cases:

Inconsistent Annotations where a token is sometimes included in the named entity expression and sometimes it is not. This issue most likely occurs on border labels. To show this we take annotation examples from the training data e.g., *the oculus quest* as the name of a product can be annotated as [O, B-PROD, I-PROD] or [B-PROD, I-PROD, I-PROD] where *the* is annotated as outside of entity type in the first case and inside on the second case which creates inconsistency on *the* token.

Variable Contexts where the same token truly occurs both within entity phrases in some context and outside as the context change. For example in *the communist party of great britain* is annotated as [O, B-GRP, I-GRP, I-GRP, I-GRP, I-GRP] where the token *great* is annotated as GRP and in a different context - *the great sum of 1,000 pounds* it is annotated as outside of entity. Both cases are difficult to resolve for a model.

Token	Annotated Labels	Label Distribution
his	I-CW, O, I-PER	[0.01, 0.99, 0.00]
songs	I-CW, O, B-CW	[0.05, 0.83, 0.12]
since	I-CW, O, B-CW	[0.01, 0.99, 0.01]

Table 7: Examples for label variation

We explored the first cause where the issue might arise from an inconsistent annotation in the training data. We first extract tokens from the training data with multiple labels along with their corresponding label proportion. In Table 7 some examples are given, where "songs" also tend to occur at the beginning of a CW. We then use this proportion to create a post processing filter where we "correct" the model output at the borders of entity phrases for tokens with overlapping False Positive and False Negative cases in case the model output deviates from the bias.
We experimented with different thresholds for the bias to apply where the maximum value represents a bias value and the minimum value represents the exception value. Although this approach did not result in a performance gain or loss, we observe that closed class words such as articles but also proper names for locations, product names, proper nouns and symbols are often affected by this filter. A possible explanation for lack of effect could be that the same annotation inconsistency also applies to the test data.

6.3 Transferability

To analyze how transfer plays out between the group-1 languages, we fine-tuned `XLM-RoBERTa-base` on the English dataset and evaluated it in a zero-shot setting on the rest of the languages. Figure 4 shows the result of this experiment. Though other factors might play a role, we can infer from this result that group-1 languages (German and Dutch) have a more positive transfer from English than the other languages, although Spanish also benefits from English.

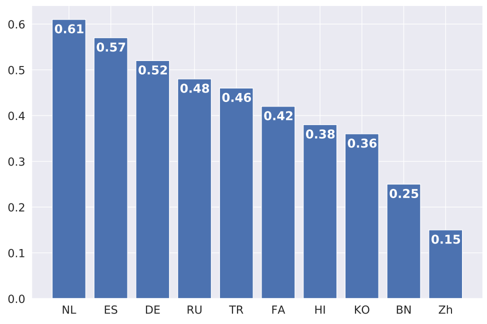


Figure 4: English Zero shot performance across languages

	NL	NL-DE	NL-ZH	NL-Rand ZH	Den-ZH	Den-DE
GRP	0.82	0.8	0.76	0.76	0.02	0.11
CW	0.72	0.68	0.67	0.71	0.14	0.09
PROD	0.69	0.62	0.61	0.69	0.14	0.08
LOC	0.87	0.88	0.84	0.86	0.13	0.09
PER	0.88	0.82	0.86	0.84	0.05	0.12
CORP	0.84	0.78	0.77	0.76	0.12	0.09

Table 8: Bilingual Models evaluated on Dutch Development Dataset (Macro F1)
The last two Column (Den-ZH and Den-DE) shows Named entity density measures

Next to zero-shot transferability, we also experimented with bilingual transfer. We selected one language as the target language, in this case, Dutch, and we measured the contribution of all other languages as training data in addition to half of the Dutch training data. We combine half of the dutch training data with half of German (Column NL-DE), half of Chinese(Column NL-ZH) and randomized Chinese (Column NL-Rand ZH) where we randomize Chinese tokens with a token from XLM vocabulary. The first four column shows bilingual models evaluated on Dutch development set measured in macro-averaged F1. The total set of training sentences was kept the same across all experiments.
We observe that contrary to the zero-shot results, Chinese contributes overall only just a bit lower than German when tested on the Dutch test set. This is remarkable because German and Dutch are typologically very close and use the same script. Apparently, the observed density of entities for Chinese is a factor that may compensate for the difference in script and language typology. We can see in Table 8 that the contributions of Chinese lag behind when the density is lower than German (GRP) but is almost the same when it is higher (CW, PROD, LOC, CORP). The only exception is PER which has the lowest density for Chinese but still a higher contribution. To test the assumption that just the label density plays a role, we even replaced the Chinese tokens with random tokens. The results show that even partially randomized Chinese training data outperforms the German contribution on most entity types.

6.4 Conclusion

In this paper, we proposed a single named entity recognition system that can process multilingual and code-mixed text based on an ensemble of transformer-based models. We have accomplished fourth and fifth positions in the test phase for track 12 (Multilingual) and track 13 (Code-Mixed). Even though the proposed system performs pretty well on the development dataset, there is a considerable performance drop in track 12 on the test dataset. Further study needs to be done to address that performance change.

Summarising the results from the error analysis and the statistics on the training data, we can conclude that there are four factors that play a role in the cross-lingual performance of this task, given that an equal amount of training data is available for fine-tuning in all languages. We provided evidence that transfer from the XLM pre-training, typological relatedness, and shared scripts can be factors that contribute to transfer but on the other hand the density of the entities in the training data is another factor.

Our system does not include external gazetteers or targeted unsupervised learning on difficult entity types such as products and creative works. In future work, we would like to include them, which could help to improve the performance due to the extra information they include.

Acknowledgements

The research by the authors of affiliation 1 and by Wondimagegnhue Tufa was funded by Huawei Finland.

References

Rodrigo Agerri and German Rigau. 2016. Robust multilingual named entity recognition with shallow semi-supervised features. *Artificial Intelligence*, 238:63–82.

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2019. A multi-task approach for named entity recognition in social media data. *arXiv preprint arXiv:1906.04135*.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *CoRR*, abs/1701.02877.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2020. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. *arXiv preprint arXiv:2007.07834*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1315–1325, Minneapolis, Minnesota. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *ArXiv*, abs/1508.01991.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Michal Konkol and Miloslav Konopík. 2015. Segment representations in named entity recognition. In *International Conference on Text, Speech, and Dialogue*, pages 61–70. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

Ilya Loshchilov and Frank Hutter. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Benjamin Muller, Antonios Anastasopoulos, Benoît Sagot, and Djamé Seddah. 2021. When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 448–462, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward

Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Erik F Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. *arXiv preprint cs/0009008*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. More embeddings, better sequence labelers? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3992–4006, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

A Ensemble Models

In Table 9 and Table 10, we provide score distribution of all models that form our ensemble on the development datasets.

#	PTM	Random Seed	P	R	F1
1	microsoft/infoxlm-large	102	87.6	85.6	86.6
2	microsoft/infoxlm-large	2022	86.4	87.1	86.8
3	microsoft/infoxlm-large	2033	87.5	85.7	86.6
4	xlm-roberta-large	2044	85.8	86.5	86.2
5	xlm-roberta-large	2055	86.4	86.4	86.4
6	xlm-roberta-large	2066	85.9	86.8	86.4
7	xlm-roberta-large	2077	86.2	86.4	86.3

Table 9: Ensemble models - Multilingual

#	PTM	Random Seed	P	R	F1
1	microsoft/infoxlm-large	102	77.6	75.4	76.5
2	microsoft/infoxlm-large	2022	76.9	76.5	76.7
3	microsoft/infoxlm-large	2033	78.1	74.9	76.5
4	xlm-roberta-large	2044	78.2	76.8	77.3
5	xlm-roberta-large	2055	78.2	76.8	77.3
6	xlm-roberta-large	2066	77.3	77.1	77.2
7	xlm-roberta-large	2077	77.1	76.1	76.6

Table 10: Ensemble models - Code-Mixed

SFE-AI at SemEval-2022 Task 11: Low-Resource Named Entity Recognition using Large Pre-trained Language Models

Changyu Hou[1], **Jun Wang**[1*], **Yixuan Qiao**[1], **Peng Jiang**[1], **Peng Gao**[1], **Guotong Xie**[1*]
1 Ping An Healthcare Technology, Beijing, China
junwang.deeplearning@gmail.com
Qizhi Lin[2], **Xiaopeng Wang**[2], **Xiandi Jiang**[2], **Benqi Wang**[2], **Qifeng Xiao**[2]
2 Ping An Puhui Enterprise Management Co. Ltd, Shanghai, China
wangxiaopeng069@1u.com

Abstract

Large scale pre-training models have been widely used in named entity recognition (NER) tasks. However, model ensemble through parameter averaging or voting can not give full play to the differentiation advantages of different models, especially in the open domain. This paper describes our NER system in the SemEval 2022 task11: MultiCoNER. We proposed an effective system to adaptively ensemble pre-trained language models by a Transformer layer. By assigning different weights to each model for different inputs, we adopted the Transformer layer to integrate the advantages of diverse models effectively. Experimental results show that our method achieves superior performances in Farsi and Dutch.

1 Introduction

NER is an essential tool in the application fields of information extraction, question answering system, syntactic analysis, machine translation, etc. It plays a vital role in the process of the practical application of natural language processing technology (Devlin et al., 2018; Meng et al., 2021). However, processing complex and ambiguous Named Entities (NEs) is a challenging NLP task in practical and open-domain settings, but has not received sufficient attention from the research community.

Complex NEs, like the titles of creative works (movie/book/song/software names), are not simple nouns and are harder to recognize (Ashwini and Choi, 2014; Fetahu et al., 2021). They can take the form of any linguistic constituent, like an imperative clause (“Dial M for Murder”), and do not look like traditional NEs (Person names, locations, organizations). This syntactic ambiguity makes it challenging to recognize them based on their context. Such titles can also be semantically ambiguous, e.g., “On the Beach” can be a preposition or refer to a movie. Finally, such entities usually grow faster than traditional categories, and emerging entities pose yet another challenge.

In recent years, the deep learning methods have achieved great success in NLP (Natural Language Processing). However, some limits need to be addressed. Challenges for deep learning in NLP mainly arise from the scarcity of labeled data. One way to alleviate the need for large labeled datasets is to pre-train a model on unlabeled data via self-supervised learning, and then transfer the learned model to downstream tasks (Devlin et al., 2018). These methods have been widely applied and have made a massive breakthroughs, such as BERT (Devlin et al., 2018), XLNET (Yang et al., 2019) and GPT (Radford et al., 2018).

Pre-trained neural models (e.g., Transformers) have produced high scores on benchmark datasets like CoNLL03/OntoNotes (Devlin et al., 2018). However, as noted by Augenstein et al. (Augenstein et al., 2017), these scores are driven by the use of well-formed news text, the presence of “easy” entities (person names), and memorization due to entity overlap between train/test sets; these models perform significantly worse on complex/unseen entities. Researchers using NER on downstream tasks have noted that a significant proportion of their errors are due to NER systems failing to recognize complex entities (Luken et al., 2018; Hanselowski et al., 2018).

2 Related Work

NER. Named entity recognition (NER) is a subtask of information extraction, which aims to locate and classify named entities in text into predefined categories. The traditional NER scheme is rule recognition based on manual induction. Based on domain dictionary and grammar rules. Lample (Lample et al., 2016) explored neural structures for NER, in which the bidirectional LSTMs are combined with CRFs with features based on character-based word representations and unsuper-

*Corresponding Author.

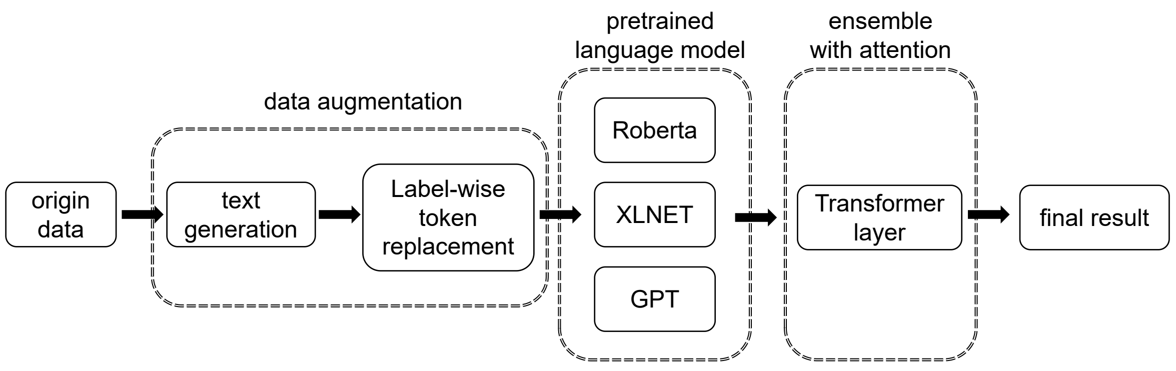


Figure 1: Overview of our architecture for the NER task. The system mainly includes two parts: data augmentation, pre-trained language model ensemble.

vised word representations. Despite BiLSTM's great success in the NER task, it has to compute token representations one by one, which massively hinders full exploitation of GPU's parallelism. Therefore, CNN has been proposed to encode words concurrently. In order to enlarge the receptive field of CNNs, iterative dilated CNNs (IDCNN) (ḞẎu and Koltun, 2016) was used.

Pre-training Language Model. Recent large-scale language model pre-training methods such as BERT (Devlin et al., 2018), XLNET (Yan et al., 2021) and GPT (Radford et al., 2018), further boosted the performance of NER, yielding state-of-the-art performances.

3 Task Descriptions

This task challenges NLP enthusiasts to develop complex Named Entity Recognition systems for 11 languages (In this work, we conducted experiments on Farsi and Dutch). The task focuses on detecting semantically ambiguous and complex entities in short and low-context settings. The task also aims at testing the domain adaption capability of the systems by testing additional test sets on questions and short search queries.

4 Datasets

For the entity recognition task (Malmasi et al., 2022b) on both Farsi and Dutch, there are 15300 pieces of training data. For Farsi, the longest sentence has 46 words, and the shortest sentence has only 1 word, with an average length of 18 words. For Dutch, the longest sentence has 47 words, and the shortest sentence has 2 words, with an average length of 15 words. This means that the length of the current corpus is too short, which means that it is difficult to use a large number of sentence models to cover a large proportion of the length of the current corpus.

In both tasks, there are six types of entities (Malmasi et al., 2022a), CW (Creative Work), PER (Person), LOC (Location), GRP (Group), CORP (Corporation), PROD (Product). The distribution of entity types is relatively average. The lowest proportion is PROD, accounting for 12.9%, and the highest proportion is LOC, accounting for 24.9%. Many types of entities, short sentence length, lack of sufficient context information, and open-domain information extraction all increase the difficulty of these tasks.

5 Systems Description

Overview of our architecture for the NER task was shown in Figure 1. The system mainly includes two parts: data augmentation, pre-trained language model ensemble.

5.1 Data Augmentation

For data augmentation, we expanded the existing training data through text generation and label-wise token replacement. Given a sentence s in the training dataset, we aim to generate a new sentence $s^{'''}$ containing the same type entities. Our augmentation consists of the following steps:

1. Entity Mask: Mask all entities in the sentence with specific symbol $[mask]$. We call this modified text $s^{'}$.

2. Sentence Generation: Generate similar sentences using the GPT model. We call this generated sentence $s^{''}$.

3. Entity Infilling: Fill in $[mask]$ tokens with text that has the same entity type. This final output text is called $s^{'''}$.

Besides, we also delete the non-entity part of the sentence randomly to improve the robustness of the model.

5.2 Model and Benchmarks

As shown in Figure 1, Roberta, XLNET, and GPT are chosen as the base model in out system. We use these large pre-trained models as the basis and then use the training data to fine-tune each one. After the fine-tune is completed, it is worth mentioning that, the outputs from the last fourth to the last second layer are used to average the parameters as the final output for each model. The reason why we did not use the last layer is to avoid over-fitting. There is a huge difference between the training dataset and the test dataset (16,000 training data, but 150,000 test data). In order to avoid overfitting on the training set and maintain the generalization ability, we did not use the output of the last layer.

Second, we take the results of all models as the input of the subsequent attention module, as demos in Figure 1. After obtaining the inputs of base models, we train a transformer layer to assign the corresponding weight to the outputs of different models at the same location, and use the weighted sum of all models as the final output after the model ensemble. Moreover, we add a CRF (Conditional Random Field) layer at the end to limit the final result.

The output of the basic model as shown in below:

$$\begin{bmatrix} r_1^1 & ... & r_i^1 & ... & r_L^1 \\ r_1^2 & ... & r_i^2 & ... & r_L^2 \\ r_1^3 & ... & r_i^3 & ... & r_L^3 \end{bmatrix} \tag{1}$$

where r_i represents the output of the ith model, L means the length of the sentence. Using this matrix as input, we train a transformer layer to learn how to assign different weights to different models. The weight matrix obtained is multiplied by the output results of all models to obtain a matrix with weight, as shown below:

$$\begin{bmatrix} a_1^1r_1^1 & ... & a_i^1r_i^1 & ... & a_L^1r_L^1 \\ a_1^2r_1^2 & ... & a_i^2r_i^2 & ... & a_L^2r_L^2 \\ a_1^3r_1^3 & ... & a_i^3r_i^3 & ... & a_L^3r_L^3 \end{bmatrix} \tag{2}$$

The vector corresponding to the position can be regarded as the influence of the model on the result of this position, so the final result corresponding to each position can be regarded as the weighted sum of the results of the three models, which is $R_i = [a_i^1r_i^1 + a_i^2r_i^2 + a_i^3r_i^3]$. This result is used as the final output of the model to predict labels.

Parameter	value
sequence length	128
batch size	24
learning rate	0.00003
CRF learning rate	0.001
Dropout	0.5
epoch	20

Table 1: Hyper-parameters of the model.

Model	ACC	Recall	F1
RoBerta	0.776	0.819	0.797
XLNET	0.791	0.816	0.804
GPT	0.799	0.764	0.781
Ensemble model w/o AUG	0.816	0.822	0.819
Ensemble model w/ AUG	**0.823**	**0.834**	**0.828**

Table 2: Benchmarks of Farsi on the dev set. The best performance is highlighted in bold. w/ AUG and w/o AUG denote model with and without data augmentation.

5.3 Experimental Setup

Our implementation is based on PyTorch and we conduct all experiments on one Tesla V100 GPU. Table 1 shows the details of the hyper-parameters for our models. The total training epoch is set to 20 and the batch size is set to 24. The initial learning rate is set to 3×10^{-5}, and the learning rate of CRF layer is set to 1×10^{-3}. Considering the average length of sentences in datesets, the sequence length is 128. Dropout (Srivastava et al., 2014) and early-stop are also used in our method to achieve better performance in the open domain.

Model	ACC	Recall	F1
RoBerta	0.894	0.892	0.893
XLNET	0.892	0.899	0.895
GPT	0.882	0.880	0.881
Ensemble model w/o AUG	0.904	0.909	0.906
Ensemble model w/ AUG	**0.910**	**0.911**	**0.910**

Table 3: Benchmarks of Dutch on the dev set. The best performance is highlighted in bold. w/ AUG and w/o AUG denote model with and without data augmentation.

6 Results and Discussion

The results on both Farsi and Dutch data-sets are shown in Table 2 and Table 3. Notably, our architecture with data augmentation and model ensemble has substantially improved performance. We also conducted the ablation study of different data augmentation ratios. We found that the best results can be obtained when the data is augmented with a ratio 2 + by three times (as shown in Table 4). Moreover, we also carried out experiments on whether post-processing can improve the performances. The post-processing was conducted as follows: based on the dictionary of the training set, we generated entity labels by exact string matching, where conflicted matches were resolved by maximizing the total number of matched tokens. Unfortunately, the final results demonstrated a relatively decreased performance about 0.4 percent. The reason lies in case-sensitive. In many cases, the case of letters plays a vital role in identifying entities. For example, 'Wat is' is a very prevalent query in Dutch, but 'IS' also represents an organization (Islamic State of Iraq and al-Sham).

Augmentation Ratio	ACC	Recall	F1
0 +	0.769	0.811	0.789
1 +	0.772	0.811	0.791
2 +	**0.776**	**0.819**	**0.797**
3 +	0.775	0.816	0.795

Table 4: Ablation study of different data augmentation ratios on the dev set of Farsi using RoBerta.

7 Conclusions

This study describes an effective NER system in the SemEval 2022 task 11: MultiCoNER, the experimental results demonstrated the effectiveness of our ensembled large pre-trained language models on low-resource Named Entity Recognition. In future work, we would further improve the performances by exploiting a more complicated ensemble strategy with more diversified models.

References

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

ḞẎu and V. Koltun. 2016. Multi-scale context aggregation by dilated convolutions.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. *arXiv preprint arXiv:1809.01479*.

G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. 2016. Neural architectures for named entity recognition.

Jackson Luken, Nanjiang Jiang, and Marie-Catherine de Marneffe. 2018. Qed: A fact verification system for the fever shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 156–160.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Rongen Yan, Xue Jiang, and Depeng Dang. 2021. Named entity recognition by using xlnet-bilstm-crf. *Neural Processing Letters*, (1).

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

NCUEE-NLP at SemEval-2022 Task 11: Chinese Named Entity Recognition Using the BERT-BiLSTM-CRF Model

Lung-Hao Lee, Chien-Huan Lu, and Tzu-Mi Lin
Department of Electrical Engineering
National Central University
No. 300, Zongda Rd., Zhongli Dist., Taoyan City 32001, Taiwan
lhlee@ee.ncu.edu.tw, 10952107@ncu.edu.tw,110521087@ncu.edu.tw

Abstract

This study describes the model design of the NCUEE-NLP system for the Chinese track of the SemEval-2022 MultiCoNER task. We use the BERT embedding for character representation and train the BiLSTM-CRF model to recognize complex named entities. A total of 21 teams participated in this track, with each team allowed a maximum of six submissions. Our best submission, with a macro-averaging F1-score of 0.7418, ranked the seventh position out of 21 teams.

1 Introduction

Named Entity Recognition (NER) is a fundamental task in information extraction that locates the mentions of named entities and classifies them (e.g., person, organization and location) in unstructured texts. The NER is a traditional NLP task that has been solved as a sequence labeling problem, where entity boundaries and category labels are jointly predicted. It is difficult to recognize complex named entities like the titles of creative works (e.g., books, songs, movies) that can take the form of any linguistic constituent (Ashwini and Choi, 2014). Syntactic and semantic ambiguity makes it challenging to recognize such complex named entities based on their context.

The SemEval-2022 Task 11 (MultiCoNER) organized a challenge to develop multilingual complex NER system for 11 human languages (Malmasi et al., 2022b). This task focuses on detecting semantically ambiguous and complex entities in short and low-context settings. The languages include: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla. The named entity categories are Person (labeled as PER), Location (LOC), Group (GRP), Corporation (CORP), Product (PROD), and Creative Work (CW). The task evaluation framework is divided in three broad tracks. 1) Multi-lingual (Track 1): participants train a single multi-lingual NER model for all the languages; 2) Mono-lingual (Track 2-12): participants train a model that works for only one language; 3) Code-mixed (Track 13): testing samples include tokens from any of the 11 mentioned languages in the shared task. We only participated the Track 9 for Chinese language.

Chinese NER is more difficult to process than English NER. Chinese language is logographic and provides no conventional features like capitalization. In addition, due to a lack of delimiters between characters, Chinese NER is correlated with word segmentation tasks, and named entity boundaries are also word boundaries. However, incorrectly segmented entity boundaries will cause error propagation in NER. For example, in a short context "這首歌出現在華特迪士尼動畫動物方城市中" (This song appeared in the Walt Disney animation Zootopia), a creative work "動物方城市" (Zootopia) may be incorrectly segmented into three words: "動物" (animal), "方"(square), and "城市" (city). Hence, it has been shown that character-based approaches outperform word-based methods for Chinese NER (He and Wang., 2008; Li et al., 2014; Zhang and Yang., 2018).

Recently, deep learning techniques have been widely used for Chinese NER, mostly with promising results. A character-based LSTM (Long Short-Term Memory)- CRF (Conditional Random Field) model with radical-level features was proposed for Chinese NER (Dong et al., 2016). The ME-CNER model exploited multiple embeddings-based character representation to improve Chinese NER performance (Xu et al., 2019). A joint training objective technique for different types of neural embeddings was adopted for Chinese NER in social media, based on Weibo messages (Peng and Dredze., 2015). The BiLSTM (Bidirectional LSTM)-CRF model was trained based on character-word mixed embeddings to improve the recognition effectiveness of Chinese NER (E and Xiang., 2017). A BiLSTM-CRF model with a self-attention mecha-

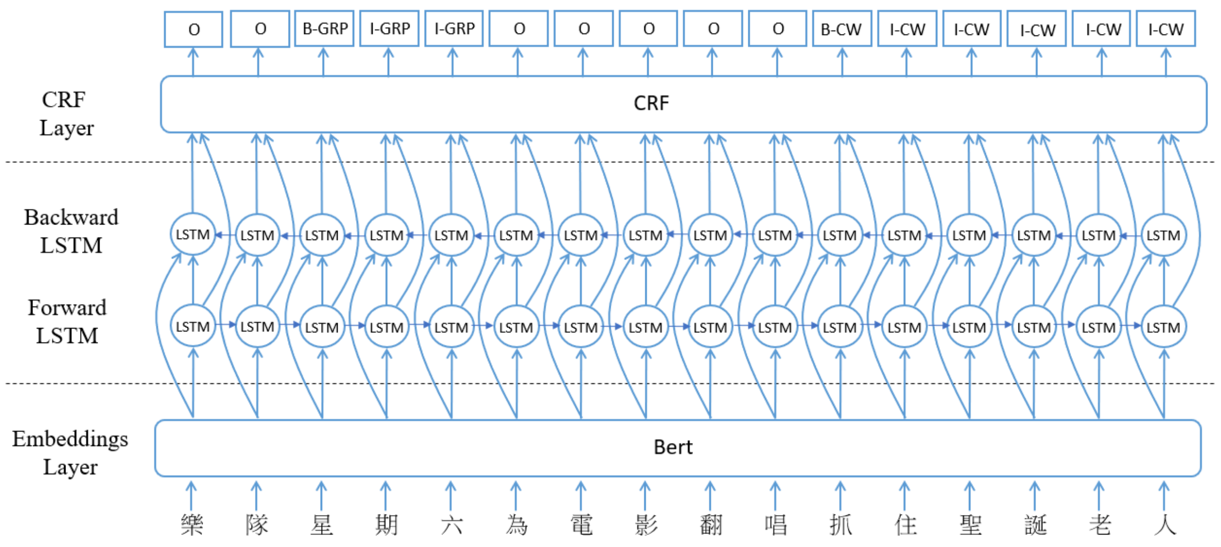


Figure 1: Our NCUEE-NLP system architecture for the Chinese track of SemEval-2022 Task 11.

nism was proposed to integrate part-of-speech labeling information to capture the semantic features of input sequences for Chinese clinical NER (Wu et al., 2019). A residual dilated CNN (Convolution Neural Network) with CRF was also presented to enhance Chinese clinical NER in terms of computational performance and training time (Qiu et al., 2019). An ME-MGNN (Multiple Embeddings enhanced Multi-Graph Neural Network) model was proposed to derive a character representation based on multiple embeddings at different granularities from the radical, character to word levels. Multiple gated graph sequence neural networks, along with standard BiLSTM-CRF, were then used to recognize Chinese named entities in the healthcare domain (Lee and Lu., 2021).

This paper describes the **NCUEE-NLP** (**N**ational **C**entral **U**niversity, Dept. of **E**lectrical **E**ngineering, **N**atural **L**anguage **P**rocessing Lab) system for the Chinese track of SemEval-2022 Task 11. We find that the neural computing approaches based on the BiLSTM-CRF achieved impressive results for Chinese NER. Hence, we follow the investigated results to develop character-based BiLSTM-CRF models.

2 The NCUEE-NLP System

Figure 1 shows our NCUEE-NLP system architecture for the Chinese NER. Our BERT-BiLSTM-CRF model is composed of three main parts: 1) BERT embeddings, 2) Bidirectional LSTM networks, and 3) CRF sequence labeling.

2.1 BERT Embeddings

Word embedding is a type of representation for text analysis that allows words with similar meanings to have similar representations in the form of a real-valued vector (Mikolov et al., 2013). Word embeddings can be obtained using a set of language modeling techniques where words are mapped to a low dimensional vector space of real numbers. Replacing static vectors, such as word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014)), and fastText (Bojanowski et al., 2017), with contextual word representations has led to significant improvements to virtually every NLP task (Ethayarajh., 2019). BERT (Bidirectional Encoder Representations from Transformer) (Devlin et al., 2019), is an encoder-decoder architecture that uses attention mechanisms to incorporate context into word embeddings. Its technical innovation lies in applying the bidirectional training of the transformer with masked language modeling to hide the partial words and infer them using their position information.

Since incorrect Chinese word segmentation may cause error propagation to affect the boundaries of named entities, we only use the last layer of BERT to obtain contextual embedding for each character.

2.2 Bidirectional LSTM Networks

In traditional neural network architectures such as multilayer perceptron, all the inputs and outputs are mutually independent. To address this issue, Recurrent Neural Networks (RNN) create networks with

Data Source		Sent.	All NE	#PER	#LOC	#GRP	#CORP	#PROD	#CW
Official	Training	15,300	23,717	2,221	6,984	710	3,756	4,818	5,228
	Validation	800	1,273	129	378	26	189	271	280
	Test	151,661	N/A	N/A	N/A	N/A	N/A	N/A	N/A
External	MSRA	16,302	36,088	10,378	25,710	-	-	-	-
	Weibo	490	891	648	243	-	-	-	-
	PD	9,281	21,238	6,256	14,982	-	-	-	-
	Boson	566	4,280	2,479	1,801	-	-	-	-
	CLUENER	2,305	3,136	-	-	-	-	-	3,136
	LG	6,135	16,266	-	-	-	-	-	16,266

Table 1: Data statistics of Chinese NER.

loops called Long Short-Term Memory (LSTM) to remember all information over time. Bidirectional LSTM (BiLSTM) (Graves et al., 2013) combines two independent RNNs that allows the networks to obtain both forward (from left to right) and backward (from right to left) information about the character sequence at every time step.

2.3 CRF Sequence Labeling

The learned feature representations of characters in the BiLSTM layer are then fed to a standard Conditional Random Field (CRF) (Lafferty et al., 2001), following the character order in the original sentence to predict the sequence of labels.

During the model training phase, a sentence represented in terms of a character sequence, along with the corresponding named entity labels, are used to train the BERT-BiLSTM-CRF model. We adopt the commonly used BIO (Beginning, Inside, and Outside) format. The B-prefix before a tag indicates that the character is the beginning of a named entity and an I-prefix before a tag indicates that the character is inside a named entity. An O tag indicates a character belongs to no named entity. For example, a sample sentence “樂隊星期六為電影翻唱抓住聖誕老人” (The band The Saturdays covered Christmas Wrapping for the movie.) in Figure 1, “星期六” (The Saturdays) are a British-Irish girl band that belongs to the Group (labeled as GRP) category. The corresponding named entity labels are “B-GRP,” “I-GRP,” and “I-GRP” for individual character “星”, “期”, and “六”. Similarly, “抓住聖誕老人” (Christmas Wrapping) is a song belonging to the Creative Work (CW) category, so we have the named entity labels “B-CW,” “I-CW”, “I-CW”, “I-CW”, “I-CW”, and “I-CW”.

During the testing phase, our trained BERT-BiLSTM-CRF model is used to predict the named entity label of each character for performance evaluation.

3 Experiments and Results

3.1 Data

Table 1 shows detailed statistics for mutually exclusive datasets. The experimental datasets were mainly provided by the task organizers (Malmasi et al., 2022a). The evaluation test set is about 10 times larger than original training dataset. In addition, according to an FAQ on the task website, the training and test data have dissimilar label distributions, though we have yet to obtain the real distribution in the test set. Hence, we also collected external data, including MSRA (Levow., 2006), Weibo (Peng and Dredze., 2015), People Daily (PD)[1] , Boson[2] , CLUENER (Xu et al., 2020), and LG [3] to train our model. The former four datasets consist of sentences annotated with the named entity categories Person (PER) and Location (LOC), while the latter two datasets were converted to contribute instances for the Creative Work (CW) category. We did not find sentences annotated with appropriate labels for the named entity categories Group (GRP), Corporation (CORP), and Product (PROD) as defined in this task.

[1]https://github.com/OYE93/Chinese-NLP-Corpus/tree/master/NER/People’s%20Daily

[2]https://static.bosonnlp.com/dev/resource

[3]https://github.com/LG-1/video_music_book_datasets

Models		**Precision**	**Recall**	**F1**
Embedding	**Data Usage**			
BERT	Official (training)	0.8856	**0.8759**	**0.8807**
	Official + External	0.8778	0.8579	0.8677
RoBERTa	Official (training)	**0.8867**	0.8618	0.8741
	Official + External	0.8647	0.8532	0.8589
MacBERT	Official (training)	0.8817	0.8610	0.8712
	Official + External	0.8759	0.8594	0.8676

Table 2: Results of our NER models on the validation set.

Models		**Precision**	**Recall**	**F1**
Embedding	**Data Usage**			
BERT	Official (training + validation)	0.7701	**0.7299**	**0.7418**
	Official+ External	0.7506	0.6991	0.7055
RoBERTa	Official (training + validation)	0.7629	0.7015	0.7207
	Official+ External	0.7477	0.6883	0.7008
MacBERT	Official (training + validation)	**0.7727**	0.7186	0.7351
	Official+ External	0.7553	0.7037	0.7151

Table 3: Results of our NER models on the test set.

3.2 Settings

For character representations, in addition to BERT[4] (Devlin et al., 2019), we also adopted RoBERTa[5] (Liu et al., 2019) and MacBERT [6] (Cui et al., 2020) to compare the performance of different embeddings. We downloaded these pre-trained models from HuggingFace and continuously trained their language models using official data including training, validation and test datasets. The hyper-parameter values for our embedding training were embedding size 768; batch size 64; epoch 20; and learning rate 4e-5.

We trained the BiLSTM-CRF model based on official data provided by task organizers and their variants with our collected external data to confirm performance differences. The hyper-parameter values for our model implementation were optimized as follows: batch size 256; epoch 40; learning rate 0.004; LSTM hidden size 1024; and LSTM dropout rate 0.1.

The evaluation metrics of this shared task are standard precision, recall, and F1-score, which are the most typically used metrics for NER systems at a character level. For each track, the task participants are allowed to have maximum 6 submissions. The final ranking is determined from the best submission based on macro-averaging F1-score.

3.3 Results

Tables 2 and 3 respectively show the results of our submissions on the validation and test sets. We obtained closely consistent results on both datasets. Comparing the embedding effects with RoBERTa and MacBERT, although these two models are modified to improve the BERT model, we did not obtain NER performance improvements when using them as the embedding representation usage. Surprisingly, including external data to train BiLSTM-CRF does not improve the overall F1 performance. The architecture of the BERT-BiLSTM-CRF model using official data training only obtained the best F1-score of 0.7418 on the test set. Table 4 further shows the detailed results per named entity category. The class LOC obtained the best F1-score, followed by PER. In our observations, these two classes are most commonly categories with relatively clear definitions that may not cause recognition confusion. Both combinations of class GRP with CORP and class PROD with CW are usually difficult to distinguish even with manual annotation if insufficient annotation training is provided.

A total of 21 teams participated in the Chinese track of SemEval-2022 MultiCoNER Task, each submitting at least one entry. Our best submission

[4]https://huggingface.co/hfl/chinese-bert-wwm-ext
[5]https://huggingface.co/hfl/chinese-roberta-wwm-ext-large
[6]https://huggingface.co/hfl/chinese-macbert-large

BERT-BiLSTM-CRF	Precision	Recall	F1
Class			
PER	0.8072	0.7356	0.7698
LOC	0.7704	**0.853**	**0.8096**
GRP	**0.8468**	0.5045	0.6323
CORP	0.7641	0.7712	0.7677
PROD	0.755	0.7594	0.7572
CW	0.6768	0.7558	0.7141

Table 4: Detailed results of our BERT-BiLSTM-CRF model on the test set.

achieved an F1 score of 0.7418, ranking in the seventh position out of 21 teams.

4 Conclusion

This study describes the NCUEE-NLP system in the Chinese track of SemEval-2022 MultiCoNER task, including system design, implementation and evaluation. We used the BERT embedding to represent each character in the original sentences and trained BiLSTM-CRF using datasets provided by the organizers to predict the named entity categories. Our best submission had a marco-averaging F1-score of 0.7418, ranking in the 7th position among a total of 21 participating teams.

Acknowledgements

This study is partially supported by the Ministry of Science and Technology, Taiwan, under the grant MOST 108-2218-E-008-017-MY3.

References

Sandeep Ashwini and Jinho D. Choi. 2014. Targetable named entity recognition in social media. *CoRR, arXiv:1408.0782.*

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

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. *In Findings of the Association for Computational Linguistics: EMNLP 2020. Association for Computational Linguistics*, pages 657–668.

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

Chuanhai Dong, Jiajun Zhan, Chengqing Zong, Masanori Hattri, and Hui Di. 2016. Character-based lstm-crf with radical-level features for chinese named entity recognition. *In Proceedings of International Conference on Computer Processing of Oriental Languages. Springer Link*, pages 239–250.

Shijia E and Yang Xiang. 2017. Chinese named entity recognition with character-word mixed embedding. *In Proceedings of the 2017 ACM Conference on Information and Knowledge Management. Association for Computing Machinery*, pages 2055–2058.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. Association for Computational Linguistics*, pages 55–65.

Alex Graves, Abdel rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. *In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. Institute of Electrical and Electronics Engineers*, pages 6645–6649.

Jingzhou He and Houfeng Wang. 2008. Chinese named entity recognition and word segmentation based on character. *In Proceedings of the 6th SIGHAN Workshop on Chinese Language Processing. Association for Computational Linguistics*, pages 128–132.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. *In Proceedings of the 18th International Conference on Machine Learning. Association for Computing Machinery*, pages 282–289.

Lung-Hao Lee and Yi Lu. 2021. Multiple embeddings enhanced multi-graph neural networks for chinese healthcare named entity recognition. *IEEE Journal of Biomedical and Health Informatics*, 25(7):2801–2810.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: word segmentation and named entity recognition. *In Proceedings* *of the 5th SIGHAN Workshop on Chinese Language Processing.Association for Computational Linguistics*, pages 108–117.

Haibo Li, Masato Hagiwara, Qi Li, and Heng Ji. 2014. Comparison of the impact of word segmentation on name tagging for chinese and japanese. *In Proceedings of the 9th International Conference on Language Resources and Evaluation. European Language Resources Association*, pages 2532–2536.

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

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition. *In Proceedings of the 16th International Workshop on Semantic Evaluation. Association for Computational Linguistics.*

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). *In Proceedings of the 16th International Workshop on Semantic Evaluation. Association for Computational Linguistics.*

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Gorrado, and Jeffery Dean. 2013. Distributed representation of words and phrases and their compositionality. *In Proceedings of the 27th Conference on Neural Information Processing Systems*, pages 3111–3119.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embedding. *In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics*, pages 548–554.

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

Jiahui Qiu, Yangming Zhou, Qi Wang, Tong Ruan, and Ju Gao. 2019. Chinese clinical named entity recognition using residual dilated convolutional neural network with conditional random field. *IEEE Transactions on NanoBioscience*, 18(3):306–315.

Guohua Wu, Guangen Tang, Zhongru Wang, Zhen Zhang, and Zhen Wang. 2019. An attention-based bilstm-crf model for chinese clinic named entity recognition. *IEEE Access*, 7:113942–113949.

Canwen Xu, Feiyang Wang, Jialong Han, and Chenliang Li. 2019. Exploiting multiple embeddings for chinese named entity recognition. *In Proceedings of the 28th ACM International Conference on Information and Knowledge Management. Association for Computing Machinery*, pages 2269–2272.

Liang Xu, Yu Tong, Qianqian Dong, Yixuan Liao, Cong Yu, Yin Tian, Weitang Liu, Lu Li, Caiquan Liu, and Xuanwei Zhang. 2020. Cluener2020: Fine-grained named entity recognition dataset and benchmark for chinese. *arXiv:2001.04351.*

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 1:1554–1564.

CMB AI Lab at SemEval-2022 Task 11: A Two-Stage Approach for Complex Named Entity Recognition via Span Boundary Detection and Span Classification

Keyu PU, Hongyi LIU, Yixiao YANG, Jiangzhou JI, Wenyi LV and **Yaohan HE**
CMB AI Lab
{pukeyu, lhy24, yangyixiao}@cmbchina.com
{jesse, lvwymail, heyh18}@cmbchina.com

Abstract

This paper presents a solution for the SemEval-2022 Task 11 Multilingual Complex Named Entity Recognition. What is challenging in this task is detecting semantically ambiguous and complex entities in short and low-context settings. Our team (CMB AI Lab) propose a two-stage method to recognize the named entities: first, a model based on biaffine layer is built to predict span boundaries, and then a span classification model based on pooling layer is built to predict semantic tags of the spans. The basic pre-trained models we choose are XLM-RoBERTa and mT5. The evaluation result of our approach achieves an F1 score of 84.62 on sub-task 13, which ranks the third on the learder board.

1 Introduction

Named entity recognition (NER)(Tjong Kim Sang and De Meulder, 2003) is a fundamental task in natural language processing, aiming at identifying the spans of texts that refer to entities. NER is widely applied to information extraction and data mining(Lin et al., 2019)(Cao et al., 2019), which is greatly challenging in practical and open domain settings. However, the previous research has not paid much attention on processing complex and ambiguous named entities.

SemEval 2022 task 11 (Malmasi et al., 2022b) containing a total of 13 sub-tasks is a complex NER task which focuses on detecting semantically ambiguous and complex entities in short and low-context settings (Meng et al., 2021). For the purpose of testing the domain adaption capability of the participating models, the task not only set 11 base sub-tasks: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi and Bangla, but also set two additional testing sets on questions and short queries: Multilingual, and code-mixed (Fetahu et al., 2021). We conduct a two-stage method to deal with the code-mixed sub-task, which achieves an F1 score of 84.62.

This paper is structured as follows. The related work of NER is briefly introduced in Section 2. The data for training and testing the model is presented in Section 3. The details of the two-stage method is described in Section 4. The experimental results of our method are exhibited in Section 5. Section 6 summarizes this paper.

2 Related Work

In the NLP field, the NER task is usually considered as a sequence labeling problem (Liu et al., 2018) (Lin et al., 2019) (Cao et al., 2019). With well-designed features, CRF-based models have achieved the leading performance (Lafferty et al., 2001) (Finkel et al., 2005) (Liu et al., 2011). Recently, neural network models have been exploited for feature representations (Chen and Manning, 2014). Moreover, contextualized word representations such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) have also achieved great success. As for NER, the end-to-end bidirectional LSTM + CRF model (Lample et al., 2016) (Yang et al., 2018) is one representative architecture. These models are only capable of recognizing regular named entities.

In e-commerce search domain, a common scenario is code-mixed queries, with query terms composed of multiple languages(Bhargava et al., 2016). The application for code-mixed Web queries still remains challenging(Gupta et al., 2014). A recent work proposed an NER hybrid approach for code-mixed queries, consisting of a gazetteer and tree based identifier(Bhargava et al., 2016). Another work leverages linguistic features to train a conditional random field (CRF) model, where the output is further processed using multi-lingual gazetteer lists(Gupta et al., 2016). In Seme Val 2022 task 11 code-mixed sub-task, we use a two-stage NER approach and get the 3rd place in the competition.

3 Data

In this task, we use the official raw data (Malmasi et al., 2022a) to train and test our model. Each line of texts in the data belongs to a sample, the languages involved are: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla, some of which are also provided with code-mixed data as an additional subtask. Entity types include Person, Location, Group, Corporation, Product, and Creative Work. The participants have to use their systems to accurately detect the entities and submit the predictions for the mixed languages task.

In a data file, samples are separated by blank lines. Each data instance is tokenized and each line contains a single token in the first column with the associated label in the last (4th) column. The second and third columns are underscores (_) to separate the tokens and the labels. The entities are labeled with the BIO scheme, which means that the token tagged O is not a part of the entity, the token tagged B-X is the first token of an X type entity, and the remaining tokens of the entity are tagged as I-X.

When the amount of training data is insufficient or unevenly distributed, data augmentation can quickly expand the corpus to avoid overfitting. At the same time, data augmentation can also improve the robustness of the model, preventing the performance of the model from being greatly reduced once the data only changes slightly. We build a dictionary with all entities of the same type to randomly replace the entities in each sample, and translate the replaced entities into other languages to expand the dataset, which is similar is to autoencoders in the computer vision. However, translation between different languages relies on a large number of parallel corpuses, and requires training first.

4 Methodology

The two-stage method we use in this task includes two separated models to recognize the named entities: one for predicting the boundaries of the spans, and the other for predicting the semantic tags of the spans. The processing flow of our approach is depicted in Figure 1.

4.1 Text Encoders

The models we employed are both trained based on XLM-RoBERTa$_{LARGE}$ and mT5$_{LARGE}$.

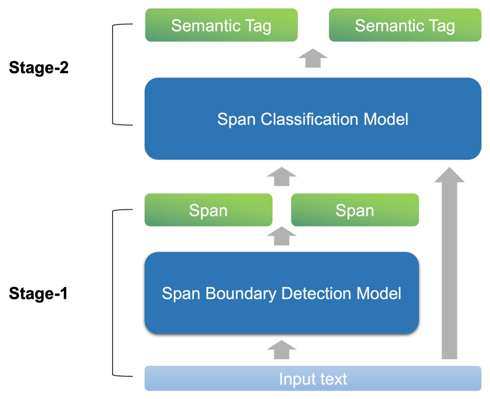


Figure 1: The processing flow of our approach

XLM-RoBERTa (Conneau et al., 2020): XLM-RoBERTa$_{LARGE}$ is pre-trained on 2.5TB of filtered common crawl data containing 100 languages, which consists of 24 transformer layers, 16 self-attention heads per layer, and a hidden size of 1024. In order to deal with a large number of common words in natural language corpus, BPE (byte pair encoding), a coding schema mixed by the character level and the word level representation, is utilized to process the text data.

mT5 (Xue et al., 2021): mT5$_{LARGE}$ is a multilingual pre-trained text-to-text transformer which is pretrained on the common crawl-based dataset corpus, covering 101 languages. We only use the encoder of the mT5$_{LARGE}$ consisting of 24 transformer layers.

4.2 Boundary Detection of Spans

The first stage of our method is to extract the phrase. As shown in Figure 2, we built a boundary detection model by connecting the last hidden states of the pre-trained model to the biaffine layer (Yu et al., 2020) to obtain the span boundaries, which can also be regarded as a named entity recognition (NER) model that only recognize one single category.

The output of the biaffine model is a span boundary matrix as illustrated in Figure 3. All pairs of start-end tokens have corresponding scores indicating whether they are the spans we need. Figure 4 shows an example of a span boundary matrix: the reason why *Jackie Ma* and *the louvre museum* are the entities in this sentence is that they are the two pairs of start-end tokens.

To better detect the span's boundary, we build a dictionary from the training data. Words that appear in training data more than twice are selected and then splice together with the original sentence

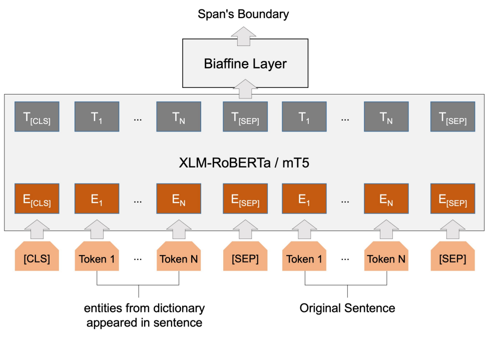


Figure 2: Span boundary detection model

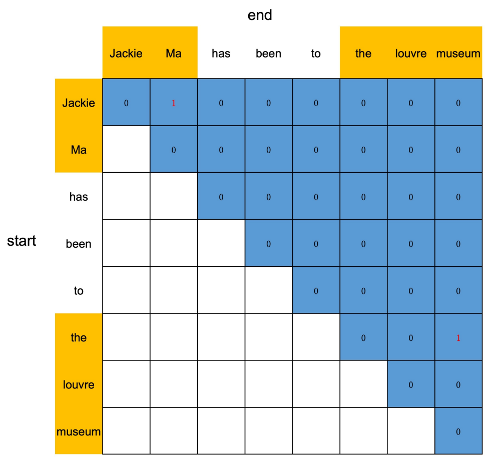
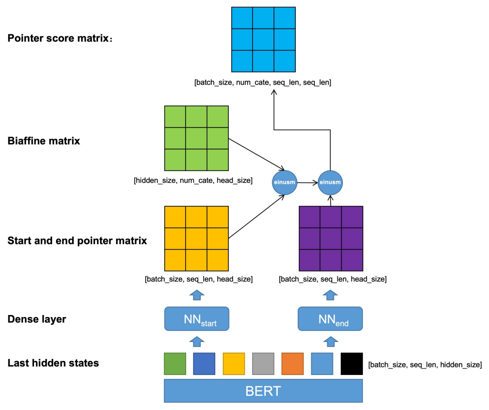


Figure 4: Span boundary matrix: scores of all start-end token-paire for detecting boundary of span



Pointer score matrix:
[batch_size, num_cate, seq_len, seq_len]
Biaffine matrix
[hidden_size, num_cate, head_size]
einsum
einsum
Start and end pointer matrix
[batch_size, seq_len, head_size]
[batch_size, seq_len, head_size]
Dense layer
NN_start
NN_end
Last hidden states
[batch_size, seq_len, hidden_size]
BERT

Figure 3: Biaffine model: using start-end pointer to explore all spans

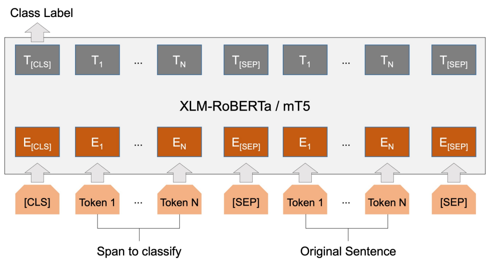


Figure 5: Span classification model

into the input text fragment, as shown below:

[CLS] *word selected 1* [SEP] *word selected 2* [SEP] ... [SEP] *sentence* [SEP].

4.3 Span Classification

The second stage of our method is to classify the spans obtained in the first stage. We built a classification model to determine which semantic tag the span belongs to. As shown in Figure 5, a full connection layer is connected to the pooling layer of the pretrained model to output the score for each category, and then is activated by a softmax function, with the cross entropy set as the loss function of the classification model.

It should be noted that the text fragment to be classified is constructed by the phrase extracted from the first model (boundary detection) and the original sentence. Similar to the sentence pair classification, a sample sentence before BPE applied appears as below:

[CLS] *phrase extracted* [SEP] *sentence* [SEP].

4.4 Training Procedures

In the first stage (span boundary detection), the Adam optimizer with a learning rate of 6×10^{-6} is employed, the batch size is set to 12, and the model is trained for 30 epochs. As for the second stage (span classification), the learning rate of the Adam optimizer is set to 8×10^{-6}, and the batch size and the number of epoch amounts are 16 and 40 respectively. For each stage, we use both the 10-fold cross-validation.

During the training phase of the span classification model, FGM adversarial learning is applied to improve the robustness of model: the samples are mixed with some fairly small disturbances that might lead to misclassification, and the neural network is then adapted to the disturbances to be robust to the adversarial samples.

4.5 Ensemble Model

The ensemble of deep learning models has a great improvement on the test dataset. We ensemble the predictions of span boundary detection models by voting strategy to get the best span boundary. Besides, the predictions of span classification models are also combined to get the final predictions. Each model is trained on different dataset augmented and based on different text encoders (i.e., XLM-RoBERTa and mT5). What is most conspicuous, however, is that the strategy of the two-stage model performs better than the traditional ner model in this task.

5 Results

A two-stage method is employed to complete the code-mixed language sub-task. Based on two pre-trained models (XLM-RoBERTa and mT5), we adopted a variety of optimization schemes, such as: biaffine network structure, two-stage entity prediction, adding distantly supervised dictionary and adversarial training, all of which have achieved a certain improvement, according to the evaluation results shown in Table 1. Lastly, we voted on all prediction results in terms of ensemble learning idea to get the final submission file.

With the XLM-RoBERTa + crf method as the baseline, and an end-to-end structure, we get an F1 score of 79.7. After using the biaffine network structure and two-stage optimization architecture instead, the F1 score improves to 80.9 and 81.3 respectively. In addition, the two-stage optimization architecture introducing supervised dictionary, adversarial training and data augmentation obtains F1 scores of 82.5, 83.1 and 82.7 respectively. Compared to the XLM-RoBERTa, the prediction results acquired based on the mT5 pre-trained model are improved by an average of 0.4 points. As a result, the final evaluation scores gained by voting is 84.62.

6 Conclusion

Aiming at the complex multilingual ambiguity and lack of context in this competition, we adopt a deep learning network model for entity extraction based on the biaffine attention mechanism, and carry out transfer learning based on different pre-trained models such as RoBERTa and mT5.

Through adversarial training, the robustness of the model is enhanced, and the two-stage training also improves the performance of the model in few-shot scenarios. Besides, a remote supervised dictionary is added to revise the results, and the entity dictionary for random replacement and multilingual machine translation is used for data augmentation. Usually for enhanced data, it is necessary to give a weight less than 1, which is different from real data. Data augmentation can also alleviate the problem of data imbalance. Ultimately, the best result (F1 score of 84.62) is achieved via ensemble learning voting strategy.

Comparison of different methods	
Method	F1(%)
Baseline XLM-RoBERTa+crf	79.7
Biaffine	80.9
Two-Stage	81.3
w/Dict	82.5
w/adv train	83.1
w/data augmentation	82.7
Baseline mT5+crf	80.2
Biaffine	81.2
Two-Stage	81.7
w/Dict	82.8
w/adv train	83.6
w/data augmentation	83.2
Ensemble strategy	84.62

Table 1: The code-mixed sub-task evaluation results

References

Rupal Bhargava, Bapiraju Vamsi Tadikonda, and Yashvardhan Sharma. 2016. Named Entity Recognition for Code Mixing in Indian Languages using Hybrid Approach. In *FIRE*.

Yixin Cao, Zikun Hu, Tat-seng Chua, Zhiyuan Liu, and Heng Ji. 2019. Low-Resource Name Tagging Learned with Weakly Labeled Data. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 261–270.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing(EMNLP)*, pages 740–750.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370.

Deepak Gupta, Shubham Tripathi, Asif Ekbal, and Pushpak Bhattacharyya. 2016. A Hybrid Approach for Entity Extraction in Code-Mixed Social Media Data. *Money*, 25(66).

Parth Gupta, Kalika Bali, Rafael E Banchs, Monojit Choudhury, and Paolo Rosso. 2014. Query Expansion for Mixed-Script Information Retrieval. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 677–686.

John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the eighteenth international conference on machine learning, ICML, volume 1*, pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Ying Lin, Liyuan Liu, Heng Ji, Dong Yu, and Jiawei Han. 2019. Reliability-aware Dynamic Feature Composition for Name Tagging. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 165–174.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Fangzheng XU, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower Sequence Labeling with Task-Aware Neural Language Model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing Named Entities in Tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 359–367.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

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

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

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

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design Challenges and Misconceptions in Neural Sequence Labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named Entity Recognition as Dependency Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476.

UA-KO at SemEval-2022 Task 11: Data Augmentation and Ensembles for Korean Named Entity Recognition

Hyunju Song, Steven Bethard
University of Arizona
{hyunjusong,bethard}@email.arizona.edu

Abstract

This paper presents the approaches and systems of the UA-KO team for the Korean portion of SemEval-2022 Task 11 on Multilingual Complex Named Entity Recognition. We fine-tuned Korean and multilingual BERT and RoBERTA models, conducted experiments on data augmentation, ensembles, and task-adaptive pretraining. Our final system ranked 8th out of 17 teams with an F1 score of 0.6749 F1.

1 Introduction

Named Entity Recognition (NER) is the task of recognizing and classifying named entities in unstructured text. NER is a critical component for NLP tasks such as question answering and relation extraction. Recent advances in neural NER have allowed state-of-the-art systems to perform well in recognizing persons, locations, and organizations in a variety of benchmark datasets. However, these systems still struggle to recognize complex named entities such as titles of creative works. SemEval Task 11 (Malmasi et al., 2022b) asks participants to build systems to identify both classic and complex named entities in multiple languages.

In this paper, we describe the UA-KO team's approach to deal with the challenge of recognizing complex named entities in Korean. We used both monolingual and multilingual transformer-based models. To improve the performance of the models, we conducted experiments on data augmentation, ensembles, and task-adaptive pretraining. Our final performance on the test set ranked 8th out of 17 teams with an F1 score of 0.675. Code to replicate our experiments is available at https://github.com/hyunssong/semeval2022-task11.

2 Related Work

NER in real-world settings is challenging. Identifying entities in short texts, such as search queries on the web, is not easy due to the lack of context. Identifying nontraditional named entities such as titles of creative works (movies, books, or TV shows) is challenging because similar phrases appear as non-named entities and the characteristics of creative work titles change over time (Ashwini and Choi, 2014). Approaches to such real-world named entity types include integrating English Wikipedia as a gazetteer within neural NER models (Meng et al., 2021), though simply relying on the robustness of large pre-trained models like XLM-R is more successful (Ushio and Camacho-Collados, 2021).

SemEval-2022 Task 11 asks participants to build systems that can perform well in real-world settings, providing a dataset with complex named entities in short sentences. The task is divided into three tracks: multi-lingual, monolingual, and code-mixed. We participated in the Korean portion of the monolingual track.

Korean is an agglutinative language, where each word (*eojeol*) is formed by combining a root morpheme with a bound morpheme (*josa*) or postposition (*eomi*) (Choi et al., 2017). More than 60 different forms can thus be created from each root morpheme. A model that is not aware of this morphological richness may end up significantly increasing the vocabulary size to represent all these different forms, at the costs of having only sparse data to learn each form and encountering a high rate of out-of-vocabulary (OOV) forms.

Recent studies have explored different representations to account for the agglutinative characteristics of Korean. Lee et al. (2020) explored both the syllable level and sub-character level representations of the text, achieving similar results to multilingual BERT with 1/10 of the training data. Kwon et al. (2017) proposed a deep learning based NER system that operates over syllables rather than words, resulting in a speedup by removing the need for morphological analysis. Kim et al. (2021) explored morpheme, syllable, and subcharacter rep-

Train	Dev	Test
15300	800	150K~500K

Table 1: The Korean language portion of the SemEval-2022 Task 11 data.

resentations and also found that syllables were the most effective representation for Korean NER.

3 Data

The SemEval-2022 Task 11 dataset follows the CoNLL format. Each token is classified into six entity types: person, location, group, corporation, product, or creative work. The size of the Korean language portion of the dataset is shown in table 1. More details about the dataset can be found in Malmasi et al. (2022a). Systems applied to this dataset are evaluated based on the macro-average F1 score.

Our data augmentation experiments (see section 4.2) utilized additional data. For locations, we used the GeoNames geographical database, which provides countries and place names in many different languages. We selected the 170 country names provided in the Korean language. For person names, we used the Encyclopedia of Korean Culture (of Korean Studies, 1989) , which is a Korean language encyclopedia. We selected their 18,506 names of Korean historical figures.

4 Methodology

We used both monolingual and multilingual pretrained models for the task. For the monolingual models, we used KoBERT[1], KR-BERT (Lee et al., 2020), Ko-ELECTRA[2], and KLUE-RoBERTa-large(Park et al., 2021) , which all have shown successful performance on Korean NLP tasks. For the multilingual model, we used XLM-RoBERTa-large (Conneau et al., 2019), which was provided as the baseline of the shared task.

We fine-tuned these models on the training set. Monolingual models were fine-tuned with a learning rate of 5e-5 for 5 epochs with a weight decay of 0.01. The multilingual model was trained with the same configuration but for 3 epochs. We also experimented with different methodologies to improve performance: task-adaptive pretraining, data augmentation, and ensembles. These approaches are described in the following sections.

[1]`https://github.com/SKTBrain/KoBERT`
[2]`https://github.com/monologg/KoELECTRA`

4.1 Task Adaptive Pretraining

Gururangan et al. (2020) showed that further pretraining models on the unlabeled task data, called task-adaptive pretraining, can improve model performance when the dataset is curated to capture language appropriate to the task. We follow the training environment as Gururangan et al. (2020). To avoid catastrophic forgetting, we additionally reduced the number of epochs following Zhao et al. (2021)'s approach of applying task-adaptive pretraining on small datasets. Specifically, we conduct task-adaptive pretraining with a batch size of 256 for 50 epochs. For the KoELECTRA model, we further train for 7k steps. XLM-R was not included in these experiments due to computing resource limitations.

4.2 Data Augmentation

Data augmentation is a strategy that enhances the amount of training data by modifying existing data or generating new synthetic data, usually by leveraging external resources. Dai and Adel (2020) proposed several data augmentation techniques for NER tasks, including generating sentences by replacing tokens or shuffling within sentence segments. Since the provided training dataset is small relative to the test set, we decided to conduct data augmentation to reduce the overfitting of the model on the training dataset. We specifically focused on person and location named entities as these entities had the highest percentage in the test set.

We generated new sentences by utilizing the external database described in section 3. For each of the sentences that contain location or person named entity, we generated k new sentences where $k = 3$ or 6. New sentences were generated by replacing the location named entity with Korean country names from GeoNames database or replacing the person named entity with historic person names from the Encyclopedia of Korean Culture. We then fine-tuned the language models on the original sentences together with these augmented sentences.

4.3 Ensembles

Ensemble methods are an effective way of combining multiple machine-learning models to make better predictions (Rokach, 2010). We created ensembles over the different monolingual and multilingual models using soft voting, which predicts the class label based on the argmax of the sums of

Model	Modification	F1
KoBERT	-	0.808
KoBERT	TAPT	**0.817**
KoBERT	Augment $k = 3$	0.816
KoBERT	Augment $k = 6$	0.806
KR-BERT	-	0.824
KR-BERT	TAPT	0.826
KR-BERT	Augment $k = 3$	**0.834**
KR-BERT	Augment $k = 6$	0.830
KoELECTRA	-	0.827
KoELECTRA	TAPT	0.767
KoELECTRA	Augment $k = 3$	**0.829**
KoELECTRA	Augment $k = 6$	0.827
KLUE	-	**0.850**
KLUE	TAPT	0.832
KLUE	Augment $k = 3$	0.846
KLUE	Augment $k = 6$	0.846
XLM	-	**0.831**
XLM	Augment $k = 3$	**0.831**
XLM	Augment $k = 6$	0.823
Ensemble(KLUE, XLM)	-	0.858
Ensemble(KLUE, XLM)	Augment $k = 3$	0.855
Ensemble(All Korean)	-	0.863
Ensemble(All Korean)	Augment $k = 3$	0.866
Ensemble(All Korean, XLM)	-	0.864
Ensemble(All Korean, XLM)	Augment $k = 3$	**0.868**

Table 2: Performance of different models on the Dev set. "All Korean" stands for all Korean monolingual models: KoBERT, KR-BERT, KoELECTRA, and KLUE. The best scoring system in each group is in bold.

the predicted probabilities of the various classifiers.

5 Results on Dev

Table 2 shows performance of task adaptive pre-training (TAPT), data augmentation (Augment $k = N$), and ensembles (Ensemble(...)) on the development set. Task adaptive pretraining yielded little benefit over the corresponding unadapted model, and sometimes dramatically worsened performance (e.g., the KoELECTRA model went from 0.827 to 0.767). Data augmentation either led to small gains over the non-augmented model or to roughly the same performance, and $k = 3$ was generally as good or better than $k = 6$. Ensembling led to consistent gains compared to the single models. The best overall model on the development set was an ensemble of KoBERT, KR-BERT, KoELECTRA, KLUE, and XLM combined with data augmentation where $k = 3$.

Given these results, for our official submissions on the test set, we applied data augmentation with $k = 3$, and we included several types of ensembles. We did not apply any task adaptive pre-training because it showed no benefits on the development data.

Model	F1
KLUE	0.651
KLUE, Augment $k = 3$	0.650
Ensemble(All Korean)	0.668
Ensemble(All Korean), Augment $k = 3$	0.626
Ensemble(All Korean, XLM)	**0.675**
Ensemble(All Korean, XLM), Augment $k = 3$	0.650

Table 3: Performance of different models on Test set. "All Korean" stands for all Korean monolingual models: KoBERT, KR-BERT, KoELECTRA, and KLUE. The best scoring system is in bold.

Class	Precision	Recall	F1
LOC	0.689	0.788	0.735
PER	0.776	0.748	0.761
PROD	0.706	0.665	0.685
GRP	0.678	0.595	0.634
CW	0.526	0.563	0.544
CORP	0.688	0.693	0.691

Table 4: Performance by class label for the best performing model, the ensemble of KoBERT, KR-BERT, KoELECTRA, KLUE, and XLM. Results for the other models look qualitatively similar.

6 Results on Test

Table 3 shows the performance of our submitted models on the test set. Unlike our results on the development set, data augmentation significantly reduced performance on the test set. For the runs without data augmentation, ensembles outperformed single models as in our development set results. The failure of data augmentation may imply a low overlap between the names we drew from GeoNames and the Encyclopedia of Korean Culture for data augmentation, and the named entities in the test data. That is, the coverage of these gazetteers may have been insufficient for the unique and nontraditional named entities of the test set, similar to problems mentioned in Meng et al. (2021).

Table 4 shows detailed performance of our best-performing model, the ensemble of KoBERT, KR-BERT, KoELECTRA, KLUE, and XLM. This model performs best in recognizing person names (0.761 F1), but has difficulty recognizing creative works (0.544 F1).

dataset	sentence	label
dev	(A1)《**유희열의 스케치북**》/은/한국방송공사/음악/전문/텔레비전/ 프로그램이다.	CW
	(A1)《**Yu Hee-yeol’s Sketchbook**》/is a/KBS/music/television/show.	
dev	(A2) 1993년/ 매크래컨은/ 해나/ 바베라/ 카툰스의/ 애니메이션/ 시리즈/《2 stupid dogs》/의 / 미술/ 감독으로/ 일했다.	CW
	(A2) In 1993/ McCracken/ Hanna/ Babera/ Cartoons/ Animation/ series/《2 stupid dogs》/’s/ art/ director/ worked.	
dev	(A3) 정도가/ 약한/ 경우는/ 완전히/치료하지는/ 않으며/ **일반 화장품**/ 간단히/ 감출/ 수/있다.	PROD
	(A3) Degree/weak/ in the case of/ completely/ cure/ don’t/ **general cosmetics**/ simply/ cover/ is/ possible.	
dev	(A4) 제 18회/ (1987년) / 영화/《 브라질 》/ **테리**/ 길리엄/	PER
	(A4) 18th/ (1987) / Film /《Brazil》/ **Terry**/ Gilliam	
test	(B1) 는/ 은여울역/ 카운티입니다.	-
	(B1) is/ Eunyeoul Station/ County.	-
test	(B2) 에/ 로그인/ 텔레페	-
	(B2) to/ login/ Telefe	-

Table 5: Example sentences from the shared task data. Gold annotations are in bold. Model predictions are underlined.

6.1 Error Analysis

We analyze our best-performing system’s predictions on the dev set to understand our system’s strengths and weaknesses. By investigating the errors where the model is highly confident, we identified the following qualities from the errors on the dev set.

Annotation errors: We observed inconsistent labeling in named entities, especially in the creative work named entities. The development data contained many creative works enclosed in《 and 》, such as example A1 in table 5. However the development data also contained creative works marked by the same punctuation that were not labeled as creative works by annotators, such as example A2. This led to a high error rate in predicting creative work named entities on the development data, so we were unsurprised to see similar low performance on the test data creative works. We speculate that the reason data augmentation was not helpful on the test data was these errors in annotation.

Token boundary issues: We see that often tokens are misclassified due to token boundary issues. For instance, in example A3 in table 5, the system found the product named entity but included an extra token before the start of the product name.

Foreign names: The model had difficulty recognizing transliterated named entities, such as names of foreign people or groups as in example sentence A4 of table 5. These names are different from traditional Korean words, likely leading to the system’s difficulty in identifying them.

Grammatical errors: The sentences of the test set differ grammatically from the training and dev set. Table 5 shows some test set inputs that are incomplete (B1) or have grammatical errors (B2). B1 is missing the subject of the sentence, and B2 does not follow the subject-object-verb order of a Korean sentence structure. We found such grammatical problems to be frequent in the test set. As such grammatical problems were not frequently present in the development data, our models were not robust to them.

7 Conclusion

We have presented a description of our different approaches for identifying complex named entities in Korean language data. A monolingual model that considers characteristics of the Korean language performs well, and an ensemble of monolignual models and a multilingual model further improves performance. Though we also explored task-adaptive pretraining and data augmentation, task-adaptive pretraining did not help on the development data, and data augmentation helped on the development data but hurt on the test data. Our results suggest that while ensembles yield reliable gains for Korean named entity recognition, further research is needed to utilize external knowledge when dealing with complex named entity recognition.

References

Sandeep Ashwini and Jinho D Choi. 2014. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*.

Sanghyuk Choi, Taeuk Kim, Jinseok Seol, and Sang-goo Lee. 2017. A syllable-based technique for word embeddings of Korean words. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 36–40, Copenhagen, Denmark. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Gyeongmin Kim, Junyoung Son, Jinsung Kim, Hyunhee Lee, and Heuiseok Lim. 2021. Enhancing korean named entity recognition with linguistic tokenization strategies. *IEEE Access*, 9:151814–151823.

Sunjae Kwon, Youngjoong Ko, and Jungyun Seo. 2017. A robust named-entity recognition system using syllable bigram embedding with eojeol prefix information. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 2139–2142, New York, NY, USA. Association for Computing Machinery.

Sangah Lee, Hansol Jang, Yunmee Baik, Suzi Park, and Hyopil Shin. 2020. KR-BERT: A small-scale korean-specific language model. *arXiv preprint arXiv:2008.03979*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512, Online. Association for Computational Linguistics.

Academy of Korean Studies. 1989. Encyclopedia of korean culture. *Acad Korean Stud*, 10:704.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, et al. 2021. Klue: Korean language understanding evaluation. *arXiv preprint arXiv:2105.09680*.

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

Asahi Ushio and Jose Camacho-Collados. 2021. T-NER: An all-round python library for transformer-based named entity recognition. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 53–62, Online. Association for Computational Linguistics.

Zhixue Zhao, Ziqi Zhang, and Frank Hopfgartner. 2021. *A Comparative Study of Using Pre-Trained Language Models for Toxic Comment Classification*, page 500–507. Association for Computing Machinery, New York, NY, USA.

USTC-NELSLIP at SemEval-2022 Task 11: Gazetteer-Adapted Integration Network for Multilingual Complex Named Entity Recognition

Beiduo Chen[1], **Jun-Yu Ma**[1], **Jiajun Qi**[1], **Wu Guo**[1], **Zhen-Hua Ling**[1] and **Quan Liu**[2]
[1]National Engineering Laboratory for Speech and Language Information Processing,
University of Science and Technology of China
[2]State Key Laboratory of Cognitive Intelligence, iFLYTEK Research
{beiduo,mjy1999,jiajun97}@mail.ustc.edu.cn
{guowu,zhling}@ustc.edu.cn, quanliu@iflytek.com

Abstract

This paper describes the system developed by the USTC-NELSLIP team for SemEval-2022 Task 11 Multilingual Complex Named Entity Recognition (MultiCoNER). We propose a gazetteer-adapted integration network (GAIN) to improve the performance of language models for recognizing complex named entities. The method first adapts the representations of gazetteer networks to those of language models by minimizing the KL divergence between them. After adaptation, these two networks are then integrated for backend supervised named entity recognition (NER) training. The proposed method is applied to several state-of-the-art Transformer-based NER models with a gazetteer built from Wikidata, and shows great generalization ability across them. The final predictions are derived from an ensemble of these trained models. Experimental results and detailed analysis verify the effectiveness of the proposed method. The official results show that our system ranked **1st** on three tracks (Chinese, Code-mixed and Bangla) and **2nd** on the other ten tracks in this task.

1 Introduction

Named Entity Recognition (NER) is a core natural language processing (NLP) task, which aims at finding entities and recognizing their type in a text sequence. In practical and open-domain settings, it is difficult for machines to process complex and ambiguous named entities (Ashwini and Choi, 2014). For example, "*On the Beach*" is the title of a movie but cannot be recognized easily by present NER systems. This issue may become even more serious in multilingual or code-mixed settings (Fetahu et al., 2021). However, it has not received sufficient attention from the research community. To alleviate the issue, SemEval-2022 Task 11 (Malmasi et al., 2022b) formulates this task which focuses on detecting semantically ambiguous and complex entities in short and low-context settings for 11 languages.

One of the classic approaches to solving this problem is to integrate external entity knowledge or gazetteers into neural architectures (Liu et al., 2019; Rijhwani et al., 2020; Meng et al., 2021). Typically, the two representations respectively from a language model like BERT (Devlin et al., 2019) and a gazetteer network like BiLSTM (Hochreiter and Schmidhuber, 1997) are combined as one merged embedding, which is further fed into a NER classifier such as a conditional random field (CRF) (Lafferty et al., 2001). However, there is a sense of "gap" between the two networks. The gazetteer network has no explicit semantic learning goal itself, which means it is just a more complex but almost isolated embedding layer for gazetteer information and cannot obtain the true meaning of NER tags actively. Almost no semantic information can be gained by the classic gazetteer network.

To effectively connect the two networks, a gazetteer-adapted integration network (GAIN) is proposed. The GAIN adopts a two-stage training strategy to adapt the gazetteer network to the language model. During the first training stage, the parameters of a language model are fixed. Then a sentence and its annotation are fed into the two networks separately. The two outputs are adapted by minimizing the KL divergence between them. This training process helps the gazetteer network truly understand the meaning of NER tags by transferring semantic information from the pre-trained language model to the gazetteer network, as the randomly initialized gazetteer network is gradually adapted to the pre-trained language model during the training. In the second stage, with maintaining the training process above, a gazetteer built from Wikidata is applied to generate pseudo annotations searched by string matching. A sentence and the corresponding pseudo annotation are then fed into the two pre-trained networks separately. Finally, integration methods like the concatenation or weighted summation are utilized

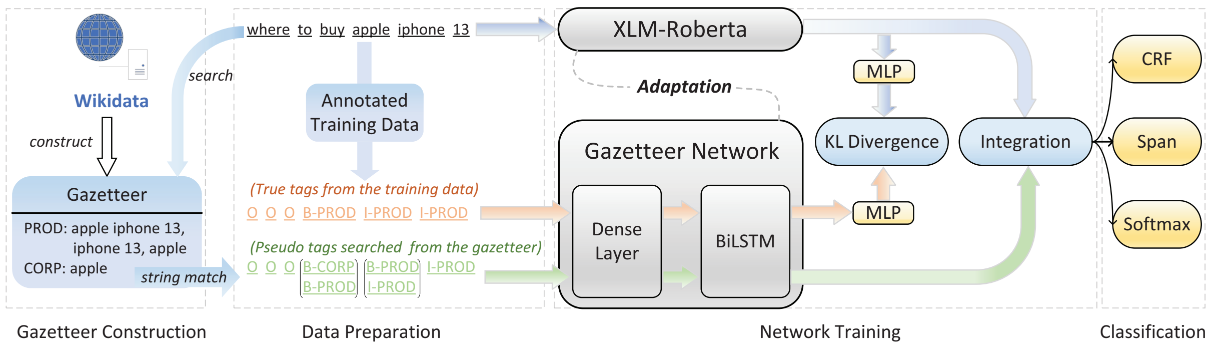


Figure 1: The overall structure of the proposed system.

on the two output representations for classifying.

The proposed method achieves great improvements on the validation set (Malmasi et al., 2022a) of SemEval-2022 Task 11 compared to baseline models and ordinary NER systems with gazetteers. Ensemble models are used for all thirteen tracks in the final test phase, and our system officially ranked **1st** on three tracks (Chinese, Code-mixed and Bangla), and **2nd** on the other ten tracks among nearly 50 teams overall. The outstanding performance demonstrates the effectiveness of our method. Fine-grained results show that our method significantly improves scores of difficult labels like "CREATIVE-WORK" and "PRODUCT", which is the key challenge of this task. To facilitate the reproduction of our results, the code is available at `https://github.com/Mckysse/GAIN`.

2 Task Description

SemEval-2022 Task 11 challenges participants to develop complex NER systems for 11 languages (English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla), focusing on recognizing semantically ambiguous and complex entities in short and low-context settings (Malmasi et al., 2022b). Each language constitutes a single track, while Multilingual and Code-mixed are added as Track 12 and 13. The task adopts the WNUT 2017 (Derczynski et al., 2017) taxonomy entity types: PERSON (PER for short, names of people), LOCATION (LOC, locations/physical facilities), CORPORATION (CORP, corporations and businesses), GROUPS (GRP, all other groups), PRODUCT (PROD, consumer products), and CREATIVE-WORK (CW, movie/song/book/etc. titles). The task also aims at testing the domain adaption capability of the systems by adding additional test sets on questions and short search queries.

For each language, a training set with 15300 samples and a validation set with 800 samples are provided. For the Code-mixed track, 1500 training samples and 500 validation samples are provided. The test data for each track have instances between 150K+ and 500K+ (Malmasi et al., 2022a).

3 System Description

This study focuses on making better use of the external entity knowledge. To describe our system clearly, in this section, we first introduce three basic mainstream NER systems used. Then we show the process of constructing a gazetteer with Wikidata, and how the gazetteer representation is generated and utilized. Finally, we illustrate the gazetteer-adapted integration network (GAIN). The overall structure of the proposed system is shown in Figure 1.

3.1 Basic NER Systems

We mainly use the XLM-RoBERTa large (Conneau et al., 2020) as the pre-trained language model, which is a widely used encoder. Generated by feeding a sentence into the encoder, the representation is then projected to a 13-dimension embedding corresponding to 13 BIO-tags (eg. B-PER, I-PER, O, ...) through a linear transformation. Three mainstream NER backend classifiers are adopted: Softmax (Devlin et al., 2019) and CRF (Huang et al., 2015) are classic sequential labeling methods that predict the tag of each token, and Span (Yu et al., 2020) is a segment-based method that predicts the start and the end of an entity separately.

3.2 The Gazetteer

It's difficult to process complex and ambiguous entities only relying on the language model itself (Ashwini and Choi, 2014). To integrate external entity knowledge, we first need to build a large

		Coverage Rate per Label						
Language	Total Num.	LOC	CW	PER	GRP	PROD	CORP	Average
BN	73,468	46.52%	20.21%	7.93%	19.58%	10.61%	14.62%	19.91%
DE	901,635	84.34%	93.62%	19.05%	87.70%	47.72%	94.73%	71.19%
EN	1,032,955	84.26%	79.30%	18.29%	86.79%	47.25%	94.04%	68.32%
ES	961,535	83.31%	83.08%	21.03%	67.32%	27.51%	77.81%	60.01%
FA	633,348	87.31%	77.54%	90.19%	65.70%	42.47%	70.88%	72.35%
HI	82,669	46.87%	13.99%	15.25%	16.36%	12.06%	13.95%	19.75%
KO	358,510	58.64%	79.94%	84.01%	74.40%	44.27%	63.51%	67.46%
NL	701,839	79.69%	72.54%	23.93%	68.35%	26.33%	72.20%	57.17%
RU	495,585	57.17%	72.19%	9.75%	41.46%	28.29%	57.20%	44.34%
TR	407,901	84.15%	79.88%	88.54%	58.29%	44.26%	76.75%	71.98%
ZH	513,704	73.42%	62.84%	19.37%	81.54%	34.67%	71.95%	57.30%
MIX	4,434,100	91.71%	93.44%	44.30%	90.58%	55.22%	96.55%	78.63%
MULTI	4,434,100	73.19%	69.71%	44.49%	58.65%	31.44%	62.00%	56.58%
Average coverage rate		73.12%	69.10%	37.39%	62.82%	34.78%	66.63%	**57.31%**

Table 1: The metrics of our gazetteer in detail. The Total Num. column means the accurate number of entries in the gazetteer for each track. Numbers with % denote the coverage rates to entities in the training and validation set.

gazetteer matching the taxonomy, then we have to consider how to fuse the gazetteer information with the semantic information from the language model.

3.2.1 Construction

Our gazetteer is built based on Wikidata. Wikidata is a free and open knowledge base. Every entity of Wikidata has a page consisting of a label, several aliases, descriptions, and one or more entity types. The entity type annotated by Wikidata is the key to constructing a gazetteer. For example, "*apple*" can be annotated as a kind of fruit or a well-known high-tech corporation in America. Thus, according to WNUT 2017 taxonomy (Derczynski et al., 2017) used in this competition, the word "*apple*" is given both PROD and CORP labels.

To construct a gazetteer fit to the data of this task, firstly every entity of the training set is searched in Wikidata. Then all the entity types returned are mapped to the NER taxonomy with 6 labels. Next, all Wikidata entities stored in these entity types can be added to the 6 labels gazetteer separately. Of course, there is a lot of entities that cannot be searched, especially in some languages such as BN and HI. Also, the elementary gazetteer has plenty of noise. By measuring the number and the coverage rate of each language on each label, the mapping relationships are adjusted manually. In the end, a multilingual gazetteer is obtained that contains entities from 70K to 1M for each language. The gazetteer approximately has a coverage rate of 57 percent on entities in the training and validation set. Basic information about our gazetteer is shown in Table 1. "Coverage Rate" is calculated as the number of entities both appeared in the official data and in our gazetteer divided by the total number of entities in the official data.

Words	O	B-CORP	I-CORP	B-PROD	I-PROD
where	1	0	0	0	0
to	1	0	0	0	0
buy	1	0	0	0	0
apple	0	1	0	1	0
iphone	0	0	0	1	1
13	0	0	0	0	1

Table 2: Example of the one-hot representation for a searched sentence. The rest 8 labels are all zero.

3.2.2 Application

To apply the gazetteer to a sentence, firstly a search tree is constructed for string matching. Once a sentence is fed into the search tree, a maximum length matching algorithm will be conducted, and a 13-dimension one-hot vector for each token will be generated. Take the sentence "*where to buy apple iphone 13*" for example. By string matching with the gazetteer, "*apple iphone 13*", "*iphone 13*" and "*apple*" are found in the PROD gazetteer, while "*apple*" is also found in the CORP gazetteer. Then a 13-dimension one-hot vector will be generated for every word as shown in Table 2.

Denote one sentence as $\mathbf{w} = (w_1, w_2, ...w_N)$ where N is the length of the sentence and w_i is the i^{th} word. By feeding $\mathbf{w}$ into the encoder such as the XLM-RoBERTa large, a semantic representation $\mathbf{e} \in \mathbb{R}^{N \times D}$ is obtained, where D is the hidden size. At the same time, the one-hot vector generated from the search tree is fed into a gazetteer network consisting of a dense layer and a BiLSTM. To match the hidden size of the language model, the output embedding $\mathbf{g}$ has the same size with $\mathbf{e}$.

Two common ways are used to integrate $\mathbf{e}$ and $\mathbf{g}$. One way is to concatenate them on each token, another way is to get the weighted summation of them by setting a trainable parameter $\lambda \in \mathbb{R}^{N \times D}$. The final representation is fed into the backend classifier for supervised NER training.

3.3 Gazetteer-Adapted Integration Network

Through the analysis in Section 6.2, it is found that only conducting the normal training process above is not enough. Since the encoder XLM-RoBERTa large and the gazetteer network BiLSTM are almost isolating each other during the whole training, almost no semantic information can be gained explicitly by the classic gazetteer network.

To address this problem, the GAIN method is proposed as a two-stage training strategy. In the first stage, the adaptation between the two networks is conducted. Take the sentence $\mathbf{w} = \{$*where to buy apple iphone 13*$\}$ for example. Assuming the correct tags are $\mathrm{T} = \{$O,O,O,B-PROD,I-PROD,I-PROD$\}$, an one-hot vector is constructed just based on the true tags. A gazetteer representation $\mathbf{g}_r \in \mathbb{R}^{N \times D}$ is obtained after passing the vector through the gazetteer network. Then the parameters of the language model are fixed, and the sentence $\mathbf{w}$ is fed into it to get a semantic representation $\mathbf{e}$. $\{\mathbf{g}_r, \mathbf{e}\}$ are projected to $\{\mathbf{g}_r^t, \mathbf{e}^t\} \in \mathbb{R}^{N \times 13}$ by two separate linear layers, where the semantic meaning is transferred to the tags meaning as a kind of logits distributions. The adaptation is implemented by the designed loss L_1:

$$L_1(\mathbf{w}) = \mathrm{KL}(sg(\mathbf{g}_r^t)||\mathbf{e}^t) + \mathrm{KL}(sg(\mathbf{e}^t)||\mathbf{g}_r^t) \quad (1)$$

where $\mathrm{KL}(\cdot)$ is the KL divergence calculation and $sg(\cdot)$ operation is used to stop back-propagating gradients, which is also employed in Jiang et al. (2020); Liu et al. (2020). The loss L_1 is the symmetrical Kullback-Leibler divergence, encouraging the distributions $\mathbf{g}_r^t$ and $\mathbf{e}^t$ to agree with each other. Thus, the gazetteer network will understand the real meaning of the NER tags and gain semantic information transferred from the language model.

In the second stage, all the parameters are trained with a gazetteer. As illustrated in Section 3.2.2, a gazetteer representation $\mathbf{g}$ is generated from the search tree and the gazetteer network BiLSTM. Next, an ordinary fusion method is applied to $\mathbf{g}$ and $\mathbf{e}$ to get an integration representation, which is then fed into the backend classifier to compute a conventional loss with true tags T. This supervised training goal is implemented by the loss L_2:

$$L_2(\mathbf{w}) = \mathrm{Classifier}(f(\mathbf{g}, \mathbf{e}), \mathrm{T}) \quad (2)$$

where $f(\cdot)$ denotes ordinary integration methods like concatenation or weighted summation. $\mathrm{Classifier}(\cdot)$ represents one of the three mainstream backend classifiers mentioned in Section 3.1. During the whole second-stage training, a multitask learning goal is conducted shown as:

$$L_3(\mathbf{w}) = \alpha L_1(\mathbf{w}) + L_2(\mathbf{w}) \quad (3)$$

where α is a hyperparameter that is manually set for different fusion and backend methods.

4 Data Preparation

For the basic training set provided officially, an entity replacement strategy is adopted using our own gazetteer to construct a double data-augmented set. This part of data is called “data-wiki”, which mainly consists of rich-context sentences.

In order to improve the performance of our models on low-context instances, a set of annotated sentences are generated from the MS-MARCO QnA corpus (V2.1) (Nguyen et al., 2016) and the ORCAS dataset (Craswell et al., 2020), which are mentioned in Meng et al. (2021). Our trained models and existing NER systems (e.g., spaCy) are applied to identify entities in these corpora, and only templates identically recognized by all models are reserved. Finally, 3753 English templates for MS-MARCO and 13806 English templates for ORCAS are obtained. After slotting the templates by our own gazetteer and translating them to the other 10 languages, we get approximately 16K annotated low-context sentences for each language. This part of data is called “data-query”.

A special operation is conducted for the Code-mixed track, because only 2000 annotated instances are provided officially. The multilingual function

Parameter	Value
Hidden size for language models	1024 for large, 768 for base
Learning rate for language models	2e-5 for large, 1e-5 for base
Learning rate for gazetteer networks	2e-4 for large, 1e-4 for base
Learning rate for the CRF layer	2e-3 for large, 1e-3 for base
First-stage training epochs	5
Second-stage training epochs	20
Batchsize	32
Dropout rate	0.1
α for the second stage training	5 for Softmax and Span, 100 for CRF
Optimizer	AdamW
Activation function	Relu

Table 3: Hyperparameters for our system. “large” means the 24-layer transformer model, and “base” denotes the 12-layer transformer model.

Model Name	Lang
XLM-R large (Conneau et al., 2020)	Multi
chinese-roberta-wwm (Cui et al., 2021)	ZH
luke-large (Yamada et al., 2020)	EN
klue-roberta (Park et al., 2021)	KO
bert-base-turkish (Schweter, 2020)	TR
bert-fa-base (Farahani et al., 2021)	FA

Table 4: Pre-trained language models used.

in Wikidata is utilized, as every entity in the Wiki page has several expressions in other languages. For every sentence in “data-wiki” and “data-query”, the entities inside are randomly replaced with their translations recorded by Wikidata. In this way, a set of annotated code-mixed data are built.

Track	Team Num.	F1	Rank
English (EN)	30	0.8547	2
Spanish (ES)	18	0.8544	2
Dutch (NL)	15	0.8767	2
Russian (RU)	14	0.8382	2
Turkish (TR)	15	0.8552	2
Korean (KO)	17	0.8636	2
Farsi (FA)	15	0.8705	2
German (DE)	16	0.8905	2
Chinese (ZH)	21	0.8169	1
Hindi (HI)	17	0.8464	2
Bangla (BN)	18	0.8424	1
Multilingual	26	0.853	2
Code-mixed	26	0.929	1

Table 5: Official rankings of our system. “F1” denotes the macro-F1 on the test set.

5 Experiments

5.1 Encoder Selection

Preliminary experiments show that it’s important to start with advanced pre-trained language models for further improvements. More than a dozen models are evaluated on the development set, and the language models listed in Table 4 are adopted eventually. For some tracks, the XLM-R large and the monolingual model are both used for ensemble. For the other tracks without monolingual models in the corresponding language, just the XLM-R large is used. All the resources can be found on the HuggingFace Page (Wolf et al., 2019).

5.2 Training Details

A lot of models have been trained with the GAIN method using different classifiers, different integration methods, different chosen language models. Hyperparameter settings are shown in Table 3. Sample codes are already available at `https://github.com/Mckysse/GAIN` to help the reproduction.

A 5-fold cross-validation training strategy is also applied in the evaluation. The prepared data “data-wiki” and “data-query” are split into five pieces, each one is used as the validation set, while the other four pieces are used as the training set. After obtaining the five best models by this strategy, the logits of them (for Softmax and Span models) are averaged to integrate them as an aggregated model. CRF models are just voted averagely.

Finally, the predictions of our best models in different methods are token-voted by setting a

Strategy	Classifier	BN	DE	EN	ES	FA	HI	KO	NL	RU	TR	ZH	MIX
A	CRF	0.771	0.886	0.846	0.834	0.78	0.771	0.813	0.878	0.802	0.835	0.866	0.654
	Softmax	0.763	0.879	0.849	0.836	0.783	0.767	0.811	0.871	0.792	0.835	0.862	0.652
	Span	0.793	0.896	0.853	0.845	0.806	0.802	0.831	0.879	0.809	0.839	0.884	0.696
B	CRF	0.816	0.906	0.865	0.857	0.821	0.8	0.853	0.888	0.817	0.865	0.908	0.788
	Softmax	0.799	0.901	0.865	0.859	0.824	0.796	0.851	0.879	0.815	0.864	0.901	0.786
	Span	0.811	0.917	0.871	0.857	0.818	0.825	0.864	0.887	0.82	0.858	0.906	0.792
C	CRF	0.841	0.943	0.891	0.87	0.835	0.831	0.871	0.902	0.829	0.884	0.913	0.833
	Softmax	0.829	0.931	0.888	0.872	0.839	0.822	0.868	0.897	0.831	0.882	0.909	0.835
	Span	0.832	0.935	0.892	0.874	0.836	0.837	0.879	0.901	0.836	0.872	0.912	0.823
weighted token-vote		**0.864**	**0.955**	**0.922**	**0.892**	**0.855**	**0.853**	**0.899**	**0.916**	**0.843**	**0.903**	**0.922**	**0.865**

Table 6: All macro-F1 scores on the validation set. Only scores of the concatenation integration method are listed due to limited spaces. Strategy A denotes baseline systems mentioned in Section 3.1, B denotes ordinary integration method with the gazetteer mentioned in Section 3.2, and C denotes the GAIN method mentioned in Section 3.3. "weighted token-vote" represents the ensemble of all our models including those with the weighted summation integration method not listed, and achieves the best performance on the validation set.

Coverage Rate	BN	DE	EN	ES	FA	HI	KO	NL	RU	TR	ZH	MIX	avg
0	0.784	0.897	0.856	0.847	0.8	0.775	0.839	0.892	0.806	0.855	0.863	0.662	0.823
30%	0.791	0.898	0.861	0.845	0.804	0.799	0.84	0.893	0.814	0.856	0.872	0.694	0.831
50%	0.858	0.901	0.867	0.844	0.807	0.871	0.866	0.897	0.814	0.861	0.903	0.709	0.85
70%	0.891	0.907	0.868	0.854	0.811	0.899	0.894	0.901	0.82	0.869	0.904	0.732	0.863
100%	0.974	0.973	0.942	0.92	0.903	0.978	0.938	0.94	0.91	0.91	0.964	0.914	0.934

Table 7: F1 scores of the gradient coverage rate trial. "Coverage Rate" means the number of entities in official data also found in our gazetteer / the number of entities in official data. "avg" denotes the average result of all tracks.

weight for each track. The weight is manually set referring to all scores on the validation set.

5.3 Official Results

Our team participate in all the 13 tracks and the results in the test phase are listed in Table 5. We ranked **1st** on three tracks and **2nd** on ten tracks.

6 Analysis

6.1 Effectiveness of The Gazetteer

To explore the effectiveness of the proposed GAIN methods, a large number of trials are conducted on the official data mentioned in Section 2. All scores under the concatenation integration setting on the validation set are listed in Table 6. Significant improvements are gained by the gazetteer and the GAIN method on all tracks. The results demonstrate that the gazetteer plays a pivotal role in processing complex entities.

6.2 Coverage Rate Trial

Our gazetteer can only reach approximately 57% coverage rate over the entities in the official data. Intuitively, the higher the coverage rate over the entities reaches, the better the performance can achieve. We carry out a toy experiment to explore this conjecture. Since entities from the official training and development set can be extracted, we can control how many of these entities appear in our gazetteer to modulate the coverage rate, as shown in Table 7.

Not as expected, scores on many tracks don't improve in step with the increase of the coverage rate when it does not reach 100%. This situation mostly happens in languages that already have good scores, like DE and EN. To explore the reason why the gazetteers under 100% coverage rate don't work, we check the weight λ mentioned in Section 3.2.2. It's surprising to find the λ is nearly zero when the coverage rate is not 100%, which indicates that our models almost don't use the gazetteer information. Further fine-grained tests find that only when the coverage rate exceeds the basic accuracy of the model, the λ starts to have a non-zero value. An empirical conclusion can be drawn that the gazetteer network and the language model almost process information separately, and the final integration module simply selects the better one for classifying. Thus, this study starts to figure out an explicit way to adapt the two networks, and the GAIN method is designed.

6.3 Detailed Results on the Test Set

As shown in Table 8 and Table 9, we display all the detailed results on all tracks during the test phase for further analyses in-depth.

Domain	Metrics\Lang	bn	de	en	es	fa	hi	ko	nl	ru	tr	zh
	macro@F1	0.8424	0.8905	0.8547	0.8544	0.8705	0.8464	0.8636	0.8767	0.8382	0.8552	0.8169
	macro@P	0.8584	0.8988	0.8641	0.8664	0.8816	0.8600	0.8739	0.8856	0.8485	0.8662	0.8394
	macro@R	0.8343	0.8835	0.8467	0.8439	0.8610	0.8392	0.8556	0.8686	0.8289	0.8470	0.8007
	ALLTRUE	135634	271979	272922	265748	192194	144940	217281	265942	249458	149369	173183
	ALLPRED	133563	269699	269658	261020	189525	143238	215304	262626	245126	148025	169983
	ALLRECALLED	123479	253694	244826	238000	172403	132392	197725	242799	219455	135212	152918
overall	MD@F1	0.9174	0.9367	0.9025	0.9036	0.9033	0.9188	0.9142	0.9187	0.8874	0.9093	0.8912
	F1@LOC	0.8339	0.8900	0.8452	0.8606	0.8853	0.8719	0.8789	0.8793	0.8258	0.8427	0.8575
	F1@PER	0.8724	0.9333	0.9315	0.9224	0.9234	0.9160	0.9100	0.9268	0.8634	0.9096	0.8021
	F1@PROD	0.7539	0.8803	0.8221	0.7885	0.8324	0.7694	0.8225	0.8421	0.8166	0.8348	0.8121
	F1@GRP	0.8956	0.8777	0.8622	0.8585	0.8670	0.8642	0.8683	0.8891	0.8329	0.8479	0.7966
	F1@CORP	0.8921	0.8961	0.8761	0.8799	0.8827	0.8674	0.8767	0.8949	0.8671	0.8722	0.8422
	F1@CW	0.8067	0.8655	0.7909	0.8165	0.8325	0.7894	0.8253	0.8278	0.8237	0.8240	0.7911
	macro@F1	0.8656	0.9541	0.9209	0.8910	0.8529	0.8518	0.9037	0.9143	0.8420	0.9005	0.9101
	macro@P	0.8732	0.9580	0.9283	0.9015	0.8613	0.8649	0.9064	0.9214	0.8522	0.9032	0.9276
	macro@R	0.8582	0.9503	0.9138	0.8808	0.8449	0.8393	0.9011	0.9074	0.8322	0.8980	0.8942
	ALLTRUE	15699	151645	152604	147408	72862	25836	97937	146090	129475	29940	52767
	ALLPRED	15436	150512	150411	144103	71764	25073	97614	144131	126514	29840	51610
	ALLRECALLED	14426	147460	142921	133408	62804	22985	89837	136200	109732	27358	49264
lowner	MD@F1	0.9267	0.9760	0.9433	0.9153	0.8685	0.9030	0.9188	0.9386	0.8573	0.9153	0.9440
	F1@LOC	0.9018	0.9657	0.9441	0.9034	0.8701	0.8961	0.9131	0.9457	0.8130	0.9038	0.9544
	F1@PER	0.9413	0.9762	0.9717	0.9572	0.8935	0.9273	0.9199	0.9621	0.8387	0.9238	0.9457
	F1@PROD	0.8112	0.9308	0.8395	0.7974	0.7677	0.7884	0.8812	0.8449	0.8202	0.8375	0.8849
	F1@GRP	0.8647	0.9519	0.9332	0.8989	0.8728	0.8691	0.9000	0.9237	0.8443	0.9287	0.8445
	F1@CORP	0.8616	0.9607	0.9409	0.9311	0.8691	0.8460	0.9230	0.9219	0.8924	0.9260	0.9247
	F1@CW	0.8132	0.9396	0.8963	0.8580	0.8439	0.7842	0.8851	0.8876	0.8436	0.8835	0.9063
	macro@F1	0.8237	0.7933	0.7559	0.7968	0.8720	0.8316	0.8182	0.8164	0.8223	0.8317	0.7564
	macro@P	0.8420	0.8050	0.7678	0.8099	0.8820	0.8466	0.8323	0.8263	0.8316	0.8422	0.7796
	macro@R	0.8178	0.7880	0.7505	0.7917	0.8657	0.8287	0.8115	0.8109	0.8171	0.8270	0.7473
	ALLTRUE	101225	101171	101197	101220	101149	101116	101027	101160	101153	101050	101179
	ALLPRED	99529	100039	100222	99970	99741	100211	99465	100097	100006	100135	99159
	ALLRECALLED	91027	87729	84475	88455	92126	91762	90263	89032	91767	90378	85483
orcas	MD@F1	0.9069	0.8720	0.8388	0.8793	0.9172	0.9116	0.9004	0.8848	0.9124	0.8985	0.8534
	F1@LOC	0.7682	0.7374	0.6745	0.7621	0.8613	0.8138	0.7985	0.7365	0.7909	0.7742	0.7410
	F1@PER	0.8445	0.8486	0.8511	0.8606	0.9342	0.8990	0.8881	0.8684	0.8776	0.8963	0.7327
	F1@PROD	0.7393	0.8217	0.8021	0.7773	0.8673	0.7608	0.7756	0.8395	0.8152	0.8343	0.7637
	F1@GRP	0.9027	0.7711	0.7633	0.8089	0.8628	0.8633	0.8418	0.8453	0.8195	0.8262	0.7923
	F1@CORP	0.8951	0.8167	0.7989	0.8202	0.8896	0.8707	0.8393	0.8653	0.8389	0.8596	0.8000
	F1@CW	0.7923	0.7641	0.6456	0.7519	0.8169	0.7820	0.7661	0.7437	0.7914	0.7997	0.7084
	macro@F1	0.7512	0.8683	0.8174	0.8510	0.9175	0.9094	0.8819	0.8695	0.8591	0.8783	0.8537
	macro@P	0.7721	0.8706	0.8010	0.8449	0.9245	0.9189	0.8845	0.8630	0.8537	0.8764	0.8594
	macro@R	0.7344	0.8665	0.8366	0.8595	0.9111	0.9014	0.8797	0.8787	0.8671	0.8810	0.8484
	ALLTRUE	18710	19163	19121	17120	18183	17988	18317	18692	18830	18379	19237
	ALLPRED	18598	19148	19025	16947	18020	17954	18225	18398	18606	18050	19214
	ALLRECALLED	18026	18505	17430	16137	17473	17645	17625	17567	17956	17476	18171
msq	MD@F1	0.9663	0.9660	0.9139	0.9474	0.9653	0.9819	0.9646	0.9473	0.9593	0.9595	0.9452
	F1@LOC	0.9372	0.9334	0.8548	0.9113	0.9553	0.9685	0.9421	0.9055	0.9209	0.9245	0.9296
	F1@PER	0.9276	0.9335	0.9366	0.9252	0.9771	0.9688	0.9525	0.9368	0.9399	0.9428	0.8762
	F1@PROD	0.7890	0.8211	0.7734	0.7810	0.8848	0.8435	0.7858	0.8129	0.7255	0.8267	0.7886
	F1@GRP	0.0000	0.7797	0.7273	0.7527	0.8771	0.8518	0.8357	0.8079	0.8063	0.7994	0.7868
	F1@CORP	0.9365	0.8603	0.8214	0.8574	0.9110	0.9231	0.8811	0.8937	0.8708	0.8797	0.8665
	F1@CW	0.9171	0.8817	0.7909	0.8784	0.8999	0.9007	0.8939	0.8605	0.8909	0.8967	0.8745

Table 8: All detailed results of the official test set on monolingual tracks.

Track\Metrics	macro@F1	macro@P	macro@R	MD@F1	F1@LOC	F1@PER	F1@PROD	F1@GRP	F1@CORP	F1@CW
Code-mixed	0.9290	0.9321	0.9261	0.9662	0.9314	0.9461	0.9164	0.9247	0.9463	0.9093
Multilingual	0.8530	0.8605	0.8489	0.9210	0.8681	0.9076	0.8105	0.8152	0.8786	0.8381

Table 9: Results of the official test set on the Code-mixed and Multilingual tracks.

As shown in the tables, the proposed GAIN method significantly improves the performance of the language model on recognizing entities of hard labels like "CW" and "PROD". Besides, results in the domain of "orcas" and "msq" indicate the effectiveness of the proposed data-augment strategy mentioned in Section 4.

6.4 Average-Logits Experiments

This section explains why we choose to average logits of softmax-based models (for Softmax and Span models) for integrating them as an aggregated model, rather than averagely token-vote. Also, a 5-fold cross-validation training is conducted with the official training data on the basic Softmax method. Without loss of generality, BN, EN, FA, and ZH are chosen to represent different language families. Results on the official validation set are shown in Table 10. It is empirically demonstrated that average-logits for the softmax-based model ensemble is better than average-token-vote in most situations.

Data\Lang	bn	en	fa	zh
1 fold	0.795	0.871	0.814	0.874
2 fold	0.799	0.873	0.8	0.871
3 fold	0.801	0.876	0.795	0.889
4 fold	0.798	0.881	0.791	0.881
5 fold	0.789	0.869	0.8	0.873
avg	0.796	0.874	0.8	0.878
avg-token-vote	0.813	0.887	0.815	0.898
avg-logits	**0.815**	**0.89**	**0.817**	**0.903**

Table 10: Results of the 5-fold cross-validation trial. "avg" denotes the average results of 5 models' scores. "avg-token-vote" represents the averagely token-vote process. "avg-logits" means average logits of 5 models are fed into the backend softmax layer for classifying, which achieves the best performance on all 4 languages.

6.5 About KL

This section illustrates why the GAIN method adopts the KL divergence rather than the MMSE loss. Softmax-based classification usually chooses the position of the maximum in all dimensions to be the predicted tag. Because the softmax module and logits-softmax module exist in the KL calculation, the 13-dimension vector can be considered as a kind of logits distribution with the tags meaning. Thus, the distribution, or the internal relationship between all logits of tokens in a sentence, is the most important thing to pay attention to, rather than the amplitude of all logits. With the KL divergence, two distributions are adapted to each other without considering the amplitude.

A toy trial has been conducted to confirm whether the two 13-dimension vectors from the two networks have the same amplitude. The result shows that they are not always in the same ratio relations for different inputs. Compared to the MMSE loss which also calculates relating to the absolute amplitude difference, the KL divergence is better since it only focuses on the distributions of the two outputs, without being disturbed by the inconsistent relationship of the amplitude.

7 Conclusion

This paper presents the implementation of the USTC-NELSLIP system submitted to the SemEval-2022 Task 11 MultiCoNER. The GAIN method is proposed to adapt the gazetteer network to the language model, and achieves great improvements on the complex NER task. Some construction methods for gazetteers and code-mixed data augmentation are also provided. In future works, we will keep exploring effective ways to integrate gazetteer networks with encoders to make better use of the external entity knowledge.

References

Sandeep Ashwini and Jinho D. Choi. 2014. Targetable named entity recognition in social media. *CoRR*, abs/1408.0782.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8440–8451. Association for Computational Linguistics.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. ORCAS: 20 million clicked query-document pairs for analyzing search. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2983–2989. ACM.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese BERT. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3504–3514.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the

WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text, NUT@EMNLP 2017, Copenhagen, Denmark, September 7, 2017*, pages 140–147. Association for Computational Linguistics.

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

Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, and Mohammad Manthouri. 2021. Parsbert: Transformer-based model for persian language understanding. *Neural Process. Lett.*, 53(6):3831–3847.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer enhanced named entity recognition for code-mixed web queries. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1677–1681. ACM.

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

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2177–2190. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28-August 2, 2019, Volume 1: Long Papers*, pages 5301–5307. Association for Computational Linguistics.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020. Adversarial training for large neural language models. *CoRR*, abs/2004.08994.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1499–1512. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won-Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Tae Hwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Eunjeong Lucy Park, Alice Oh, Jung-Woo Ha, and Kyunghyun Cho. 2021. KLUE: korean language understanding evaluation. *CoRR*, abs/2105.09680.

Shruti Rijhwani, Shuyan Zhou, Graham Neubig, and Jaime G. Carbonell. 2020. Soft gazetteers for low-resource named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8118–8123. Association for Computational Linguistics.

Stefan Schweter. 2020. Berturk - bert models for turkish.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6442–6454. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6470–6476. Association for Computational Linguistics.

Multilinguals at SemEval-2022 Task 11: Transformer Based Architecture for Complex NER

Amit Pandey*, Swayatta Daw*, and **Vikram Pudi**
International Institute of Information Technology, Hyderabad, India
{amit.pandey, swayatta.daw}@research.iiit.ac.in
vikram@iiit.ac.in

Abstract

We investigate the task of complex NER for the English language. The task is non-trivial due to the semantic ambiguity of the textual structure and the rarity of occurrence of such entities in the prevalent literature. Using pre-trained language models such as BERT, we obtain a competitive performance on this task. We qualitatively analyze the performance of multiple architectures for this task. All our models are able to outperform the baseline by a significant margin. Our best performing model beats the baseline F1-score by over 9%.

1 Introduction

The Named Entity Recognition (NER) task aims to detect entities from unstructured text and classify them into predefined categories. Although the task of NER has been investigated adequately by previous research work (Mansouri et al., 2008; Nadeau and Sekine, 2007; Lample et al., 2016a; Florian et al., 2003; Ritter et al., 2011), the detection of named entities in open-domain settings is non-trivial. Moreover, the introduction of additional layers of complexity, in the form of semantic ambiguity and a lower amount of contextual availability, poses further challenges. For example, in a low-context and semantically ambiguous sentence such as *Let us play Among Us*, the token sequence *Among Us*, can refer to a common phrase or a popular video game, and hence be categorized as a Creative Work (CW).

Recently, deep learning models have gained popularity for NER (Yadav and Bethard, 2019; Li et al., 2020; Habibi et al., 2017). However, these approaches are data-intensive and become ineffective when there is a lack of labeled data. To foster research in this area, (Malmasi et al., 2022b) has introduced the SemEval MultiCoNER shared task that deals with multilingual complex named entity recognition. This task in based on the complex NER, search query and code-mixing NER challenges introduced by Meng et al. (2021) and Fetahu et al. (2021). The baseline introduced by the organizers for this MultiCoNER challenge is a pre-trained XLM-RoBERTa model (Conneau et al., 2019) that was further fine-tuned on the task-specific training dataset.

This paper describes our approach to tackle complex NER task for the English language using state-of-the-art deep learning models and introduces a simple neural network architecture that builds on top of pre-trained language models. We compare multiple architectures on the validation and test set of the shared task. All our models outperform the baseline by a significant margin. Through our experiments, we discover that leveraging transformer models based on attention mechanism (Vaswani et al., 2017) results in better performance even in low context and ambiguous settings. The code is available at `https://github.com/AmitPandey-Research/Complex_NER`

We describe the prior research work done with respect to both general and low-resource NER tasks in Section 2. We provide the formal task description in Section 3, the dataset details in Section 4, the method and the model architecture in Section 5. We provide details about the experimental implementation in Section 6. We discuss the results obtained and error analysis in Sections 7 and 8 respectively, and finally, we conclude the paper in Section 9.

2 Related Work

A widely used benchmark for NER was the CoNLL 2003 shared task. It contained annotated newswire text from the Reuters RCV1 corpus. Previous researchers (Baevski et al., 2019) had used BiLSTM models with attention to predict named entities on this dataset. (Ma and Hovy, 2016) used a BiLSTM-CNN-CRF to predict the named entities.

*Equal Contribution

Sequence labeling for Named Entity Recognition: Recent approaches have aimed at utilizing deep learning techniques for training NER models. However, these techniques require a large amount of token-level labeled data for NER tasks. Annotation for such kinds of labeled datasets can be expensive, time-consuming, and laborious. The datasets introduced in this task encompass a large number of low-resource and complex NER entities.

Recent work on NER in scientific documents has been concentrated around detecting biomedical named entities (Kocaman and Talby, 2020) or scientific entities like tasks, methods and datasets (Luan et al., 2018; Jain et al., 2020; Mesbah et al., 2018).

NER has been traditonally modelled as a sequence labelling task, using CRF (Lafferty et al., 2001) to classify the labels. Recent approaches have used deep learning based models (Li et al., 2018). These approaches are data intensive in nature. To tackle the label scarcity problem, methods like Distant Supervision (Wang et al., 2020; Liang et al., 2020; Hedderich et al., 2021), Active Learning (Goldberg et al., 2017), Reinforcement Learning-based Distant Supervision (Nooralahzadeh et al., 2019; Yang et al., 2018) have been proposed.

3 Task Description

The objective of this shared task is to build complex Named Entity Recognition systems. The task presents a unique challenge in the form of detecting the entities in semantically ambiguous and low-context settings. Moreover, the shared task also tests the generalization capability and domain adaptability of the proposed systems by testing the system over additional (low-context) datasets containing questions and short search queries, such as Google Search queries.

Label	Description
PER	Person
LOC	Location
GRP	Group
CORP	Corporation
PROD	Product
CW	Creative Work

Table 1: Entity types in the label space

For this task, given an input sentence (an arbitrary sequence of tokens), the systems have to identify the B-I-O format (Ramshaw and Marcus, 1999) (short for beginning, inside, outside) tags for 6 NER entity classes: Person, Product, Location, Group, Corporation, and Creative Work. The description attributed to each class label is described in Table 1.

4 Dataset

The MultiCoNER dataset (Malmasi et al., 2022a) consists of labeled complex Named Entities (NE). For the monolingual track, the participants have to train a model that works for a single language. For training and validation purposes, train and dev sets are provided with labeled entities. The monolingual model trained needs to be used for the prediction of named entities in the test set that consists of more than 150K instances. The labels from the test set are not provided directly. In this system description for the monolingual track, we have considered the English NER dataset for our task. The dataset follows a BIO tagging scheme, and there are six entity types in the label space. The statistics for the English dataset in the monolingual track for the train and dev set are provided in Table 2.

	Train	Dev
# sentences	15300	800

Table 2: Total sentences in English monolingual track

5 System Overview

This section describes our approach to designing a system to solve the problem of classifying the tokens (words) of a given sentence into one of the six NE categories. We also briefly describe features of the BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) model employed in our system.

We designed three architectures based on pre-trained language model BERT: 1) BERT+Linear, 2) BERT+CRF, and 3) BERT+BiLSTM+CRF. A detailed explanation of these architectures is as follows:

5.1 BERT+Linear

We model this task as a multiclass classification problem. The first step to finding labels for the entities is to find dense vector representations of the tokens in the given sentence.

Instead of using static pre-trained word embeddings, such as Word2Vec (Mikolov et al., 2013)

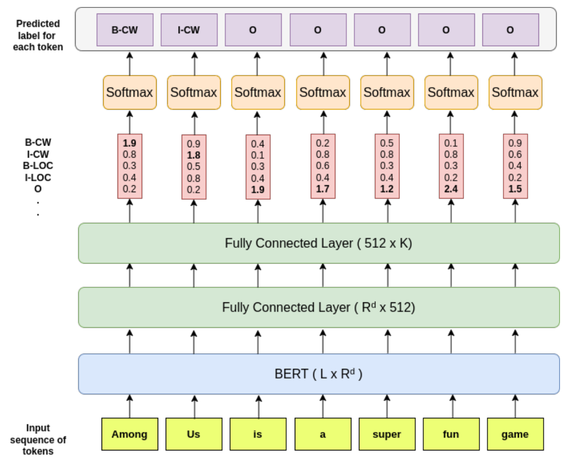


Figure 1: BERT-Linear architecture

and GloVe (Pennington et al., 2014) that rely only on static global representations of word vectors, we employ BERT-based context-aware representations (BERT embeddings) that leverage the full context of the entire sentence.

This helps in extracting more information for the task of NER that is highly dependent on the inter-token relationship. BERT learns the representations for the tokens in the given text by jointly considering both the left and right context of the tokens at each layer (Devlin et al., 2019). To better learn the inter-token dependencies, BERT leverages the attention mechanism with multiple attention heads that focus on different aspects of a token's relation to other tokens. For an ith token x_i among a sequence of tokens $x = (x_1, x_2, x_3, ..., x_m)$, we obtain a low-dimensional BERT embedding, $\tilde{x_i} \in R^d$ where d is the embedding dimension.

We pass this BERT token embedding to a dense classifier that consists of two fully connected layers. This classifier layer maps the BERT embeddings to lower dimension logit vectors $\tilde{x_i} \in R^k$, where k is the total number of labels. The logits are then passed to the softmax normalization function. The softmax generates a probability distribution across all labels for each token, which is then used to predict the most probable label. The system architecture details are shown in Figure 1.

5.2 BERT+CRF

We use a pre-trained BERT model to obtain the token embeddings. These embeddings are passed to a token-level classifier followed by a Linear-Chain CRF. The CRF learns the transfer rules between adjacent entity labels and returns likelihood for a sequence of labels. More formally: 1) For a sequence of tokens $x = (x_1, x_2, x_3, ..., x_m)$, where x_i is the ith token among the sequence of tokens, we obtain a low-dimensional dense embedding, $\tilde{x_i} \in R^d$ where d is the embedding dimension. 2) This embedding is mapped to a lower dimensional space $\tilde{x_i} \in R^k$ where k is the total number of labels. 3) The output emission scores from the linear layer are obtained as $P \in R^{m \times k}$, where m is the number of tokens. These scores are passed to the CRF layer, whose parameters are $A \in R^{k+2 \times k+2}$. Each element A_{ij} signifies the transition score from the ith label to the jth label. The 2 additional states in A are the start and the end state of a sequence. For a series of tokens $x = (x_1, x_2, x_3, ..., x_m)$, we obtain a series of predictions $y = (y_1, y_2, y_3, ..., y_m)$. As described in (Lample et al., 2016b), the score of the entire sequence is defined as :

$$s(x, y) = \sum_{i=0}^{m} A_{y_i, y_{i+1}} + \sum_{i=1}^{m} P_{i, y_i}$$

The model is trained to maximize the log probability of the correct label sequence:

$$\log(p(y|x)) = s(x, y) - \log(\sum_{\tilde{y} \in Y_X} e^{s(x, \tilde{y})})$$

where Y_X are all possible label sequences.

5.3 BERT+BiLSTM+CRF

We use a pre-trained BERT model to obtain the contextual token embeddings for the input sentence. These BERT embeddings are passed to the BiLSTM layer, where the BiLSTM layer captures the information into a hidden state representation. This representation is passed to a CRF layer that obtains the probability distributions across the sequences of labels. Specifically, the fine-tuned BERT language model is used to map the tokens in each sentence to a distributed representation. This is used as the word embedding layer for the BiLSTM+CRF model. The BiLSTM+CRF layer is used to sequence label the sentence, and the predicted labels are obtained. The supervised learning algorithm iterates to improve its predicted label accuracy over every iteration. More formally, the process can be described as follows : 1) The target sentence comprising of m tokens, is represented as $x = (x_1, x_2, x_3, ..., x_m)$, where x_i represents the ith token of the entire target sentence. 2) x_i is mapped to a low dimensional dense vector, $\tilde{x_i} \in R^d$ using the pretrained BERT embeddings, where d is the dimension of dense embedding. 3)

	BERT+Linear			**BERT+CRF**			**BERT+BiLSTM+CRF**		
Class Label	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
LOC	0.9304	0.9145	**0.9224**	0.903	0.9145	0.9087	0.9025	0.9103	0.9064
PER	0.9659	0.9759	**0.9708**	0.936	0.9586	0.9472	0.8882	0.9586	0.9221
PROD	0.7365	0.8367	0.7834	0.7785	0.7891	**0.7838**	0.7372	0.7823	0.7591
GRP	0.8923	0.9158	**0.9039**	0.8341	0.9000	0.8658	0.8466	0.8421	0.8443
CW	0.7955	0.7955	**0.7955**	0.7963	0.733	0.7633	0.7353	0.7102	0.7225
CORP	0.893	0.8653	**0.8789**	0.8877	0.8601	0.8737	0.8837	0.7876	0.8329
Average	0.8689	0.8839	**0.8758**	0.8559	0.8592	0.8571	0.8322	0.8318	0.8312

Table 3: Results of our models on validation dataset

	Prec	**Rec**	**F1**
Baseline System	0.773	0.780	0.776
BERT + CRF	0.855	0.859	0.857
BERT+BiLSTM+CRF	0.832	0.831	0.831
BERT + Linear	**0.868**	**0.883**	**0.875**

Table 4: Comparison of model performances with baseline on validation dataset

	BERT+Linear		
Class Label	Prec	Rec	F1
LOC	0.7292	0.7614	0.7449
PER	0.8776	0.8922	0.8848
PROD	0.7079	0.6460	0.6755
GRP	0.7699	0.6600	0.7107
CW	0.5527	0.6299	0.5888
CORP	0.7253	0.6759	0.6998
Average	0.7271	0.7109	0.7174

Table 5: Performance of model on test dataset

The sequence of vectors is taken as an input to the BiLSTM in each time step, and the forward hidden states $\overrightarrow{h_f} = (\overrightarrow{h_1}, \overrightarrow{h_2}, \overrightarrow{h_3}, ..., \overrightarrow{h_m})$ and the backward hidden states $\overleftarrow{h_b} = (\overleftarrow{h_1}, \overleftarrow{h_2}, \overleftarrow{h_3}, ..., \overleftarrow{h_m})$ are concatenated to form the combined hidden state representation $h = [\overrightarrow{h_f}, \overleftarrow{h_b}]$. 4) The combined hidden state representation $h \in R^{m \times n}$, where n is the total size of BiLSTM, is reduced to a k dimensions using a linear layer, where k is the number of labels to distribute the probabilities across. 4) Finally, the CRF layer is used to obtain the probability of label sequence.

6 Implementation Details

We implement all our transformer-based models using Pytorch and Huggingface library. We implement 3 models: 1) BERT+Linear, 2) BERT+CRF, and 3) BERT+BiLSTM+CRF. We also experiment with feature engineering by concatenating label encoded Part-of-Speech (POS) tags to the token embeddings. We use a dropout from 0.2 to 0.5 in all models and find that a dropout probability of 0.3 gives the best results throughout.

In the BERT+Linear model, we use two fully connected dense linear layers as a classifier on top of the BERT embedding layer. We add a softmax layer to obtain the probability distribution across all the labels. For the BERT+Linear model, we run our experiments across 1-20 epochs. We find that the model starts to overfit after 10 epochs, and the best results are obtained after 5 epochs of training. We further experiment with BERT-base (12 attention heads) and BERT-large (16 attention heads).

For BERT+CRF and BERT+BiLSTM+CRF, we experiment across 1-100 epochs. We find that the models give the most optimal result at the 20th epoch, after which they start to overfit. We use a learning rate of $1e^{-6}$ for all the models. We validate the results of all models using our dev set and then use the best performing model for final evaluation on the blind test set.

7 Results

We compare the performance of our models in the validation set against the baseline. We use the best performing model for the final submission in the evaluation phase. We provide details of the performance of the best performing model over the blind test dataset provided in the evaluation phase. We provide a detailed comparison of the performance of our models across all the class labels in the validation dataset in Table 3. Table 3 shows that the simple BERT+Linear model (0.8758 F1 score) consistently performs better across all the labels (except for PROD) as compared to other larger models. We attribute this to the limited number of samples in the training dataset. The lack of a sufficient number of training samples limits the ability of larger models to generalize properly over the entire training set.

Also, it can be observed from Table 4 that all the 3 models outperform the baseline by a significant margin. BERT+CRF, BERT+BiLSTM+CRF, BERT+Linear advances the baseline by around 8%, 6%, and 9% respectively. Table 5 shows the performance of our BERT+Linear model on the blind test set. Our best performing model ranks 9th on the validation dataset and 15th on the final blind test set. Moreover, through our experiments, we find that the BERT-large offers a significant boost in performance over BERT-base, due to the larger number of attention heads.

8 Error Analysis

We perform error analysis for all 3 different model performances on the validation dataset. We find that for all 3 models, each model has the greatest difficulty in accurately predicting the *CW (Creative Work)* label. This can be attributed to the higher degree of ambiguity when it comes to *CW* named entities, as these often share a similar type of textual structure as regular non-named entity text tokens. It can be inferred that all 3 models are memorizing entity names from the training data to some extent. It is most prevalent in BERT+BiLSTM+CRF model, as we can see that it has the least amount of prediction accuracy among other models. This is consistent with our reasoning that heavier models tend to overfit the dataset faster. Hence, we deduce that named entity memorization can be attributed to a type of overfitting behavior by the model in question when the training data is scarce. The BERT+Linear model, which is the lightest model with the least amount of trainable parameters among all 3, is found to be significantly less prone to memorize entity names.

Furthermore, upon qualitative analysis, we find that our models often have difficulty in recognizing longer named entities (entities comprising of 5 or more tokens). This can be attributed to the lack of occurrence of such entities in the training dataset. The models are majorly exposed to shorter length entity spans across the training set. Due to the lack of exposure of the models to adequate training instances of longer spans, the models are often unable to predict such longer entity spans.

It is also worth noting that an increase in the number of attention heads in the BERT layer helps in substantial improvement in the accuracy. As discussed, this can be attributed to better learning of the context with the help of attention mechanism. We conclude that the larger number of attention heads are able to classify longer entity spans with greater accuracy.

9 Conclusion and Future Work

We experiment with 3 model architectures for a novel dataset introduced for the shared task of detection of complex NER. Our best performing model comprises of a simple linear classifier on top of fine-tuned BERT-based language model. We find that this simple approach performs competitively as compared to its heavier counterparts. Upon analysis, we attribute this observation to the scarcity of labeled training data. BERT+Linear model is able to optimally avoid overfitting to a larger extent and hence performs better than other heavier models. We find that our simpler model ranks in the top 10 in the validation phase and outperforms numerous teams in the final evaluation phase. For future work, we aim to utilize other data augmentation techniques and distant supervision to create clean silver labels in order to increase our training instances. We believe that this would help us leverage larger models for training purposes.

References

Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. Cloze-driven pretraining of self-attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5360–5369, Hong Kong, China. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR*

Conference on Research and Development in Information Retrieval, pages 1677–1681.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 168–171.

Sean Goldberg, Daisy Zhe Wang, and Christan Grant. 2017. A probabilistically integrated system for crowd-assisted text labeling and extraction. *J. Data and Information Quality*, 8(2).

Maryam Habibi, Leon Weber, Mariana Neves, David Luis Wiegandt, and Ulf Leser. 2017. Deep learning with word embeddings improves biomedical named entity recognition. *Bioinformatics*, 33(14):i37–i48.

Michael A. Hedderich, Lukas Lange, and Dietrich Klakow. 2021. ANEA: distant supervision for low-resource named entity recognition. *CoRR*, abs/2102.13129.

Sarthak Jain, Madeleine van Zuylen, Hannaneh Hajishirzi, and Iz Beltagy. 2020. Scirex: A challenge dataset for document-level information extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Veysel Kocaman and David Talby. 2020. Biomedical named entity recognition at scale. *CoRR*, abs/2011.06315.

J. Lafferty, A. McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016a. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016b. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

J. Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2018. A survey on deep learning for named entity recognition. *ArXiv*, abs/1812.09449.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. BOND: bert-assisted open-domain named entity recognition with distant supervision. *CoRR*, abs/2006.15509.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Alireza Mansouri, Lilly Suriani Affendey, and Ali Mamat. 2008. Named entity recognition approaches. *International Journal of Computer Science and Network Security*, 8(2):339–344.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Sepideh Mesbah, Christoph Lofi, Manuel Valle Torre, Alessandro Bozzon, and Geert-Jan Houben. 2018. Tse-ner: An iterative approach for long-tail entity extraction in scientific publications. In *International Semantic Web Conference*, pages 127–143. Springer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Farhad Nooralahzadeh, Jan Tore Lønning, and Lilja Øvrelid. 2019. Reinforcement-based denoising of distantly supervised NER with partial annotation. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 225–233, Hong Kong, China. Association for Computational Linguistics.

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

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1524–1534.

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

Xuan Wang, Yingjun Guan, Yu Zhang, Qi Li, and Jiawei Han. 2020. Pattern-enhanced named entity recognition with distant supervision. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 818–827.

Vikas Yadav and Steven Bethard. 2019. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2159–2169, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

L3i at SemEval-2022 Task 11: Straightforward Additional Context for Multilingual Named Entity Recognition

Emanuela Boros
University of La Rochelle, L3i
La Rochelle, France
emanuela.boros@univ-lr.fr

Carlos-Emiliano González-Gallardo
University of La Rochelle, L3i
La Rochelle, France
carlos.gonzalez_gallardo@univ-lr.fr

Jose G. Moreno
University of Toulouse, IRIT
Toulouse, France
jose.moreno@irit.fr

Antoine Doucet
University of La Rochelle, L3i
La Rochelle, France
antoine.doucet@univ-lr.fr

Abstract

This paper summarizes the participation of the L3i laboratory of the University of La Rochelle in the SemEval-2022 Task 11, *Multilingual Complex Named Entity Recognition* (MultiCoNER). The task focuses on detecting semantically ambiguous and complex entities in short and low-context monolingual and multilingual settings. We argue that using a language-specific and a multilingual language model could improve the performance of multilingual and mixed NER. Also, we consider that using additional contexts from the training set could improve the performance of a NER on short texts. Thus, we propose a straightforward technique for generating additional contexts with and without the presence of entities. Our findings suggest that, in our internal experimental setup, this approach is promising. However, we ranked above average for the high-resource languages and lower than average for low-resource and multilingual models.

1 Introduction

Named entity recognition (NER) is the task of detecting entities and recognizing their type (e.g., person, location, organization) (Grishman and Sundheim, 1996). Standard benchmarks (e.g., CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003)) for NER are either focused on a high-resource language or on a single domain (e.g., news). However, NER is required in different fields, bringing thus several new challenges regarding: the size of the textual content (e.g., short textual snippets like Web queries (Fetahu et al., 2021) or social media posts (Rajoria, 2021)), the characteristics of the entities to be extracted (i.e., capitalization/punctuation features (Mayhew et al., 2019)), unseen and emerging entities (e.g., entities that have a fast growth rate such as new songs and movies are released weekly (Bernier-Colborne and Langlais, 2020)), complex entities (i.e., complex noun phrases such as particularly long nested person names and dates in historical documents (Boros et al., 2020a,c)). Moreover, multilingual documents in which entities of different languages than the rest of the text are present (i.e., code-mixed documents (Winata et al., 2021; Fetahu et al., 2021)) add another level of complexity to the task. Transformer-based (Vaswani et al., 2017) architectures for NER became popular since the release of the BERT model and they hold the state of the art in NER (Akbik et al., 2018; Nie et al., 2020; Yamada et al., 2020; Wang et al., 2020, 2021). However, while most NER systems have been developed to generally address contemporary news data in high resource languages (e.g. English) (Yamada et al., 2020; Wang et al., 2020, 2021), many challenges remain in NER from short texts with complex entities in low-resource scenarios (Meng et al., 2021).

The SemEval-2022 Task 11, *Multilingual Complex Named Entity Recognition* (MultiCoNER) (Malmasi et al., 2022b) aims at developing complex NER systems for 11 languages. The task focuses on detecting semantically ambiguous and complex entities in short, lowercased, low-context monolingual, and multilingual settings. The languages were: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla. For some languages, an additional track with code-mixed data was offered. The organizers aim at testing the domain and language adaption capability of a NER system. For this reason, NER systems for monolingual settings are also constrained to avoid multilingual prediction models and vice versa.

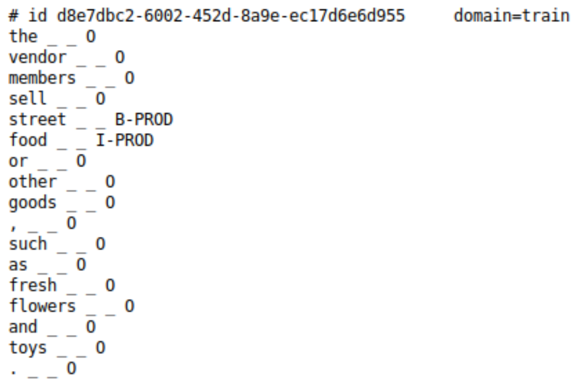
Multilingual and code-mixed documents have motivated the development of systems based on multilingual Transformer-based architectures (Winata et al., 2021; Fetahu et al., 2021). Also, recently, Wang et al. (2021) proposed an approach to NER from short texts by finding external contexts of a sentence. The authors retrieved and selected a set of semantically relevant texts through a search engine, with the original sentence as the query, which proved to be efficient in detecting entities.

In this paper, we propose a robust approach for NER and we tackle the following challenges for the language-specific sub-task, including the multilingual and mixed sub-tasks: (1) short texts: by adding multilingual contexts with and without entities, in order to improve and focus more on the context and less on the other surrounding entities, (2) lowercased texts: we prioritize the usage of uncased language models, and (3) code-mixed and low-resource languages: we add a multilingual language model along with a language-specific language model.

2 Data

For each language, organizers provide fixed-sized training and development sets of $15,300$ and 800 sentences respectively, whereas test sets are composed of an average of $181,418$ ($\sigma = 35,181$) sentences. A multilingual dataset consisting of the concatenation of all monolingual sets is also provided (with a selection of $471,911$ sentences for the test set)[1].

In addition, a smaller code-mixed dataset with entities in a foreign language of the rest of the sentence was given. In this case, $1,500$ sentences constitute the training set, 500 the development set, and $100,000$ the test set. We present a sample annotated example from the English training set in Figure 1.



```
# id d8e7dbc2-6002-452d-8a9e-ec17d6e6d955     domain=train
the _ _ O
vendor _ _ O
members _ _ O
sell _ _ O
street _ _ B-PROD
food _ _ I-PROD
or _ _ O
other _ _ O
goods _ _ O
, _ _ O
such _ _ O
as _ _ O
fresh _ _ O
flowers _ _ O
and _ _ O
toys _ _ O
. _ _ O
```

Figure 1: English annotated example from the training set.

[1]MultiCoNER Dataset can be accessed at https://registry.opendata.aws/multiconer.

To highlight the important difference between the number of phrases in the train and the test sets, we present these numbers in Table 1. The corpus is based on LOWER, MSQ-NER (Bajaj et al., 2016), and ORCAS-NER (Craswell et al., 2020) statements and queries from Wikipedia and Microsoft Bing (Meng et al., 2021). A detailed description and exhaustive statistics of all datasets are presented in Malmasi et al. (2022a,b).

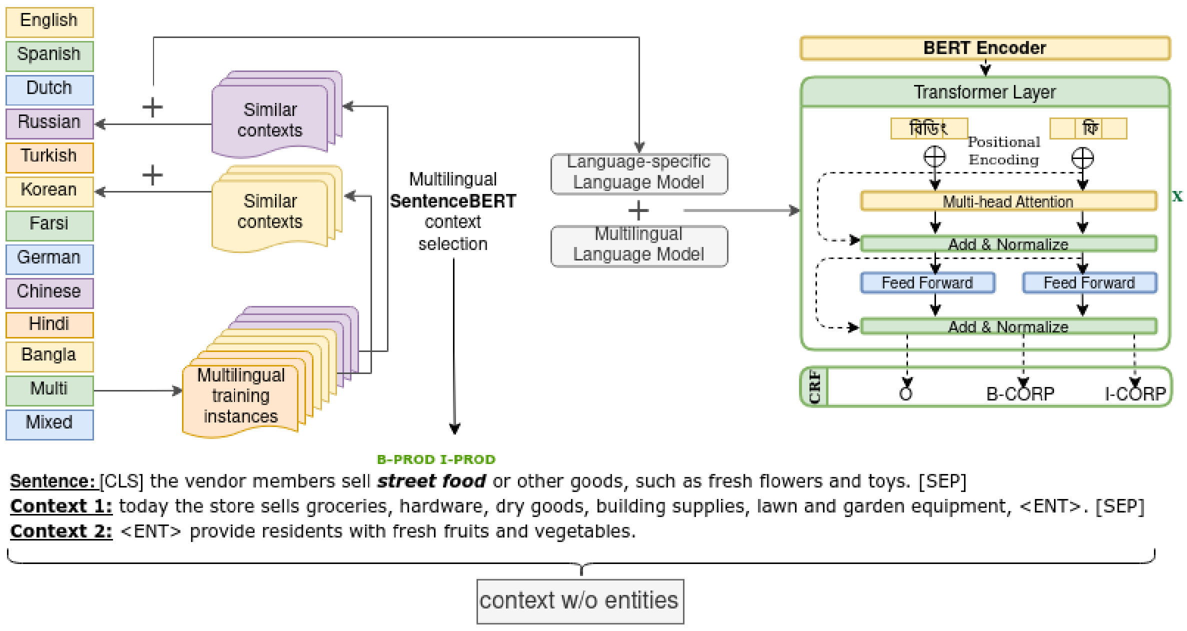
Language	Train	Dev	Test
English			217,818
Spanish			217,887
Dutch			217,337
Russian			217,501
Turkish			136,935
Korean	15,300	800	178,249
Farsi			165,702
German			217,824
Chinese			151,661
Hindi			141,565
Bangla			133,119
Multilingual	168,300	8,800	471,911
Mixed	1,500	500	100,000

Table 1: Number of sentences in the dataset splits per language.

3 Methodology

Our proposed framework consists in augmenting each input sentence with similar contexts taken from the multilingual train collection and a NER model based on a BERT-based pre-trained and fine-tuned language model as an encoder with several Transformer (Vaswani et al., 2017) layers stacked on top. For each language, we consider a language-specific BERT model, as presented in Table 2. We prioritize the use of uncased models, with some exceptions where there were no such models (Dutch, Polish). Finally, we tackle the multilingualism and lack of resources for all languages by concatenating these representations with those provided by a multilingual language model. The methodology is outlined in Figure 2.

Named Entity Recognition Model Our base model was recently proposed for coarse-grained and fine-grained named entity recognition (Boros et al., 2020a,b). The method consists of a hierarchical approach, with a pre-trained and fine-tuned BERT-based encoder (Devlin et al., 2019a). This model consists of a stack of Transformer blocks



Sentence: [CLS] the vendor members sell ***street food*** or other goods, such as fresh flowers and toys. [SEP]
Context 1: today the store sells groceries, hardware, dry goods, building supplies, lawn and garden equipment, <ENT>. [SEP]
Context 2: <ENT> provide residents with fresh fruits and vegetables.

Figure 2: MultiCoNER methodology.

Language	**Model**
English	`bigbird-roberta-large`
Spanish	`bert-base-spanish-wwm-uncased`
Dutch	`bert-base-dutch-cased`
Russian	`rubert-base-cased`
Turkish	`bert-base-turkish-uncased`
Korean	`bert-kor-base`
Farsi	`bert-fa-base-uncased-persiannews`
German	`bert-base-german-dbmdz-uncased`
Chinese	`bert-base-chinese`
Hindi	`bert-base-multilingual-uncased`
Bangla	`bert-base-multilingual-uncased`
Multilingual	`bert-base-multilingual-uncased`
Mixed	`bert-base-multilingual-uncased`

Table 2: Language-specific models. All models can be found at `https://huggingface.co`.

on top of the BERT encoder, and a conditional random field (CRF) layer to decode the best tag path in all possible tag paths. First, the encoder is already based on a stack of Transformer layers. A Transformer block (encoder) is a deep learning architecture based on multi-head attention mechanisms with sinusoidal position embeddings. It is composed of a stack of identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. A residual connection is around each of the two sub-layers, followed by layer normalization. These language models generally expect the input data in a specific format: a special token, [SEP], to mark the end of a sentence or the separation between two sentences, and [CLS], at the beginning of the text. A Transformer encodes the information contained in a given text in the [CLS] token that flows afterward through the layers. This architecture with additional Transformer layers proved to be efficient when the input is noisy and of varying lengths.

Adding Context First, we encode the collection of training multilingual sentences using a pre-trained multilingual Sentence BERT model (Reimers and Gurevych, 2019, 2020a). We consider that using only sentences from the training set is enough for enriching the contexts, even if they are seen during the training process. The underlying idea behind this is that, since the data is multi-domain, we would expect a very small overlap between the train and the test set, thus the additional contexts could benefit from information from other domains that those existing. Then, for each input sentence (for each dataset split), we rank them using the cosine similarity. From these retrieved texts, we select the first $n \in [1, 10]$ sentences and we consider them as semantically relevant.

Next, as shown in the example in Figure 2 with "street food" as a product (PROD) entity, we concatenate the initial input sentence with the retrieved texts, separated by the special token [SEP]. For this step, we propose two versions of taking advan-

Approach	P	R	F1	P	R	F1
		English			**Spanish**	
Baseline	-	-	61.20	-	-	57.40
BERT+2×T	82.99	86.10	84.52	83.55	80.78	82.14
2×BERT+2×T	86.14	89.92	87.99	85.21	82.31	83.74
2×BERT+2×T-context w/ entities	**87.86**	90.00	88.92	85.80	**85.80**	**85.80**
2×BERT+2×T-context w/o entities	87.83	**90.33**	**89.06**	**86.90**	84.61	85.74
		Dutch			**Russian**	
Baseline	-	-	61.60	-	-	59.10
BERT+2×T	83.73	84.10	83.92	79.33	79.17	79.25
2×BERT+2×T	86.51	86.43	86.47	78.02	**81.77**	**79.85**
2×BERT+2×T-context w/ entities	86.46	**88.24**	**87.34**	**80.00**	77.93	78.95
2×BERT+2×T-context w/o entities	**86.77**	87.29	87.03	78.28	74.38	76.28
		Turkish			**Korean**	
Baseline	-	-	45.70	-	-	54.60
BERT+2×T	84.58	86.35	85.45	83.30	83.49	83.39
2×BERT+2×T	84.67	**87.39**	**86.01**	82.22	84.87	83.52
2×BERT+2×T-context w/ entities	84.11	87.15	85.60	83.17	84.25	83.71
2×BERT+2×T-context w/o entities	**85.03**	86.67	85.84	**86.58**	**87.71**	**87.14**
		Farsi			**German**	
Baseline	-	-	51.80	-	-	63.40
BERT+2×T	77.65	**82.19**	79.86	89.47	89.18	89.33
2×BERT+2×T	**80.69**	81.29	**80.99**	89.86	90.15	90.01
2×BERT+2×T-context w/ entities	78.99	80.87	79.92	**90.24**	**91.04**	**90.64**
2×BERT+2×T-context w/o entities	78.18	82.11	80.10	89.94	90.96	90.45
		Chinese			**Hindi**	
Baseline	-	-	51.10	-	-	46.90
BERT+2×T	**88.20**	88.13	**88.17**	72.47	70.89	71.67
2×BERT+2×T	88.05	86.89	87.47	**77.57**	75.60	**76.57**
2×BERT+2×T-context w/ entities	86.74	86.81	86.77	76.04	75.12	75.58
2×BERT+2×T-context w/o entities	87.67	**88.29**	87.98	74.04	**76.81**	75.40
		Bangla			**Multilingual**	
Baseline	-	-	39.10	-	-	54.10
BERT+2×T	71.94	69.88	70.89	84.54	85.29	84.91
2×BERT+2×T	76.14	75.00	75.57	**85.76**	**86.66**	**86.21**
2×BERT+2×T-context w/ entities	**78.79**	**79.88**	**79.33**	84.91	85.10	85.00
2×BERT+2×T-context w/o entities	77.49	78.75	78.12	84.99	86.09	85.54
		Mixed				
Baseline	-	-	58.10	-	-	-
BERT+2×T	72.33	71.15	71.74	-	-	-
2×BERT+2×T	71.85	71.97	71.91	-	-	-
2×BERT+2×T-context w/ entities	71.97	71.97	71.97	-	-	-
2×BERT+2×T-context w/o entities	**73.92**	**72.95**	**73.43**	-	-	-

Table 3: MultiCoNER results on the development set.

tage of this additional context. First, we consider that simply concatenating the semantically relevant contexts as they are with the sentence in question are sufficient for bringing an improvement to the detection of entities. These models are referred to as *context w/ entities*.

We also examine the scenario where these additional contexts are added without any entity present, as shown in Figure 2 where the entities from the additional contexts are replaced with a special token <ENT> (Boros et al., 2021, 2022)). The idea behind this choice is that we are trying to improve the contextual information without confusing the detection of entities with the presence of others. We refer to these models as *context w/o entities*.

For finding semantically relevant contexts, we used a multilingual Sentence-BERT (SBERT) (Reimers and Gurevych, 2020b) model[2], a modified pre-trained BERT (Devlin et al., 2019b) that uses a siamese and triplet network structure to derive semantically meaningful sentence embeddings that can be compared using cosine similarity. We keep the top-10 semantically relevant contexts.

Hyperparameters For each language, including the multilingual and the code-mixed cases, we choose a language-specific BERT pre-trained model[3], as presented in Table 2, with the exception for Hindi and Bangla. For the models for which we do not have a language-specific model, we use multilingual BERT. For each language, we additionally use a multilingual model (XLM-RoBERTa-large (Conneau et al., 2020)), approaches referred as BERT$\times$2 in Table 3.

4 Experiments

Next, we perform a detailed error analysis of our approaches[4]. The evaluation is performed in terms of macro precision (P), recall (R), and F1. Our results are presented in Table 3. Each type of approach is detailed with the corresponding pre-trained models.

Table 3 presents the results for our preliminary results on the provided dev set. We, first, observe that all our results considerably outperform the baseline scores provided by the organizers. Also, overall, it is clear that adding a multilingual language model to an already language-specific model brings an increase in performance. Next, we notice that adding *context w/ entities* slightly improved the performance of a limited number of languages (English, Dutch, Korean, German, Bangla) and *context w/o* improved considerably the performance of a larger number of languages (English, Spanish, Dutch, Korean, German, Chinese, Bangla). This allows us to understand that the presence of entities in the additional contexts can influence the detection of the entities of interest by hindering the importance of context with other entities. Thus, adding context brings performance improvements (marginally, with entities, or considerably, without entities). However, the fact that we used the already seen contexts from similar domains or topics (multilingual train data) could also be a factor that contributed to the drop in F1 for some of the languages.

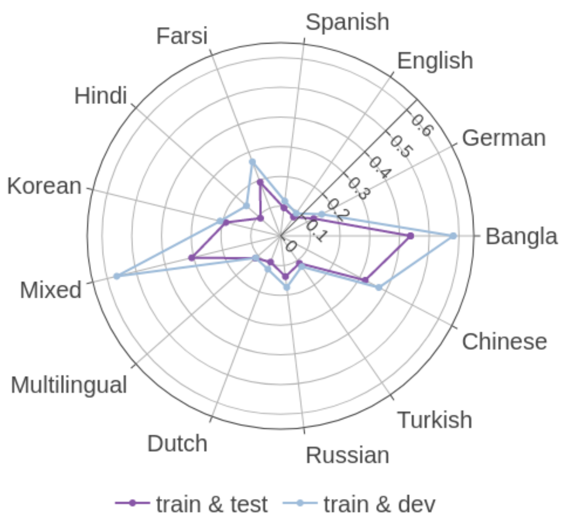


Figure 3: Similarity scores between the topics of the train and test (purple) & train and dev (blue) sets for each language.

Error Analysis Since one of the main challenges and purposes of SemEval-2022 Task 11 is to test not only the language adaption, but also the domain capability of a NER, we analyze the ability of our proposed models to handle multi-domain data. To measure the domain distribution among datasets, we obtain their topic vectors by a joint representation of documents and word semantic embeddings as in Angelov (2020). For this, we first obtain a common embedding of sentences and words with a pre-trained multilingual Sentence-BERT model[5] (Reimers and Gurevych, 2020a). Second, we utilize UMAP (McInnes et al., 2018) to create embeddings of lower dimension for all sentence vectors and find dense areas of sentences with HDBSCAN (Campello et al., 2013). Finally, we calculate the centroid of each dense area that corresponds to the topic vector.

Topic Detection Hyperparameters We set UMAP to a 5-dimensional space with a default

[2]We used the MULTI-QA-MPNET-BASE-DOT-V1 model.

[3]All models can be found at `https://huggingface.co`.

[4]Our code is available at `https://github.com/EMBEDDIA/stacked-ner`

[5]We use the PARAPHRASE-MULTILINGUAL-MINILM-L12-V2 model.

Split	Hyperparameter		
		neigh	dist
	2	5	0.0
Train	Mixed	Bangla	Mixed, Dutch, Chinese, Bangla
Dev	Korean, Mixed, Bangla	Farsi, Dutch	Mixed, Dutch, Chinese, Bangla, Korean
Test	-	-	Mixed, Chinese

Table 4: UMAP hyperparameters.

size of the local neighborhood (`neigh`) of 15, an effective minimum distance between embedded points (`dist`) equal to 0.1, and cosine distance as a similarity metric. We tuned these hyperparameters depending on the language and split when the obtained number of topic vectors was less than 2. These values are summarized in Table 4. Regarding HDBSCAN, we fix the minimum number of samples in a cluster to 15.

Language	**Domains / Topics**		
	Train	**Dev**	**Test**
English	72	12	1,188
Spanish	99	8	1,274
Dutch	112	9	1,404
Russian	95	6	1,330
Turkish	101	8	1,088
Korean	107	10	854
Farsi	126	5	1,097
German	93	6	1,194
Chinese	89	8	607
Hindi	105	5	1,209
Bangla	55	6	142
Multilingual	573	55	1,551
Mixed	32	11	644

Table 5: Number of topics in the dataset splits per language.

As seen in Table 5, the number of topics within the train set for each language is on average 96. The multilingual dataset has roughly six times more topics than the monolingual dataset even though it is the concatenation of all monolingual datasets. This suggests that certain portions of the monolingual datasets were obtained by a translation mechanism. Topic diversity from the development set is clearly smaller than the train set given the limited amount of samples it contains. Monolingual test sets present on average 11 times more topics with respect to the train sets, which is explained by the big amount of out-of-domain data organizers added to test sets with the objective of measuring out-of-domain performance.

To estimate the amount of out-of-domain sentences within data sets, we compute the topic overlap between (train and dev) and (train and test) sets. For each topic vector in the train set, we compute the cosine similarity with the topic vectors of the corresponding test or dev set. Then, we compare the topics by calculating the mean of the cosine similarities between the two sets of topics, as shown in Figure 3. We observe that Bangla and Mixed have very similar topics in the train, test, and dev sets. Our results in Table 3 seem to prove otherwise, thus, we interpret this behavior as the result of the lack of training of the multilingual Sentence-BERT model for this language.

Table 3 shows improvements when adding contexts for English, Spanish, Dutch, Korean, Bangla, and Mixed. With the exception of Bangla and Mixed that we just discussed, we notice that for the other languages, the similarity between the topics is rather small (around 0.2). Table 3 also shows a decrease in performance for Russian, Turkish, Farsi, Chinese, and Hindi. While for Farsi and Chinese, the similarity of topics is slightly higher (between 0.3 and 0.4), for the other languages, it plateaus at 0.2, which could indicate that there is a correlation between the number of overlapping topics between the train, test, and dev sets.

Metric	**Correlation (train, dev)**	**Correlation (train, test)**
P	-0.3812	-0.4995
R	-0.3450	-0.3266
F1	-0.3648	-0.4070

Table 6: Pearson correlation values between the dev and train topic similarity and evaluation metrics.

We, therefore, decide to compute the Pearson correlation between the precision, recall, and F1, and the similarity between topics. We observe weak negative correlation coefficients, allowing us to understand that the higher the topical similarity, the lower the performance scores. These

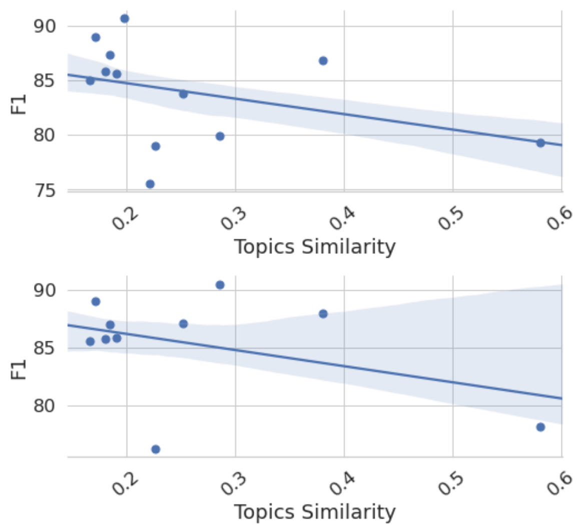


Figure 4: Topics similarity between train and dev set, 2×BERT+2×T-context *w/ entities* (upper figure), train and dev sets, 2×BERT+2×T-context *w/o entities* (lower figure), versus the F1 scores for the context models.

correlation values are shown in Table 6. Moreover, in order to understand if there is a correlation between the number of overlapping topics and the F1 scores for the models that use *context w/* or *w/o entities*, we draw a scatterplot of these two variables, then fit a regression model and plot the resulting regression line and a 95% confidence interval for that regression, in Figure 4. We observe again that, generally, the higher the similarity between topics in test and dev, the lower the F1 scores are.

SemEval-2022 Task 11 In the official SemEval-2022 Task 11, our best results ranked above the average for English, Spanish, and German, close to the average for Dutch, Turkish, Farsi, and Chinese. For the other languages, our approach obtained lower scores (Russian, Korean, Hindi, Bangla, Multilingual, and Mixed). For Hindi and Bangla, we did not have a specialized language model (we used multilingual BERT), which is clearly another reason for which the results were the lowest. Interestingly, we expected higher results for Korean and Spanish, but the size of the test set was most probably another important factor to consider.

5 Conclusions

In this paper, we presented a straightforward approach for adding semantically relevant contexts for NER in the monolingual, multilingual, and code-mixed datasets provided by SemEval-2022 Task 11 *Multilingual Complex Named Entity Recognition* (MultiCoNER). Our findings show that, while adding contexts from the train set, with and without entities, is promising, the topics or domains overlap could influence the performance in both directions. Future work will include the automatic generation of semantically relevant contexts without the presence of entities.

Acknowledgements

This work has been supported by the European Union's Horizon 2020 research and innovation program under grants 770299 (NewsEye) and 825153 (EMBEDDIA), and by the ANNA and Termitrad projects funded by the Nouvelle-Aquitaine Region.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Gabriel Bernier-Colborne and Philippe Langlais. 2020. Hardeval: Focusing on challenging tokens to assess robustness of ner. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1704–1711.

Emanuela Boros, Ahmed Hamdi, Elvys Linhares Pontes, Luis Adrián Cabrera-Diego, Jose G. Moreno, Nicolas Sidere, and Antoine Doucet. 2020a. Alleviating Digitization Errors in Named Entity Recognition for Historical Documents. In *Proceedings of the 24th Conference on Computational Natural Language Learning (CoNLL)*, pages 431–441, Online. Association for Computational Linguistics.

Emanuela Boros, Jose G Moreno, and Antoine Doucet. 2021. Event detection with entity markers. In *European Conference on Information Retrieval*, pages 233–240. Springer.

Emanuela Boros, José G Moreno, and Antoine Doucet. 2022. Exploring entities in event detection as question answering. In *European Conference on Information Retrieval*, pages 65–79. Springer.

Emanuela Boros, Elvys Linhares Pontes, Luis Adrián Cabrera-Diego, Ahmed Hamdi, José Moreno, Nicolas Sidère, and Antoine Doucet. 2020b. Robust named entity recognition and linking on historical multilingual documents. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum*, volume 2696, pages 1–17, Thessaloniki, Greece. CEUR-WS.

Emanuela Boros, Verónica Romero, Martin Maarand, Kateřina Zenklová, Jitka Křečková, Enrique Vidal, Dominique Stutzmann, and Christopher Kermorvant. 2020c. A comparison of sequential and combined approaches for named entity recognition in a corpus of handwritten medieval charters. In *2020 17th International conference on frontiers in handwriting recognition (ICFHR)*, pages 79–84. IEEE.

Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172, Berlin, Heidelberg. Springer Berlin Heidelberg.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8440–8451. Association for Computational Linguistics.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 20 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2983–2989.

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

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Ralph Grishman and Beth M Sundheim. 1996. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Stephen Mayhew, Tatiana Tsygankova, and Dan Roth. 2019. ner and pos when nothing is capitalized. *arXiv preprint arXiv:1903.11222*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving named entity recognition with attentive ensemble of syntactic information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4231–4245, Online. Association for Computational Linguistics.

Lakshya Rajoria. 2021. Named entity recognition in tweets. *International Journal of Research in Engineering, Science and Management*, 4(1):43–50.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020a. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020b. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, volume 30, pages 5998–6008, Long Beach, CA, USA. Curran Associates, Inc.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. Automated concatenation of embeddings for structured prediction. *arXiv preprint arXiv:2010.05006*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. *arXiv preprint arXiv:2105.03654*.

Genta Indra Winata, Samuel Cahyawijaya, Zihan Liu, Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2021. Are multilingual models effective in code-switching? In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 142–153, Online. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*.

MarSan at SemEval-2022 Task 11: Multilingual complex named entity recognition using T5 and transformer encoder

Ehsan Tavan[1,*], **Maryam Najafi**[1,*]
[1] NLP Department, Part AI Research Center, Tehran, Iran
{ehsan.tavan, maryam.najafi}@partdp.ai

Abstract

The multilingual complex named entity recognition task of SemEval2020 required participants to detect semantically ambiguous and complex entities in 11 languages. In order to participate in this competition, a deep learning model is being used with the T5 text-to-text language model and its multilingual version, MT5, along with the transformer's encoder module. The subtoken check has also been introduced, resulting in a 4% increase in the model F1-score in English. We also examined the use of the BPEmb model for converting input tokens to representation vectors in this research. A performance evaluation of the proposed entity detection model is presented at the end of this paper. Six different scenarios were defined, and the proposed model was evaluated in each scenario within the English development set. Our model is also evaluated in other languages.

1 Introduction

Named Entity Recognition (NER) is a key component of Natural Language Processing (NLP) assigned to identify regions of text that contain references to entities. It is the process of identifying the informative part of data or applicable labels from unstructured data. In NER, data is gathered from unstructured data such as emails, blogs, newspapers, tweets, etc., to extract meaningful information.

To put it another way, the term NER refers to identifying token spans of entities mentioned in the text and classifying them into a set of predetermined categories. The system finds entities from unstructured data and organizes them into multiple categories. As an extension of NLP, the field of NER can be considered as Information Extraction (IE).

For NLP, IEs are among the trending fields and play an essential role in the following tasks: Find and understand limited relevant parts of texts, Gather information from many pieces of text, and Produce the unified representation of all the relevant information. The NER problem falls into a general class of NLP problems called sequence tagging. Part of Speech (POS) tagging and chunking are sequence tagging NLP tasks in addition to NER. It is possible to detect NER in three ways: flat NER, nested NER, and discontinuous NER. Nested NER has overlapping in the span of text Yan et al. (2021). Most approaches only target flat entities, ignoring nested structures common in many scenarios. It is challenging to identify spans as well as types of named entities in text Lample et al. (2016). So to find the best architecture, we implement a transformer-based language model in this research. NLP has significantly benefited from transfer learning in recent years. Transfer learning gains power and effectiveness from pre-training on large, unlabeled text datasets. The model can then be fine-tuned to a smaller labeled dataset, resulting in better performance. Many models have achieved success in this field, including the Text-To-Text Transfer Transformer (T5) Raffel et al. (2019). The T5 is a pre-trained encoder-decoder language model that employs the "text-to-text" format to accomplish all types of NLP work, including generation, translation, and summarization tasks.

In SemEval-2022 task 11 Malmasi et al. (2022b), a multilingual complex NER is provided. Languages presented in Malmasi et al. (2022a) are Bangla, German, English, Spanish, Farsi, Hindi, Korean, Dutch, Russian, Turkish, and Chinese. Since this dataset contains words from different languages, it is challenging to choose an appropriate word representation for converting them into their corresponding vectors. The multilingual nature of this task necessitated the selection of the multilingual variant of Google's T5 model named MT5 Xue et al. (2020) that had already been trained on a database of more than 101 languages and con-

[*] Equal contribution. Listing order is random.

tained up to 13 billion parameters. In this paper, MT5 is used as the main Embedding.

We evaluated the proposed model using the English test set and achieved the F1-score of 71.45% as part of this competition. In our next step, we considered this model in other subtasks, and our rank varied from 9 to 21 depending on which subtask we evaluated. Our code is available at GitHub[1] for researchers.

The contributions of this paper are summarized as follows. Section 2 introduces previous attempts in the NER. In Section 3, information about the task and datasets is presented. We then offer a deep learning framework for recognizing named entities in Section 4. Section 5 details the experimental setup, while Section 6 presents the results of the experiments. Section 7 presents both quantitative and qualitative error analysis. We conclude our paper in Section 8.

2 Background

The NER field has undergone enormous changes in recent years Meng et al. (2021) Fetahu et al. (2021). As mentioned before, NER is the process of identifying relevant objects such as persons, products, genes, place, organization, etc., that are mentioned in the string of the text, sentence, or paragraph. NER typically forms the basis of other tasks such as event detection from news, online shopping customer service, knowledge graph construction, and biological analysis Bokharaeian et al. (2017).

Yu et al. (2020) stated that since NER tags are nested, it uses the graph-based dependency parsing method and examines eight separate corpora to achieve State-of-the-art for all. The embedding layer in this article was composed of BERT, fasttext, and character embedding. There is widespread usage of CRF in the field of NER. CRF is used for the first time in Collobert et al. (2011) for the NER. The representation of the sample in this research was obtained by Convolutional Neural Network(CNN). After that, many articles used CRF in various languages and combined it with other methods, such as Part Of Speech Tagging(POS), Long Short-Term Memory(LSTM), Embeddings from Language Models(ELMo), etc.Lample et al. (2016); Alves-Pinto et al. (2022); Rajan and Salgaonkar (2022); Ma and Hovy (2016); Huang et al. (2015). Peters et al. (2018) which is known as ELMo, extends LSTM-CRF and leverages pre-trained word-level language models for better context-aware representations. Peters et al. (2018) focuses on introducing and defining a Bidirectional language model that is tested on numerous NLP tasks in 2018. The accuracy of this research of NER with the language model and CRF layer was 93.42%.

To resolve ambiguity in NER tags, Straková et al. (2019) encoded nested entities in a sequence (seq2seq) and prepared an LSTM-CRF-based model. By incorporating pre-trained character-level language models into Flair Akbik et al. (2019), researcher presented contextualized representations. The following year, Akbik et al. extended this model to incorporate dataset-level word embeddings, dynamically aggregating embeddings and implementing pooling to extract a global word representation from all instances Akbik et al. (2018).

Word representations and character representations are used in Ma and Hovy (2016), an end-to-end system designed using LSTM, CNN, and CRF. This idea has been implemented on the Penn Treebank WSJ corpus Marcus et al. (1999) for POS and CoNLL 2003 Sang and De Meulder (2003) for NER datasets. A CNN has been used in this research to extract the character-level representation of words, with its output then being input into LSTMs, followed by a CRF layer. There is another extension to CRF known as hybrid semi-Markov conditional random fields (HSCRFs) is explained in Ye and Ling (2018) by contributing word-level labels in the building of SCRFs Lafferty et al. (2001). The purpose of using word-level tags to derive segment scores is to obtain segment scores.

To solve different sequence tagging models with a CRF inference, Yang and Zhang (2018) implemented NCRF++ with three steps: a character sequence layer, a word sequence layer, and an inference layer. RodrigoAgerri et al. presented a multilingual NER system that combines many features to cluster based on local information Agerri and Rigau (2016). Results from standard task evaluation data such as CoNLL for English, Spanish, and Dutch were reposted.

According to Wang et al. (2020), contextualized language models can produce better results when different embeddings are combined. This paper presents a framework for generating and scoring the output of embedded combinations using rein-

[1]`https://github.com/MarSanTeam/Complex_NER_SemEval`

forcement learning, which achieves the best accuracy in 6 different fields and 21 large datasets. According to Wang et al. (2020), contextualized language models can produce better results when different embeddings are combined. By rewarding model scores for better concatenations of embeddings, can propose Automated Concatenation of Embeddings to find better concatenations of embeddings for structured prediction tasks.

Majumder et al. (2022) considers the issue of informal data, whose unstructured and incomplete nature makes the process more challenging. To solve the mentioned challenge, Bi-LSTM based architecture for informal tweets in Hindi and English was implemented. There is more than one way to do it in this field. Finding the part of the text containing entity information, classifying and identifying the right entity, and applying that entity to the appropriate part of the text are just some ways. Generating tags is one other way to implement them. As mentioned in Yan et al. (2021), Hang Yan et al. propose that they use a novel and simple Bidirectional Auto-Regressive Transformer(BART) sequence-to-sequence (Seq2Seq) framework that uses a pointer mechanism Vinyals et al. (2015) to generate the entity sequence directly.

Another approach in NER, which is mentioned in Islam et al. (2022), consists of using an attention mechanism to minimize the problem of detecting redundant and inessential data and ignoring them entirely. Combining semantic, glyph, and phonetic features to improve the expression ability of Chinese character embedding, Li and Meng (2021) proposes an architecture based on Fusion Embedding for the Chinese language.

There is also a paper for the Chinese language entitled Jia et al. (2020) that identifies entities from Chinese social media texts, using uncertain information from word segmentation. Researchers have proposed that interactions between spans of tokens can help determine discontinuous mentions and have developed a transition-based model with a generic neural encoding to be able to detect discontinuous mentions Khan et al. (2020).

A model of bidirectional transformers is presented in Yamada et al. (2020), which produces a contextualized representation of words and tokens. BERT's masked language model is used as pre-trained word embeddings. As a result, Yamada et al. present advancement in attention known as entity-aware self-attention mechanisms and achieve state-of-the-art in five benchmarks that include: Open Entity (entity typing), TACRED (relation classification), CoNLL-2003 (NER), ReCoRD (cloze-style question answering), and SQuAD 1.1 (extractive question answering). Since carelessness or a lack of background knowledge of annotators might lead to model performance errors, some research has been conducted to identify and solve these issues. Wang et al. (2019) provides a framework for finding human errors in NER annotations. Following the correction of labels in the test set, they re-evaluated state models in NER, claiming that the results were more accurate than the original test set. This paper's main idea is cross weighs, which accommodates label mistakes during training and then trains a more robust NER model.

Wang et al. (2021) finds the external context of input sentences by retrieving relevant sentences. The process of selecting the top similar text involves re-ranking retrieved samples according to their semantic relevance to the input sentence. As a result, the inputs are the concatenation of input sentences and external contexts. Both input types are used to implement Cooperative Learning (CL), and different representations are encouraged to produce similar contextual representations or output label distributions. Results on eight other NER datasets achieve state-of-the-art results. But one drawback of this method is that there are no document-level contexts in practice.

3 Task Description

Our investigation aims to comparatively study MultiCoNER-2022 datasets that have been considered individually for complex NER systems in 11 languages. Under short and low-context settings, the task detects semantically ambiguous and complex entities.

Languages presented in MultiCoNER-2022 are: English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, and Bangla. The number of instances for each language varies between 150k to 500k. There are 15300 train data and 800 validation data for each language.

This dataset is labeled with the following tags: PER: Person, LOC: Location, GRP: Group, CORP: Corporation, PROD: Product, CW: Creative Work. There is detailed information of datasets illustrated in Table 1.

The datasets used in this task include sentences labeled with the IOB format. Using this format,

Languages	LOC		PER		PROD		GRP		CW		CORP	
	Span	Token	Span	Token	Span	Token	Span	Token	Span	Token	Span	Token
English	4799	7550	5397	11538	2923	4723	3571	10038	3752	9782	3111	6222
Spanish	4968	7204	4706	9999	3040	4404	3226	7993	3690	8734	2898	5208
Dutch	5529	6767	4408	9369	2935	3572	3306	7204	3340	7142	2813	4544
Russian	5529	6767	4408	9369	2935	3572	3306	7204	3340	7142	2813	1731
Turkish	5804	6862	4414	8446	3184	4392	3568	6649	3574	7715	2761	4420
Korean	6299	6837	4536	7171	3082	4165	3530	5525	3883	3665	3313	1370
Farsi	5683	8720	4272	8613	2955	4496	3199	7676	3694	7528	2991	5382
German	4778	6566	5288	11230	2961	3898	3509	5878	3507	9054	3083	6210
Chinese	6986	28762	2225	14048	4854	16084	713	3200	5248	18817	3805	18069
Hindi	2614	4218	2418	5254	3077	2295	2843	8664	2304	5896	2700	5617
Bangla	2351	3804	2606	5738	3188	5152	2405	6653	2157	5001	2598	5299
Code-Mixed	325	493	296	680	316	560	248	677	298	755	294	653

Table 1: Distribution of spans and tokens for each entity

tokens that are not a part of an entity are tagged as ’O’, the first token of an entity is represented by the ’B’ tag, while the rest of the entity’s tokens are represented by an ’I’ tag. The entity category precedes both a hyphen and the "B" and "I" tags. Therefore, NER is a task that labels tokens according to their text, which is multi-class token classification.

4 System overview

In this section, we will introduce our proposed NER framework. The proposed framework consists of three parts:

1. Word Representation Module
2. Feature extraction Module
3. Prediction Module

The proposed architecture takes the token sequence $S = \{s_1, s_2, s_3, ..., s_n\}$, and predicts entity sequence $O = \{o_1, o_2, o_3, ..., o_n\}$ as output. Figure 1 is an illustration of the proposed architecture.

4.1 Word Representation Module

In light of the multilingual nature of the SemEval NER task, the T5 language model was applied to convert tokens into their representation vector. In the word representation module, the last hidden state of the T5-large encoder is chosen to learn the k-dimensional (1024 here) representation for input tokens. The T5 uses SentencePiece encoding and assigns named entity tags to its extracted tokens. Tokens extracted from T5 are always equal to or greater than main tokens. Each token thus becomes one or more subtokens. The label of the first subtoken corresponds to the label of the first token, while the other subtokens have the label X.

Subtoken Check A key consideration is that each token becomes one or more subtokens. To provide better training, a feature called subtoken check is used. This feature checks whether the input token is tokenized into the subtoken or not. After tokenizing the tokens, the first subtoken of each token has a value of 1, and the rest have a value of 0. Hence, the T5 encoder takes two input sequences of the same length; one is the subtoken index, and the other is the subtoken check index. After adding this feature and improving the results, it was found that in token-based tasks such as NER, the existence of this feature is extremely helpful for managing subtokens.

Byte-pair Embeddings The Byte-pair Embeddings(BPEmb) Heinzerling and Strube (2017) consists of pre-trained subword embeddings in 275 languages. BPEmb is a variable-length encoding that views the text as a sequence of symbols, iteratively merging the pair with the highest frequency into a new symbol. It provides a mechanism for properly tokenizing input sequences so that unknown tokens can prepare appropriate representations by using subtokens. To predict the named entity tag for the input sequences, we concatenate the output vector of the BPEmb model with the output vector of the feature extraction layer since fine-tuning of the model is not possible during training.

4.2 Feature Extraction Layer

Since the goal of NER is to predict the entity label of each token, an awareness of the semantic dependencies between tokens can be extremely helpful. The which uses the multi-encoder architecture Vaswani et al. (2017), which uses the multi-head self-attention mechanism, is one of the most suitable deep learning architectures for extracting relation between tokens. There are two sublayers in this encoder module. The first sublayer is a multi-head self-attention mechanism, while the second is

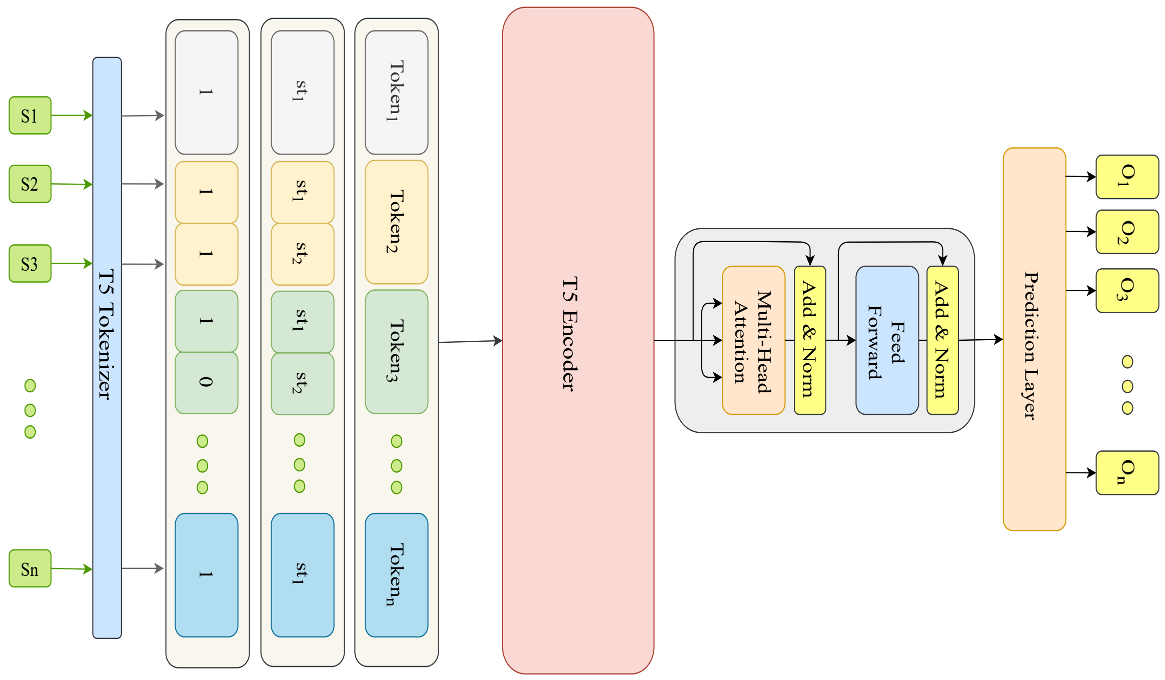


Figure 1: Proposed multilingual NER architecture

a position-wise fully connected feed-forward network. This architecture uses residual connections around each of the two sublayers followed by layer normalization. A multi-head attention module comprises several scaled dot-product attention used in parallel. In scaled dot-product attention, the input consists of three matrices Q, K, and V. The scaled dot-product attention is calculated using the following formula.

$$W_i^Q, W_i^K, W_i^V \in R^{d_{model} \times d_k}$$

$$Q = XW_Q, K = XW_K, V = XW_V \quad (1)$$

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

The attention module has three trainable parameters, W_Q, W_K, and W_V. The three matrices Q, K, V are constructed by multiplying the input vector X by the corresponding matrices W_Q, W_K, W_V. Consequently, the dot product between Q and K is divided by $\sqrt{d_k}$ in order to prevent the dot product from becoming too large.

4.3 Prediction Layer

The vector obtained in the feature extraction Layer is given to a fully connected layer to predict the named entity label of the input sequence. For the input sequence $S = s_1, s_2, ..., s_n$ the output sequence $O = o_1, o_2 \ldots, o_n$ is predicted.

5 Experimental setup

We implemented the model in PyTorch and trained it on Nvidia V100 GPUs. The AdamW optimizer with a learning rate of 2e-5 is used to train the network. Our training method includes early stopping, which ensures the validation loss reduction with patience of 10 epochs. The training batch size is set to 32, and the dropout rate is 0.2. Transformer encoders have eight attention heads, and position-wise feed-forward layers have 2048 hidden sizes. In both T5 and MT5 tokenizers, the max length varies between 100 and 250 characters according to the evaluated language. The hyper-parameters of each subtask were tuned with the dev set. All other parameters are initialized randomly.

6 Results

Several experiments have been conducted to develop the most appropriate model for NER. Experiments with the English dataset can be found in Table 2. Due to the success of T5 in this study, this language model has been used to compute the word representation vectors. According to Table 2, using the T5 can improve F1-score by 4% compared to MultiCoNER Baseline that uses XLM-RoBERTa.

We have attempted to improve the language model results by adding deep learning architectures and textual features Tavan et al. (2021). We evaluated LSTM and Transformer architecture on top of the T5 and found that using the transformer improved the F1-score further than LSTM. After various experiments, it was found that the use of BPEmb could not improve the F1-score of the model.

Models	Train	Dev	Test
MultiCoNER Baseline	-	77.60	-
T5	94.40	81.92	65.99
T5 + LSTM	96.40	82.20	67.54
T5 + Transformer	89.09	82.91	67.63
T5 + Transformer + subtoken + bpemb	90.12	82.36	67.22
T5 + subtoken + Transformer (ours)	**97.32**	**86.73**	**71.45**

Table 2: Experiments with English dataset

According to Table 2, the proposed model has achieved the F1-score of 86.73% and 71.45%, on the dev and test dataset for English, respectively, which is the highest F1-score among the other experiments. Figure 2 Compares the F1-score of different deep learning architecture.

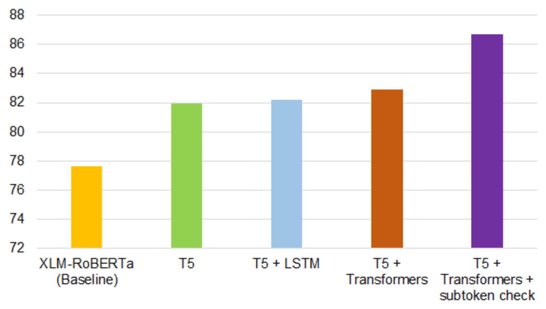


Figure 2: Compares the F1-score of deep learning architectures.

The results of the proposed model on the test dataset, as well as its ranking in the competition, are shown in Table 3. According to these results, Chinese, Hindi, and Bangla have lower F1-scores than other languages like German, Dutch, and English due to their language complexity.

The results of proposed model for the different entities are shown in Table 4. According to Table 4, the "PER" entity has reached the highest F1-score among other entities in all languages. Except for Russian, Turkish and Chinese, "CW" has the lowest F1-score in all other languages. As a result, the proposed model is weaker in identifying the "CW" entity than other entities.

7 Error Analysis

Several scenarios could occur when comparing the golden standard annotation with the output of a NER system Nejadgholi et al. (2020):

Scenario 1, Complete True Positive: An entity is predicted by the NER model correctly.

Scenario 2, Complete False Positive: An entity is predicted by the NER model but is not annotated in the hand-labeled text.

Scenario 3, Complete False Negative: The model does not predict a hand-labeled entity.

Scenario 4, Wrong label, Right Span: A hand-labeled entity and a predicted one have the same span but different tags.

Scenario 5, Right label, overlapping spans: A hand-labeled entity and a predicted one have overlapping spans and the same tags.

Scenario 6, Wrong label, overlapping spans: A hand-labeled entity and a predicted one have overlapping spans but different tags.

The output of the proposed model on the English dev dataset has been evaluated on six scenarios in Table 5. From Table 5, 96.20% of "PER" entities are in Scenario 1, and the most significant proportion of Complete True Positives are related to this entity. Approximately 17% of "PROD" entities are in Scenario 2, higher than other entities. "PROD" actually has the highest Complete False Positive value among all entities. The "CW" has the greatest number of entities in Scenario 3 and the highest proportion of Complete False Negatives among other entities.

Scenario 4 includes 6.21% of "CORP" entities. Scenario 5 has the most significant number of entities compared to other scenarios, having 6.12 percent of the "PROD" entities. The "CORP" also has the highest number of entities in Scenario 6. Table 6 shows examples of the English test samples that are categorized in different scenarios.

Language	Precision	Recall	F1-score
English (17)	71.11	71.91	71.45
Spanish (11)	68.65	68.71	68.30
Dutch (10)	71.18	71.98	71.13
Russian (10)	66.83	68.44	67.49
Turkish (10)	60.22	62.70	61.09
Korean (14)	61.13	63.92	62.26
Farsi (9)	61.80	63.06	62.14
German (12)	73.10	73.60	72.12
Chinese (19)	60.15	57.10	56.64
Hindi (12)	56.39	57.01	56.31
Bangla (11)	56.48	53.77	54.22
Multilingual (14)	69.38	70.47	69.28
Code-Mixed (21)	67.36	67.41	67.03

Table 3: Precision, Recall and F1-score on test dataset in all languages. The rank of the proposed model in each language is shown in parentheses.

Language	LOC	PER	PROD	GRP	CW	CORP
English	72.23	86.71	70.09	67.62	62.32	69.70
Spanish	66.95	84.77	63.50	63.89	61.31	69.40
Dutch	68.64	86.77	69.32	67.44	65.18	69.44
Russian	69.67	74.88	66.50	61.11	62.56	70.24
Turkish	64.29	70.76	63.83	52.31	54.93	60.44
Korean	71.48	68.32	59.81	60.13	50.12	63.72
Farsi	68.46	71.29	62.86	64.75	46.13	59.33
German	74.61	87.13	71.97	69.70	63.82	71.51
Chinese	67.93	57.59	64.38	34.08	51.95	63.93
Hindi	60.81	62.85	54.19	57.72	41.74	60.54
Bangla	57.04	67.26	51.12	64.63	33.03	52.22
Multilingual	74.19	81.11	63.96	63.61	63.33	69.49
Code-Mixed	72.98	78.99	68.81	59.50	58.52	63.39

Table 4: F1-score in English test dataset for each entity

Entity	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6
LOC	215 (91.88%)	17 (7.26%)	8 (3.41%)	2 (0.85%)	5 (2.13%)	4 (1.7%)
PER	**279 (96.20%)**	9 (3.1%)	1 (0.34%)	4 (1.37%)	1 (0.34%)	5 (1.72%)
PROD	117 (79.59%)	**25 (17.0%)**	18 (12.24%)	1 (0.68%)	**9 (6.12%)**	2 (1.36%)
GRP	168 (88.42%)	5 (2.63%)	3 (1.57%)	9 (4.73%)	2 (1.05%)	9 (4.73%)
CW	134 (76.13%)	22 (12.5%)	**22 (12.5%)**	7 (3.97%)	5 (2.84%)	8 (4.54%)
CORP	155 (80.31%)	7 (3.62%)	3 (1.55%)	**12 (6.21%)**	8 (4.14%)	**15 (7.77%)**

Table 5: Results of different scenario in English dev dataset.

Scenario 1	================================ HUMMAN ANOTATION ============================== in 1841, he established a production of whale oil . ================================ MODEL PREDICTION ================================ in 1841, he established a production of whale oil .
Scenario 2	================================ HUMMAN ANOTATION ============================== these desktop application launchers work with microsoft windows operating systems only. ================================ MODEL PREDICTION ================================ these desktop application launchers work with microsoft windows operating systems only.
Scenario 3	================================ HUMMAN ANOTATION ============================== upper head lug joins the head tube and top tube ================================ MODEL PREDICTION ================================ upper head lug joins the head tube and top tube
Scenario 4	================================ HUMMAN ANOTATION ============================== the caps were jointly designed by major league baseball and the new era cap company . ================================ MODEL PREDICTION ================================ the caps were jointly designed by major league baseball and the new era cap company .
Scenario 5	================================ HUMMAN ANOTATION ============================== molten chocolate and a piece of a chocolate bar ================================ MODEL PREDICTION ================================ molten chocolate and a piece of a chocolate bar
Scenario 6	================================ HUMMAN ANOTATION ============================== he was also the inventor of the nerf football. ================================ MODEL PREDICTION ================================ he was also the inventor of the nerf football .

Table 6: Example of different scenario in English dev dataset.
PROD 🟫 CW 🟥 CORP 🟩 GRP 🟦

8 Conclusion

This paper proposes a model that uses an encoder module of transformers in the top of the hidden state of the T5 to process the extracted features to obtain the most important feature representations. Many experiments were conducted to evaluate the model's performance and find the best language model. The experiments prove that our architecture is most compatible with the T5 language model and does cover a reasonable range of results. Since this dataset is multilingual, the MT5 embedding module was selected.

The model's accuracy is confirmed by an in-depth analysis of the provided datasets. Accordingly, the same architecture was used in all other sub-tasks. Error analysis enabled us to identify specific NER challenges, creating immediate future tasks. We plan to apply more robust deep architectures to multilingual datasets as part of our future work.

References

Rodrigo Agerri and German Rigau. 2016. Robust multilingual named entity recognition with shallow semi-supervised features. *Artificial Intelligence*, 238:63–82.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics*, pages 1638–1649.

Ana Alves-Pinto, Christoph Demus, Michael Spranger, Dirk Labudde, and Eleanor Hobley. 2022. Iterative named entity recognition with conditional random fields. *Applied Sciences*, 12(1):330.

Behrouz Bokharaeian, Alberto Diaz, Nasrin Taghizadeh, Hamidreza Chitsaz, and Ramyar Chavoshinejad. 2017. Snpphena: a corpus for extracting ranked associations of single-nucleotide polymorphisms and phenotypes from literature. *Journal of biomedical semantics*, 8(1):1–13.

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Benjamin Heinzerling and Michael Strube. 2017. Bpemb: Tokenization-free pre-trained subword embeddings in 275 languages. *arXiv preprint arXiv:1710.02187*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Tanvir Islam, Sakila Mahbin Zinat, Shamima Sukhi, and MF Mridha. 2022. A comprehensive study on attention-based ner. In *International Conference on Innovative Computing and Communications*, pages 665–681. Springer.

Shengbin Jia, Ling Ding, Xiaojun Chen, Yang Xiang, et al. 2020. Incorporating uncertain segmentation information into chinese ner for social media text. *arXiv preprint arXiv:2004.06384*.

Muhammad Raza Khan, Morteza Ziyadi, and Mohamed AbdelHady. 2020. Mt-bioner: Multi-task learning for biomedical named entity recognition using deep bidirectional transformers. *arXiv preprint arXiv:2001.08904*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Jiatong Li and Kui Meng. 2021. Mfe-ner: Multi-feature fusion embedding for chinese named entity recognition. *arXiv preprint arXiv:2109.07877*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Amit Majumder, Apurba Paul, and Abhishek Banerjee. 2022. Deep learning-based approach using word and character embedding for named entity recognition from hindi-english tweets. In *Applications of Networks, Sensors and Autonomous Systems Analytics*, pages 237–243. Springer.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mitchell P Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. *Linguistic Data Consortium, Philadelphia*, 14.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Isar Nejadgholi, Kathleen C Fraser, and Berry De Bruijn. 2020. Extensive error analysis and a learning-based evaluation of medical entity recognition systems to approximate user experience. *arXiv preprint arXiv:2006.05281*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Annie Rajan and Ambuja Salgaonkar. 2022. Named entity recognizer for konkani text. In *ICT with Intelligent Applications*, pages 687–702. Springer.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Jana Straková, Milan Straka, and Jan Hajič. 2019. Neural architectures for nested ner through linearization. *arXiv preprint arXiv:1908.06926*.

Ehsan Tavan, Ali Rahmati, Maryam Najafi, and Saeed Bibak. 2021. Bert-dre: Bert with deep recursive encoder for natural language sentence matching. *arXiv preprint arXiv:2111.02188*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *arXiv preprint arXiv:1506.03134*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. Automated concatenation of embeddings for structured prediction. *arXiv preprint arXiv:2010.05006*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. *arXiv preprint arXiv:2105.03654*.

Zihan Wang, Jingbo Shang, Liyuan Liu, Lihao Lu, Jiacheng Liu, and Jiawei Han. 2019. Crossweigh: Training named entity tagger from imperfect annotations. *arXiv preprint arXiv:1909.01441*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various ner subtasks. *arXiv preprint arXiv:2106.01223*.

Jie Yang and Yue Zhang. 2018. Ncrf++: An open-source neural sequence labeling toolkit. *arXiv preprint arXiv:1806.05626*.

Zhi-Xiu Ye and Zhen-Hua Ling. 2018. Hybrid semi-markov crf for neural sequence labeling. *arXiv preprint arXiv:1805.03838*.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. *arXiv preprint arXiv:2005.07150*.

SU-NLP at SemEval-2022 Task 11: Complex Named Entity Recognition with Entity Linking

Buse Çarık[*], **Fatih Beyhan**[*], **Reyyan Yeniterzi**
Sabancı University, İstanbul, Turkey
{busecarik,fatihbeyhan,reyyan}@sabanciuniv.edu

Abstract

This paper describes the system proposed by Sabancı University Natural Language Processing Group in the SemEval-2022 MultiCoNER task. We developed an unsupervised entity linking pipeline that detects potential entity mentions with the help of Wikipedia and also uses the corresponding Wikipedia context to help the classifier in finding the named entity type of that mention. The proposed pipeline significantly improved the performance, especially for complex entities in low-context settings.

1 Introduction

Named Entity Recognition (NER) is a widely studied task of Natural Language Processing. Recent leading architectures achieved impressive performances, especially in formal contexts (like news articles, etc.) and for commonly worked named entity types (like Person, Organization, etc.). In *Multilingual Complex Named Entity Recognition* task (Malmasi et al., 2022b), organizers introduced a dataset which is a large collection of short texts that includes complex entities. As they stated, complex entities can be noun phrases, gerunds, or full clauses (Malmasi et al., 2022a). The participants were challenged to develop NER systems for identifying these complex entities in short and low-context settings.

Recent contextual neural architectures like transformers work quite well on a variety of NLP tasks and datasets. However, they are strongly dependent on the context of the given text. In case of lack of context, these models alone may not gather enough evidence to make a correct prediction. When that is the case, getting help from knowledge bases can be useful. The dataset provided with this task has similar low-context characteristics together with some complex named entity types to be resolved. Therefore, in this work, we investigated whether performing entity linking to provide additional context for a transformer model can be useful for identifying complex named entities. We initially used an unsupervised entity linking approach to identify the possible entity mentions and their corresponding Wikipedia pages. Later on, we used the additional context retrieved from Wikipedia to predict the types of the identified entities. Performing entity linking prior to the classifier seems to be beneficial in the case of complex entities in short and low-context settings. Our code is publicly available[*].

[*]These authors contributed equally to this work.

2 Related Work

NER is an important and also well-studied task in NLP. Similar to other NLP tasks, recent neural architectures provided significant improvements in NER as well. However, since these models are heavily dependent on context information, applying them to short, noisy texts such as very short social media posts or web queries results in significant performance degradation (Meng et al., 2021).

To overcome the lack of context in these short texts, as well as the code-mixed terms, it has been prominent to use external knowledge in neural approaches. Recent studies (Meng et al., 2021; Fetahu et al., 2021) combining multilingual transformer models with gazetteers have shown remarkable improvement in performance. Another study (Yamada et al., 2015) utilized the Entity Linking task to address the issues of Twitter NER by detecting entity mentions from Wikipedia.

In this study, we also focused on external information due to the lack of context. However, instead of using a static auxiliary like gazetteers, we tried to detect entities dynamically from the continuously extending Wikipedia by using a search engine architecture. We integrated the Wikipedia content into the input representation in order to improve the performance of entity type classification.

[*]https://github.com/SU-NLP/SU-NLP-at-SemEval-2022-Task-11-Complex-Named-Entity-Recognition-with-Entity-Linking

3 Data

Malmasi et al. (2022a) introduced a new multilingual NER dataset consisting of 6 named entity categories (Person, Location, Group, Corporation, Product, and Creative Work) annotated in short texts for 11 languages. In this paper, even though we propose a language-independent approach, due to the limited resources we only focused on the Turkish part of the dataset. Statistics from this part are shown in Table 1. As seen from Table 1, even though the number of samples is much higher, the length of the samples is significantly lower in the test set compared to train and validation splits. This short length in context may cause state-of-the-art contextual models to perform worse than expected.

	# Samples	Avg. # Tokens
Train	15,300	14.27
Validation	800	14.27
Test	136,935	5.28

Table 1: Distribution of samples and average number of tokens per sample for Turkish part

An issue we found with the dataset is that in some examples, a phrase is labeled as *OTHER* although there are similar phrases that are labeled as a named entity. Consider the following example: *"Nokia 1011, 1994'e kadar, Nokia 2010 ve Nokia 2110'un halefler olarak tanıtılmasıyla üretime devam etti." English: "Nokia 1011 continued in production until 1994, with the introduction of the Nokia 2010 and Nokia 2110 as successors."*

The text includes three different cellphone models produced by Nokia. Two of these models (the blue ones) were annotated as *PRODUCT*; however, *Nokia 1011* was mislabeled as *OTHER*. A more major issue is when the same named entity occurs more than once in a sentence, but their annotations are different. An example is illustrated below: *"Whitney Houston yine bu sene içinde New Jack Swing'e adını yazdırdı, I'm Your Baby Tonight adlı şarkı Whitney Houston'a başarı üstüne başarı getirmiştir." English: "Whitney Houston made her name on New Jack Swing again this year, the song I'm Your Baby Tonight brought Whitney Houston success after success."*

In the example, Whitney Houston in blue was labeled as *PERSON*; but, the red Whitney Houston was labeled as *OTHER*. These types of wrong annotations cause ambiguities and learning problems in supervised approaches.

4 Methodology

In order to overcome the challenges described in previous sections, we propose an unsupervised entity linking pipeline followed by a classifier as our proposed NER architecture. The entity linking pipeline is designed to detect potential entity mentions and retrieve corresponding candidate documents for each mention. Later, these detected mentions and the linked content is used together in a classifier to detect the entity type of the mention. A pre-trained BERT model was also fine-tuned in order to create a strong baseline for comparison.

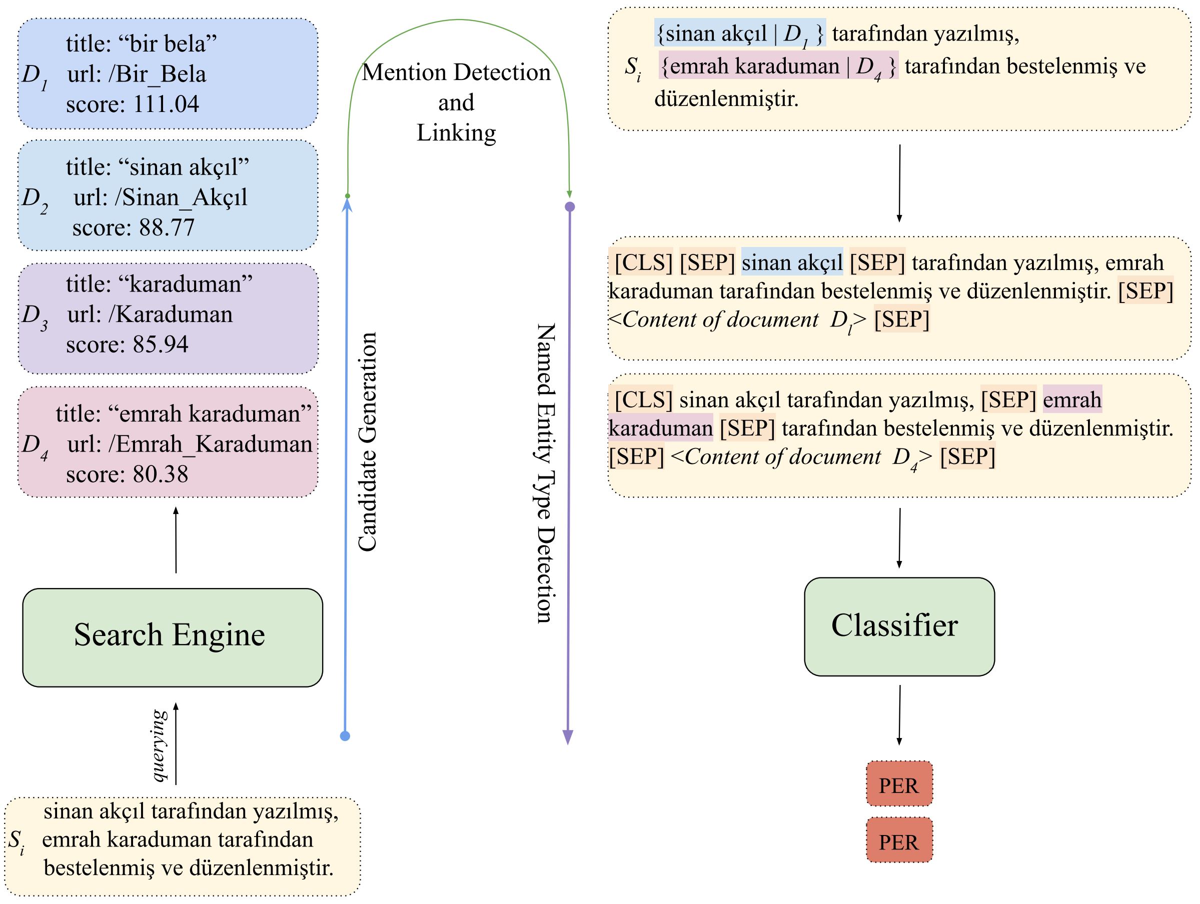
4.1 BERT

The BERT (Devlin et al., 2018) model was adopted as a baseline approach due to its outstanding performance in many NLP tasks. We fine-tuned the BERTurk (Schweter, 2020) model with a feed-forward layer on top to classify the named entities. The base model with 12 encoder layers and 768 hidden units was used. Since the data provided as part of the task was all lowercase, the uncased version of BERTurk was preferred.

4.2 NER with Entity Linking

BERT like transformer models heavily depend on the context. Therefore having a very short context (like a couple of words, not even a sentence) can be challenging for these models. Especially when the task is to identify complex named entities like creative work or product, it is much more challenging. In a very short context, even human beings may have a hard time resolving entities and their types.

In order to overcome this challenge, additional help from a gazetteer or a knowledge base will be very useful. Therefore in this paper, we propose using Wikipedia in order to both identify entities and also their types. Our proposed architecture which is depicted in Figure 1 consists of three steps. The first two steps work towards identifying possible entities and linking them to their corresponding Wikipedia articles. In this unsupervised entity linking approach as the first step, the original input text is used to retrieve relevant Wikipedia pages which are possible candidate pages for entities. In the second step, content within the retrieved documents (like their titles) is used to identify the entity mentions in the original text. Finally, in the third step, each identified mention and its linked Wikipedia article are used together to predict the type of the



title: "bir bela"
D_1 url: /Bir_Bela
score: 111.04
title: "sinan akçıl"
D_2 url: /Sinan_Akçıl
score: 88.77
title: "karaduman"
D_3 url: /Karaduman
score: 85.94
title: "emrah karaduman"
D_4 url: /Emrah_Karaduman
score: 80.38
Search Engine
querying
S_i sinan akçıl tarafından yazılmış, emrah karaduman tarafından bestelenmiş ve düzenlenmiştir.
Mention Detection and Linking
Candidate Generation
Named Entity Type Detection
S_i {sinan akçıl | D_1 } tarafından yazılmış, {emrah karaduman | D_4 } tarafından bestelenmiş ve düzenlenmiştir.
[CLS] [SEP] sinan akçıl [SEP] tarafından yazılmış, emrah karaduman tarafından bestelenmiş ve düzenlenmiştir. [SEP] <Content of document D_1> [SEP]
[CLS] sinan akçıl tarafından yazılmış, [SEP] emrah karaduman [SEP] tarafından bestelenmiş ve düzenlenmiştir. [SEP] <Content of document D_4> [SEP]
Classifier
PER
PER

Figure 1: Overall pipeline of the system which is showing the path of sentence S_i.

entity.

4.2.1 Candidate Generation

ElasticSearch[*] was used as the underlying search engine architecture. The most recent Wikipedia dump was used to create an index. For each document D_i, the *title*, *referred_by* and *interwikies* were indexed to create the following four fields:

- *title*: This is the title of the document D_i.
- *referred_by*: This is a set of text spans which are parsed from other Wikipedia pages where D_i is being referred with a interwiki link. This set includes all of these text spans sorted from longest to shortest. The title is added to the set as well.
- *interwikies*: This is the list of interwikies[*] in the document D_i.
- *all_text*: Concatenation of all fields (title, text content, interwikies, referred_by, and categories[*]) of the document D_i.

Each sample was queried and the most relevant 200 articles were retrieved. Later on for each query, documents retrieved with different field searches were pooled together. The reason we are pooling these different field results is to make sure we do not miss a possibly relevant article that could have been linked.

For the train and validation tests, around 199 documents were retrieved for each field. After pooling, we ended up with a pool of size 578 documents on average. Since test queries are shorter, the average number of documents returned got as low as 161 with the title field and 465 over the pool. Short queries also affected the empty ones, in other words, no document retrieved samples. After pooling there was not such a case for train and validation, however even pooling could not help the 169

[*] https://www.elastic.co/

[*] Interwikies are the types of hyperlinks which is linking a text span from a Wikipedia page to another Wikipedia page. For instance, Wikipedia pages of Earth, Mercury and Venus are in the interwikies section of the Sun's Wikipedia page, since they are mentioned within its text.

[*] The categories section of a Wikipedia page includes a list of topics which is related to the Wikipedia page.

Field	Train	Validation
title	96.52	97.00
interwikies	57.16	61.26
referred_by	96.46	96.11
all_text	86.50	86.68
After pooling	98.89	98.85

Table 2: Recall Scores over Detected Mentions

queries over 136,935 samples, and no Wikipedia articles were retrieved for those.

4.2.2 Mention Detection and Linking

After generating a set of candidate documents for each input text, the next step is to map these documents to the possible entity mentions in the input. For each input, our proposed algorithm iterated over the retrieved and pooled documents $\{D\}$, and cross-checked each item in the *referred_by* field of the document to see if there was an exact match to any phrase in the input text. In case there was an exact match, then that matched phrase was tagged as a possible entity and linked to the corresponding Wikipedia page. If there were multiple matches for a span of text, then the longest match was considered and shorter ones were ignored. Furthermore, if there was a tie in terms of length, then the relevance score of the Wikipedia document was used. That relevance score was calculated after pooling by doing a summation of the relevance scores from different field retrievals.

In order to see which fields in Wikipedia are more useful for mention detection, recall was calculated over detected entity mentions, without considering the type of entity. The results are summarized in Table 2. Based on the results *title* and *referred_by* are the most useful fields and pooling all retrievals is the best over both train and development sets.

4.2.3 Named Entity Type Detection

After linking the entities, the only thing remaining is to find their NER type. In order to do that we used the original context where the entity was mentioned in the input text and the linked Wikipedia article content. We combined these two as follows:

$$\texttt{[CLS]}\, ctx_l\, \texttt{[SEP]}\, M\, \texttt{[SEP]}\, ctx_r\, \texttt{[SEP]}\, WP\, \texttt{[SEP]}$$

where ctx_l and ctx_r are tokens for left and right context of mention, M is mention itself and WP is the first two paragraphs of a Wikipedia page. `[SEP]` tokens are used to tag the mention. The maximum length of the representation is set to 256. This representation is given to BERT for encoding.

We used two classifier heads following the BERT encoder for classifying the pairs of mentions M and Wikipedia contents WP. The proposed architectural design is presented in Figure 2. The first one is a binary classifier head which aims to solve an auxiliary task of whether the given candidate entity mention is a real entity or not with the help of input context tokens and Wikipedia content. The second classifier head tries to learn the type of the candidate named entity. The loss function for our model is the summation of losses from these two classifier heads. In case the binary classifier head returns O (means mention is not an entity), the second classifier head's decision is ignored, and the given candidate entity mention is directly predicted as O (which stands for Other, not an entity). This multi head model is referred to as *EL_MultiBERT*.

In order to test the effectiveness of multiple heads, we created another model called *EL_BERT*, which has a single classifier head only for identifying named entity types including the Other class.

In our pipeline, the Other class can be assigned to tokens in any of the three steps. Whenever a token is assigned the Other (O) label, then its label does not change during the remaining steps.

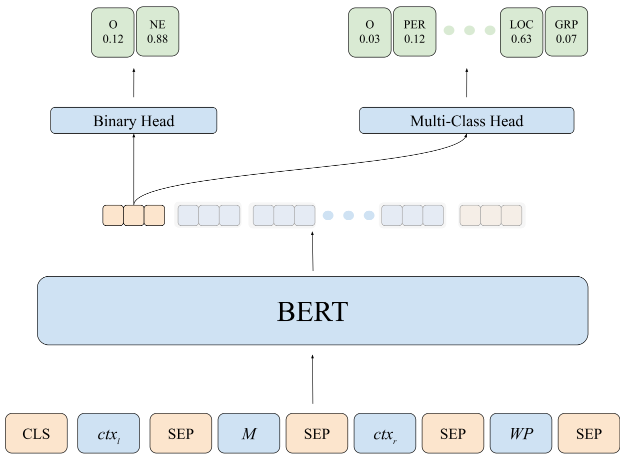


Figure 2: The architecture of the classifier. NE in the binary classifier head's output stands for *Named Entity* and O for *Other*.

5 Experimental Setup

Due to limited resources and time, we only experimented with the Turkish part of the dataset and therefore indexed 749,204 Turkish Wikipedia pages. During retrieval, *OR* operator was used as

Models	Validation			Test		
	Precision	Recall	F1	Precision	Recall	F1
BERT	85.84	85.01	**85.39**	58.02	58.39	57.82
EL_BERT	84.59	79.40	81.74	74.78	59.03	65.47
EL_MultiBERT	**86.18**	77.56	81.60	**80.69**	65.13	71.91
Ensemble$_{Token+EL}$	83.78	85.35	84.47	74.23	59.44	65.66
Ensemble$_{Token+EL\_Multi}$	83.66	**85.51**	84.49	78.86	**66.43**	**72.02**

Table 3: Results of our proposed methods on development and test data

part of the query and the default BM25 was used for ranking. We fine-tuned the BERTurk model[*]. As the optimizer, we used AdamW (Loshchilov and Hutter, 2019) with a fixed learning rate in all our BERT models as 3 x 10^{-5}. We also applied dropout before the feed-forward layer with 0.3 probability. Our batch size was 32, with a maximum sequence length of 128 for the baseline BERT. We extended the maximum sequence length to 256 for EL_MultiBERT and reduced the batch size to 8. All experiments were conducted with seed 22.

6 Results

The official evaluation metric was chosen as macro-average F1-score by the organizers. In Table 3, we present our results with models described above. Our baseline BERT, the one fine-tuned directly for NER, worked quite well on the validation set but performed badly on the test set. This substantial difference in these two sets shows that contextual models like BERT do not perform well in low-context settings. On the other hand, although our proposed entity linking approaches fell slightly behind the baseline model in the development set, they performed significantly better than the baseline on test set. Among the two entity linking methods, the EL_MultiBERT model with two classification heads performed better. While determining the named entity type, training with this auxiliary task even on the same data seems to be useful.

Table 4 illustrates the performance of BERT and EL_MultiBERT models for each named entity class. As seen from the table, our proposed approach improves the performance, especially in complex entities like *Creative Work*, *Group* and *Product*. Two approaches returned similar results only for *Person* and *Location*. Since we do not have the gold standard labels, we cannot perform any detailed analysis of the reason.

[*]https://huggingface.co/dbmdz/bert-base-turkish-uncased

Entity Type	BERT	EL_MultiBERT
Person	70.72	72.43
Location	59.73	57.42
Group	51.49	76.51
Corporation	57.69	73.79
Product	63.13	76.93
Creative Work	44.14	74.40

Table 4: F1-Scores by NER type on Test Set

6.1 Ensemble Approach

Although the majority of the samples in the test set are short texts, there are some longer sentences as well. From the validation set experiments, we already know that when there is enough context, transformer models like BERT perform quite well. Hence, we decided to see if we can use our proposed architecture as a fall back mechanism when there is not enough context available, but otherwise use BERT. We applied an ensemble approach relying on BERT for samples with more than 11 tokens and EL_MultiBERT for shorter ones. So when sentence length is less than 11, we depended on our entity linking architecture and Wikipedia for retrieving and using additional useful context about the input. As shown in Table 3, the ensemble of BERT and EL_MultiBERT achieved 72.02% on the test set, and this system ranked as third in the Turkish track.

7 Conclusion

In this work, we proposed a language-independent method for the Complex NER task in low-context settings. Our results showed that utilizing the use of entity linking for NER provides a significant improvement over state-of-the-art transformer models when there is not enough context. Yet, since our resources are limited, we experimented with only the Turkish part. In future work, we will extend this approach in a multilingual setting.

References

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

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. *Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries*, page 1677–1681. Association for Computing Machinery, New York, NY, USA.

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

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512, Online. Association for Computational Linguistics.

Stefan Schweter. 2020. Berturk - bert models for turkish.

Ikuya Yamada, Hideaki Takeda, and Yoshiyasu Takefuji. 2015. Enhancing named entity recognition in Twitter messages using entity linking. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 136–140, Beijing, China. Association for Computational Linguistics.

Qtrade AI at SemEval-2022 Task 11: An Unified Framework for Multilingual NER Task

Weichao Gan[*†], **Yuanping Lin**[*†], **Guangbo Yu**[*†], **Guimin Chen** [†] and **Qian Ye** [§]
[†]Qtrade AI Lab, Shenzhen China
[§]Guangdong Power Grid Corporation, Guangdong China
{waiciugam, yuenpinglynn, ygbusc}@gmail.com
chenguimin@foxmail.com, eason_sysuye@163.com

Abstract

This paper describes our system, which placed third in the Multilingual Track (subtask 11), fourth in the Code-Mixed Track (subtask 12), and seventh in the Chinese Track (subtask 9) in the SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition. Our system's key contributions are as follows: 1) For multilingual NER tasks, we offer an unified framework with which one can easily execute single-language or multilingual NER tasks, 2) for low-resource code-mixed NER task, one can easily enhance his or her dataset through implementing several simple data augmentation methods and 3) for Chinese tasks, we propose a model that can capture Chinese lexical semantic, lexical border, and lexical graph structural information. Finally, our system achieves macro-f1 scores of 77.66, 84.35, and 74.00 on subtasks 11, 12, and 9, respectively, during the testing phase.

1 Introduction

SemEval 2022 Task 11: MultiCoNER Multilingual Complex Named Entity Recognition(Malmasi et al., 2022b) focuses on extracting semantically ambiguous complex named entities(Meng et al., 2021) in short, low-context and code-mixed(Fetahu et al., 2021) scenarios. The domain adaptability capacity of the system is in high demand for this shared task, which contains 13 tracks in English, Spanish, Dutch, Russian, Turkish, Korean, Farsi, German, Chinese, Hindi, Bangla, multi-language, and code-mixed. Among them, multi-language, and code-mixed tracks are in the mixed language, while the other tracks are in the single language. The difference between multi-language track and code-mixed track is that the corpus of multi-language track is multilingual, but the words in each sentence are in a single language, while the corpus of code-mixed track is a corpus in which the words or phrases in each sentence may be from different languages, for example the sentence I Liebe Sie (I love you) consists of English, French and German. Furthermore, the datasets offered by the organizers , which may mainly from Wikipedia, web questions and user queries, comprise data from 11 languages, with each language containing around 15,000 training samples and 800 development (dev) samples(Malmasi et al., 2022a). The corpus contains a total of six entity categories, namely location (LOC), person (PER), product (PROD), group (GRP), corporation (CORP) and creative works (CW). The main contributions of our system are as follows: 1) we propose a unified framework for multilingual NER tasks, using which one can easily perform monolingual or multilingual NER tasks; 2) for low-resource code-mixed NER tasks, we provide several simple and effective data augmentation methods to easily increase the amount of data; 3) for Chinese tasks, we propose a model that captures Chinese lexical semantics, lexical boundaries and lexical graph structure information in a model.

[*]These authors contributed equally to this work.

2 Related work

Named Entity Recognition (NER)—a classic and fundamental task that aims to extract named entities from a sentence—plays an important role in a variety of downstream tasks in the field of NLP, including relation extraction (Zhong and Chen, 2021), knowledge graph construction (Bosselut et al., 2019), question answering (Diefenbach et al., 2018) and so on. For a long time, the development of NER was slow, especially before the rise of neural networks, and NER mostly used statistical machine learning methods like HMM(Morwal et al., 2012) and CRF(Konkol and Konopík, 2013). Although these methods were effective at the time, they were still stretched for complex scenarios. After the rise of neural networks, especially structured networks such as RNNs(Sherstinsky, 2020) and CNNs(LeCun et al., 1998), NER has been greatly developed, and the accuracy has been greatly improved compared with traditional statistical learning methods. There are also methods that combine neural networks with traditional statistical learning methods, such as BiLSTM+CRF(Chen et al., 2017). Although these methods take NER to another higher level, they also have certain problems. For example, RNNs cannot model bi-directionally context, and then there is the inability of RNNs to capture long-term dependencies, and though CNNs can model bi-directionally, but due to the size of convolutional kernels, they can only model local contextual information at the same time and cannot capture global contextual information. Recently, self-training pre-training models on large-scale corpus such as BERT(Kenton and Toutanova) and its variant versions have greatly improved the accuracy of NER tasks and effectively solved the problem that RNNs cannot capture long-range dependencies as well as dynamic contexts. From statistical models like HMM and CRF to deep learning models like CNN, RNN, and transformer-based pre-trained models, the accuracy of NER tasks is increasing and has reached commercial levels in many scenarios(Yadav and Bethard, 2018). Despite its remarkable progress, NER still faces some challenges(Ma et al., 2020), such as the problem of discontinued and nested named entities(Yan et al., 2021), the challenge of cross-domain and cross-lingual transfer learning(Mueller et al., 2020), the lack of lexicon information when using sub-character tokenization strategy and word boundary information of Chinese NER tasks(Zhang and Yang, 2018), and the ambiguity of named entities(Meng et al., 2021) under different semantic circumstance.

3 System overview

Figure 1 depicts our system's overall architecture and technological process. As we can see, the unified framework allows us to complete all of the subtasks.

3.1 Backbone encoder

We selected XLM-RoBERT-large (Conneau et al., 2020) as our backbone encoder to establish framework unity and make full use of data from different tracks to realize language transfer. XLM-RoBERTa is a multilingual version of RoBERTa that has been pre-trained on 2.5TB of data in 100 languages. The large version has 24 transformer layers, 16 self-attention heads per layer, and a hidden size of 1024.

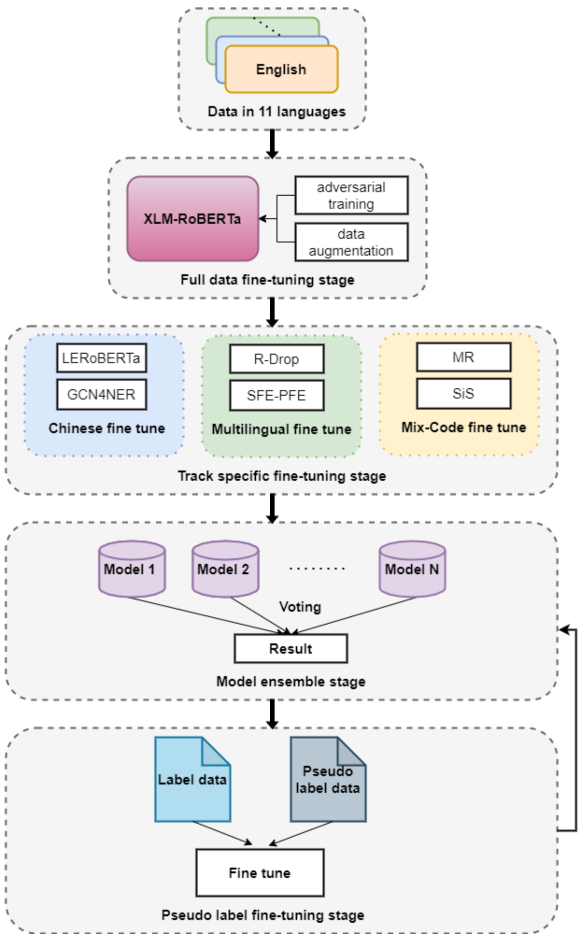


Figure 1: The architecture and procedures of our system

3.2 Training procedures

We take multiple steps to fine tune our system, including full data fine-tuning stage, track specific fine-tuning stage, model ensemble stage and pseudo label fine-tuning stage.

3.2.1 Full data fine-tuning stage

As the first step of our system, we fine tune our model with data from all languages provided by official. At this stage, we hope that the model can learn the distribution of datasets and capture the named entity information from the total data, which will aid our model in transferring to those data on specific tracks. To improve the model's cognitive ability and stability, we implemented the following set of training skills.

Data Augmentation: We did not simply feed the data into our model, but enhanced data firstly in a simple way—concatenating sentences randomly, including **bisent-uni**—concatenating two sentences from the same language randomly into a new sentence, **bisent-mix**—concatenating two sentences from different language randomly into a new sentence, **mulsent-uni**—concatenating several sentences from the same language randomly into a new sentence with a length of no more than 512 and **mulsent-mix**—concatenating several sentences from different language randomly into a new sentence with a length of no more than 512.

Adversarial Training: Considering that the deep neural networks are vulnerable to adversarial examples, we adopt FGM(Miyato et al., 2017), PGD(Mądry et al., 2017) and FreeLB(Zhu et al., 2019) adversarial training techniques to keep our model tolerant of adversarial examples and more robust. These methods create adversarial examples by adding a small perturbation to input that maximizes the loss. The formulation of adversarial training can be drew below,

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D}\left[\max_{\delta\in S} L(\theta, x+\delta, y)\right] \qquad (1)$$

where δ is a small perturbation. The inner maximization can be solved by projected gradient ascent and the outer minimization can be solved by gradient descent. PGD and FreeLB are the improved version of FGM, both of which are devoted to find a more reasonable perturbation.

3.2.2 Track specific fine-tuning stage

At this stage, we use different methods to fine tune according to the specific track.

Track 9 - Chinese (ZH): Track 9 is devoted to the recognition of Chinese named entities. There are significant differences between Chinese and Indo-European languages like English, German, and Spanish. For example, there is no word boundary in the Chinese lexicon, however Chinese word semantic and boundary information is useful for NER. Therefore, after the fine-tuning at the first stage, we utilize some techniques to improve our model's ability to understand Chinese. In particular, we incorporate LEBERT(Liu et al., 2021) and CoGraph4NER(Sui et al., 2019) into our framework to improve our model's understanding of Chinese lexicon boundaries and semantics. It is worth noting that, we did not directly implement these methods; instead, we drew lessons from the ideas presented in these two papers and appropriately transformed the structures mentioned in these two papers to apply to our system.

LERoBERTa: LERoBERTa is our modified version of LEBERT that integrates lexicon information into the specific layers of the pre-training model to obtain word-related information via a structure called Lexicon Adapter. The specific structure of Lexicon Adapter is shown in Figure 2,

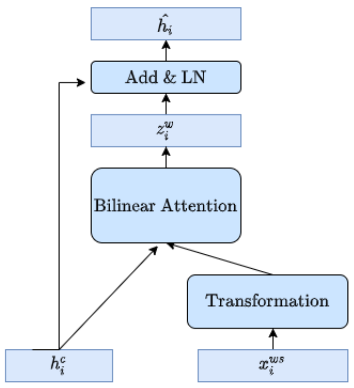


Figure 2: Structure of Lexicon Adapter. This structure pays bilinear attention to characters and vocabulary at the same time, weights vocabulary features into vectors, and then adds them to the input character level vector, and then performs layer normalization.

Because the original LEBERT paper is applied to BERT, it is not consistent with our framework. In order to apply it to our framework, we converted it to LERoBERTa, which differs slightly from LEBERT.

LERoBERTa-GCN4NER: LERoBERTa does not fully utilize the graph relation of containing, transition, and lattice between words and characters because it only integrates information from the bottom encoding layer. An assumption is that the characters in the sentence can capture the boundaries and semantic information of self-matched lexical words using the containing graph (C-graph), the transition graph (T-graph) can assist the character in capturing the semantic information of the nearest contextual lexical words implicitly, and the lattice graph (L-graph) can capture some information of self-matched lexical words explicitly. So, we make some improvements to CoGraph4NER, called GCN4NER, to make full use of the graph relation and thus improve the model's ability to capture Chinese lexicons. GCN4NER is a modified version of CoGraph4NER that replaces the LSTM encoder with LERoBERTa to conform to our algorithm framework and replaces the GAT module with GCN to improve calculation speed. The overall system of LERoBERTa-GCN4NER is shown in Figure 3,

Track 12 - Multilingual: Track 12 focuses on multilingual named entity recognition and the difficulty with this track is that it contains multiple languages, each with its own syntax. So the model

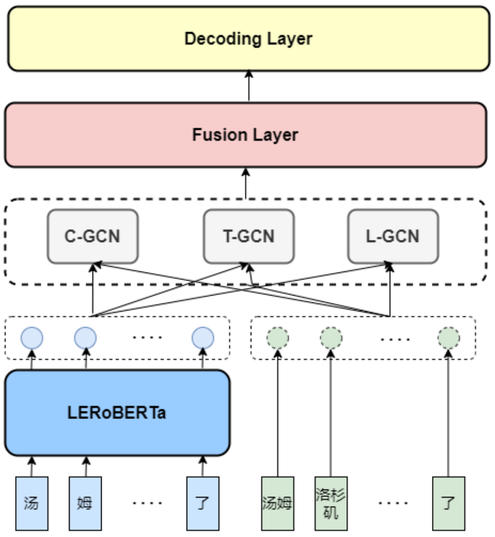


Figure 3: Main architecture of LERoBERTa-GCN4NER. Characters of "Tom left Los Angeles Airport" are encoded by LERoberta and then the hidden vectors and vectors of lexicons are inputted into the GCN module, the outputs of which will then be fused by weighted summation in fusion layer. Finally, the decoding layer assigns a label to each character.

must learn multiple language characteristics concurrently. It is difficult for participants to analyze the model's bad cases because they are not familiar to every language. At this stage, we not only use R-Drop(Wu et al., 2021) but also utilize shared feature extractors and private feature extractors(Chen et al., 2019), considering the fact that there are differences in grammar and common characteristics in semantics of texts in different languages.

R-Drop: Since deep neural networks are very prone to overfitting, the Dropout method randomly discards some neurons in each layer to avoid overfitting during training. Based on this idea, researchers improve dropout method, called R-Drop. Given a training sample $D = \{x_i, y_i\}, i = 1, \cdots, n$, for each sample , it goes through the forward feedback of two different sub-networks to get two predicted probabilities P_1 and P_2. Although the two sub-networks come from the same model, they are not exactly the same because Dropout randomly discards some neurons, and P_1 is not equal to P_2.

Shared Feature Extractor and Private Feature Extractor: We leveraged the ideas of the paper(Chen et al., 2019) and modified the model in combination with datasets offered by the official. With this method, our model can learn general and specific characteristics between different language which may improve model's comprehension.

The shared feature extractor consists of a shared feature learner and a language discriminator. The shared feature learner extracts general features from different languages, and then inputs them into the language discriminator to judge which language these features belongs to. When the language discriminator is unable to determine which language the current text belongs to, it signifies that the shared feature learner has learnt the commonalities between the languages, hence fulfilling the goal of perplexing the language discriminator. There are differences in grammar and meaning of texts in different languages, so the model of this scheme also designs a private feature extractor. The private feature extractor performs feature extraction on the word vector output by backbone encoder, and then outputs it to the multilayer perceptron (MLP) of each language. The MLP of each language extracts the exclusive language features in the text, and then splices the output of each language.

Track 13 - Code-Mixed: Track 13 is a track dedicated to code-mixed data. The prevalence of code-mixed text is fast increasing on social media platforms such as Twitter. The code-mixed task is difficult because it introduces a large number of unseen constructions as a result of merging the lexicon and syntax of two or more languages, and the available data is insufficient in comparison to the other sub-tasks. Therefore, after fine-tuning at the first state, we utilized data augmentation(Dai and Adel, 2020) to supplement the training data in this task. When the labeled datasets are insufficient, data augmentation is a frequently used strategy for enhancing generalization. However, the NER task is concerned with sequence labeling at the token level, and the majority of data augmentation methods at the sentence level may compromise label integrity. We used **Mention replacement (MR)** and **Shuffle within segments (SiS)** techniques to resolve the issue. MR is a data augmentation approach that replaces the position of entities in the original sentence with other entities of the same category while keeping the non-entity part of the sentence unchanged. And SiS is a data enhancement method that keeps the order of words of entities in the original sentence unchanged and disrupts the order of words in the non-entity part.

3.3 Model ensemble stage

Model Ensemble is a method for integrating multiple trained models in order to improve the system's generalization ability in test datasets(Allen-Zhu and Li, 2020). Model ensemble can be achieved in a variety of ways, including voting, averaging, stacking, and confidence blending. Voting is one of them, and it is a simple yet effective method, so we use it in our system. We trained several models for each track based on different hyper parameters and then predicted test datasets to get corresponding results. We then voted for the final result based on these intermediate results.

3.4 Pseudo label fine-tuning stage

Pseudo labelling is a type of semi-supervised learning in which the model trained by labelled data is used to generate pseudo labels for an unlabelled dataset(Lee et al., 2013). We then put both the original labelled dataset and a portion of the unlabelled dataset with pseudo labels into our models for final training during the pseudo label fine-tuning stage. Voting is how we generate pseudo labels. We choose sentences that can be consistently predicted by all models generated in the model ensemble stage as training samples with reasonable pseudo-labels.

4 Experimental setup

In this session, we are going to describe the implementation details of our system.

Dataset and word embeddings: Train and dev datasets we used are provided by official and we don't use test dataset before testing phase because it was not available. Besides, experiments at model ensemble stage and pseudo label fine-tuning stage are carried on test dataset.Since LERoBERTa and GCN4NER techniques require word embedding and considering the lack of embedding of relevant words in the pre-trained model XLM-RoBERTa, we use word embedding sgns.merge.word [†] trained by skip-gram.

Processing and hyper parameters: Limited by the length of the paper, we will only briefly introduce our main processing and part of hyper parameters, with which our system can reach the best result. Learning rate was set to 1×10^{-6}, warming up proportion to 0.06, drop out rate to 0.2, batch size to 32, epoch to 30, random seed to 42 and the number of voting models to 7. Augmentation method for full data fine-tuning stage was **mulsent-uni**, decoder strategy was softmax and adversarial training method was PGD with $\epsilon = 1.0, \alpha = 0.1, K = 3$. For more details, please check the Appendix A. Through out our system we almost only use softmax as our decoder strategy because we have found CRF has no effect on improving the scores compared to softmax but consumes more computing resources and storage space. So, we drop CRF strategy. One plausible explanation for this is that pretrained models have already caught the relations between tokens which may not be well captured by non-pretrained models. We selected the best epoch and the best hyper parameters using performance (measured in terms of macro-f1 score) on corresponding dev dataset.

[†]https://drive.google.com/file/d/1Zh9ZCEu8_eSQ-qkYVQufQDNKPC4mtEKR/view.

5 Results

In this section, we will report our main experiment and provide an analysis of the results. Unless otherwise specified, the hyper parameters in our experiments are configured in accordance with Section 3.

Full data fine-tuning stage: Table 1 displays the results of experiments performed with the following parameters: FGM with $\epsilon = 0.8$, PGD with $\epsilon = 1, \alpha = 0.1, K = 3$ and FreeLB with $adv\_lr = 0.3, mag = 0.05, K = 3$. These hyper parameters are the best that we have found.

As we can see from Table 1, three tracks have an improvement in the mulsent-uni data augmentation method where Chinese(ZH) and Multilingual tracks improve nearly 0.5 percent and Mix-Code tack increases 1.2 percent to 77.19. What's more, all adversarial training techniques we employed in our system make a great improvement in Chinese(ZH) and Multilingual track, but nearly have no effect on Mix-Code track. However, when we adopt mulsent-uni and PGD simultaneously, scores are surprisingly high, 88.46 for ZH-Chinese, 86.93 for Multilingual and 78.21 for Mix-Code and this checkpoint is our best checkpoint at full data fine-tuning stage which will continue to be used at track specific fine-tuning stage.

Chinese (ZH) track fine-tuning: As for Chinese (ZH) track, we employ LERoBERTa, GCN4NER and LERoBERTa-GCN4NER for fine-tuning. Table 2 demonstrates that, LEBERT gets the lowest score without our full data fine-tuning stage and our modified version, LERoBERTa,

Table 1: Marco-f1 scores(%) on dev dataset under different methods at full data fine-tuning stage

methods		Chinese(ZH)	Multilingual	Mix-Code
\	\	86.31	85.82	75.93
data augmentation	bisent-uni	86.20	86.02	77.01
	mulsent-uni	86.06	86.06	77.19
	bisent-mix	85.27	86.01	74.67
	mulsent-mix	86.50	85.91	74.43
adversarial training	FGM	86.85	86.85	75.54
	PGD	87.37	86.33	75.96
	FreeLB	86.81	86.14	75.88
methods ensemble	mulsent-uni+PGD	**88.02**	**86.93**	**78.21**

GCN4NER and LERoBERTa-GCN4NER have a positive effect on improving macro-f1 score. We can prove our hypothesis that LERoBERTa only integrates the information of words from the bottom encoding layer, and that it does not make full use of the graph relation of containing, transition, and lattice between words and characters, by observing the differences in results between our modified models. As a result, LERoBERTa just climbs by 0.18 percent. Similar to GCN4NER, it may just consider character graph relationships while ignoring lexical semantic information. So, GCN4NER increases by 0.32 percent. When these two approaches are combined, we discover that LERoBERTa-GCN4NER improves by 1%, giving it the highest score on the dev dataset in our system.

Multilingual track fine-tuning: This experiment continually use the best checkpoint from full data fine-tuning stage . We try R-Drop technique and private feature extractor and shared feature extractor (PFE-SFE) method to improve the performance of our model on this track. In the R-Drop experiment, we use the average method for the KL divergence, and set coefficient parameter α to 0.1 and it obtains a marco-f1 score of 87.15 on the dev dataset which means that this method has a good effect on multilingual tasks. What‘s more, we adopt PFE-SFE method and the marco-f1 score achieve 87.21 in our experiment.

It can be seen from Table 3 that both R-Drop and PFE-SFE have improved macro f1 score compared to the baseline model, R-Drop is 0.22% higher than the baseline, and PFE-SFE is 0.28% higher than the baseline. It can be seen that both methods have certain effects. When we combine R-Drop with PFE-SFE, the macro-f1 value reaches 87.42, which is 0.49% higher than the baseline.

Mix-Code track fine-tuning: As for Mix-Code track, we used mention replacement (MR) and shuffle within segments (SiS) techniques to resolve the insufficiency issue of mix-code data and increase the generalization ability of our model on this track. We not only utilize the parameter setting at full data fine-tuning stage, but also employ the best checkpoint at full data fine-tuning stage.

As shown in Table 4, all of the data augmentation techniques we utilized increase macro-f1 over the baseline value when no augmentation is applied. The MR technique enables the model to get a more accurate representation of entity knowledge and entity boundary information based on external knowledge. Additionally, by changing the order of the sequences, the SiS technique enables the model to learn a more robust position embedding.And the result in bold is our best score on the leaderboard before testing phase.

Model ensemble stage: At this stage, test data is available and we choose the top-7 models based on marco-f1 score under 7 sets of different parameters to vote on the final results. For details, please check the Appendix B. The second row in Table 5 shows the best results predicted singly by our chosen models. As we can see from Table 5, model ensemble has a positive effect. Before the number of models increases to 7, scores increase as the number of models increases but the growth rate slows down which means more models will have little effect even negative effect.

Pseudo labeling fine-tuning stage: After model ensemble stage, we select 7 models with the highest f1 values on the development set to make predictions on the test set, and choose the results that all

Table 2: Marco-f1 scores(%) on dev dataset under different models at Chinese (ZH) track fine-tuning stage

Models	XLM-RoBERTa	LEBERT	LERoBERTa	GCN4NER	LERoBERTa-GCN4NER
Chinese(ZH)	88.46	86.02	88.62	88.78	**89.4**

Table 3: Marco-f1 scores(%) on on dev dataset under different methods at multilingual track fine-tuning stage

methods	\	R-Drop	PFE-SFE	R-Drop+PFE-SFE
Multilingual	86.93	87.15	87.21	**87.42**

Table 4: Marco-F1 scores(%) on dev dataset under different data augmentation strategies at code-mixed fine-tuning stage

Data Augmentation Methods	Mix-Code
\	79.8
Mention replacement (MR)	82.0
Shuffle within segments (SiS)	81.1
MR + SiS	**82.2**

Table 5: Macro-F1 scores(%) on test dataset under different number of models at model ensemble stage(N denotes the number of models)

N	Chinese(ZH)	Multilingual	Mix-Code
1	67.71	73.18	80.32
3	69.42	73.86	82.43
5	**69.73**	**74.32**	**82.75**
7	68.20	73.67	81.53

seven models predict consistently as the pseudo-labeled data and then we put them together with train dataset for fine-tuning at this stage. We continually use the best checkpoint of LERoBERTa-GCN4NER to fine tune on the Chinese(ZH) track and same as the Multilingual track and Code-mixed track. In addition, the hyperparameters remain the same as in the previous phase. The results we obtained are as shown in Table 6. Obvious as the effect is, macro-f1 scores increases 5.22 points on the Chinese(ZH) track, 4.04 points on the Multilingual track and 3.59 points on the Code-mixed track.

Table 6: Macro-F1 scores(%) on test dataset at pseudo labeling fine-tuning stage

pseudo	Chinese(ZH)	Multilingual	Mix-Code
✗	67.71	72.87	80.32
✓	**72.93**	**76.91**	**83.91**

Next, same as model ensemble stage we choose 7 sets of different parameters to train on the pseudo label dataset and get 7 sets of checkpoints. For more details about these models, please check the Appendix A. The results are shown in Table 7. As we can see, the results are similar to those at model ensemble stage. Macro-f1 scores also have an improvement but the increments are less than before.The value in bold is our result in the final ranking.

Table 7: Macro-F1 scores(%) on test dataset under different number of models at pseudo labeling fine-tuning stage (N denotes the number of models)

N	Chinese(ZH)	Multilingual	Mix-Code
1	72.93	76.91	83.91
3	73.60	77.37	84.29
5	**74.00**	**77.66**	**84.35**
7	73.25	77.32	84.02

6 Conclusion

In this paper, we have introduce our system step by step which can be regarded as an universal framework for multilingual NER task. Besides, we proposed a graph-based model to make up for the lack of understanding capacity of Chinese lexicon. Adequate ablation experiments shows that our methods work for this task. In future efforts, we plan to further improve our system to better handle polysemous scenarios.

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Tao Chen, Ruifeng Xu, Yulan He, and Xuan Wang. 2017. Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. *Expert Systems with Applications*, 72:221–230.

Xilun Chen, Ahmed Hassan, Hany Hassan, Wei Wang, and Claire Cardie. 2019. Multi-source cross-lingual model transfer: Learning what to share. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3098–3112.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867.

Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information systems*, 55(3):529–569.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Universal Language Model Fine-tuning for Text Classification*, page 278.

Michal Konkol and Miloslav Konopík. 2013. Crf-based czech named entity recognizer and consolidation of czech ner research. In *International conference on text, speech and dialogue*, pages 153–160. Springer.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced chinese sequence labeling using bert adapter. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5847–5858.

Ruotian Ma, Minlong Peng, Qi Zhang, Zhongyu Wei, and Xuan-Jing Huang. 2020. Simplify the usage of lexicon in chinese ner. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5951–5960.

Aleksander Mądry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *stat*, 1050:9.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. *stat*, 1050:6.

Sudha Morwal, Nusrat Jahan, and Deepti Chopra. 2012. Named entity recognition using hidden markov model (hmm). *International Journal on Natural Language Computing (IJNLC) Vol*, 1.

David Mueller, Nicholas Andrews, and Mark Dredze. 2020. Sources of transfer in multilingual named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8093–8104.

Alex Sherstinsky. 2020. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2019. Leverage lexical knowledge for chinese named entity recognition via collaborative graph network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3830–3840.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various ner subtasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5808–5822.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for language understanding.

Appendix

A Hyper parameters setting

In these tables, T, C and L under graph parameter denote T-graph, C-graph and L-graph respectively which means we use one or all of these three graph in our LERoBERTa-GCN4NER model.

Table 8: Scope of each hyper parameters that we have tried in our experiments

parameters	scope
learning rate	5×10^{-6}, $\{1, 3, 5\} \times 10^{-5}$
warming up rate	{ 0.06, 0.1 }
seed	{ 42, 100, 200, 2022 }
batch size	{ 8, 16, 32 }
epoch	{ 30, 50 }
FGM(ϵ)	{ 0.1, 0.3, 0.5, 0.8, 1 }
PGD(ϵ)	{ 0.1, 0.3, 0.5, 0.8, 1 }
PGD(α)	{ 0.3, 1 }
PGD(K)	{ 1, 2, 3, 5 }
FreeLB(adv_lr)	{ $\{1, 3, 5\} \times 10^{-5}$ }
FreeLB(mag)	{ 0.05, 0.1, 0.5 }
FreeLB(K)	{ 1, 2, 3, 5 }
R-Drop(α)	{ 0.01, 0.05, 0.2, 0.5 }
GCN4NER(graph)	{ T+C+L, T, C, L }

B Macro-f1 scores($\%$) predicted singly by different models under different parameters

Table 9: Macro-f1 scores(%) predicted by every single model under different parameters on Chinese(ZH) test dataset at model ensemble stage

id	macro-f1	learning rate	warming up rate	batch sieze	PGD	graph
1	**67.71**	1×10^{-5}	0.06	32	+	T+C+L
2	67.56	5×10^{-6}	0.1	32	+	T+C+L
3	67.68	1×10^{-5}	0.06	16	+	T+C+L
4	67.43	1×10^{-5}	0.06	32	-	T+C+L
5	67.35	1×10^{-5}	0.06	32	+	T
6	67.32	1×10^{-5}	0.06	32	+	C
7	67.21	1×10^{-5}	0.06	32	+	L

Table 10: Macro-f1 scores(%) predicted by every single model under different parameters on Chinese(ZH) test dataset at pseudo labeling fine-tuning stage

id	macro-f1	learning rate	warming up rate	batch sieze	PGD	graph
1	72.56	1×10^{-5}	0.06	32	+	T+C+L
2	72.89	5×10^{-6}	0.1	32	+	T+C+L
3	**72.93**	1×10^{-5}	0.06	16	+	T+C+L
4	72.53	1×10^{-5}	0.06	32	-	T+C+L
5	72.58	1×10^{-5}	0.06	32	+	T
6	72.42	1×10^{-5}	0.06	32	+	C
7	72.33	1×10^{-5}	0.06	32	+	L

Table 11: Macro-f1 scores(%) predicted by every single model under different parameters on multilingual test dataset at model ensemble stage

id	macro-f1	learning rate	warming up rate	batch sieze	PFE-SFE	R-Drop(α)
1	72.32	5×10^{-6}	0.06	16	+	0.05
2	**73.18**	5×10^{-6}	0.06	16	+	0.1
3	72.86	5×10^{-6}	0.06	16	+	0.2
4	72.64	1×10^{-5}	0.06	16	+	0.1
5	72.12	5×10^{-6}	0.06	16	-	0.1
6	73.01	5×10^{-6}	0.1	16	+	0.1
7	73.09	5×10^{-6}	0.06	32	+	0.1

Table 12: Macro-f1 scores(%) predicted by every single model under different parameters on multilingual test dataset at pseudo labeling fine-tuning stage

id	macro-f1	learning rate	warming up rate	batch sieze	PFE-SFE	R-Drop(α)
1	76.43	5×10^{-6}	0.06	16	+	0.05
2	**76.91**	5×10^{-6}	0.06	16	+	0.1
3	76.74	5×10^{-6}	0.06	16	+	0.2
4	76.18	1×10^{-5}	0.06	16	+	0.1
5	75.96	5×10^{-6}	0.06	16	-	0.1
6	76.63	5×10^{-6}	0.1	16	+	0.1
7	76.78	5×10^{-6}	0.06	32	+	0.1

Table 13: Macro-f1 scores(%) predicted by every single model under different parameters on mix-code test dataset at model ensemble stage

id	macro-f1	learning rate	warming up rate	batch sieze	data augment	decoder
1	**80.32**	1×10^{-6}	0.06	32	MR + SiS	crf
2	80.21	1×10^{-6}	0.06	32	MR + SiS	crf
3	79.84	1×10^{-5}	0.06	32	MR + SiS	softmax
4	80.09	1×10^{-6}	0.1	32	MR + SiS	softmax
5	79.94	1×10^{-6}	0.06	32	MR	crf
6	80.11	5×10^{-6}	0.06	16	MR	crf
7	79.75	1×10^{-6}	0.06	32	MR	softmax

Table 14: Macro-f1 scores(%) predicted by every single model under different parameters on mix-code test dataset at pseudo labeling fine-tuning stage

id	macro-f1	learning rate	warming up rate	batch sieze	data augment	decoder
1	**83.91**	1×10^{-6}	0.06	32	MR + SiS	crf
2	83.38	1×10^{-6}	0.06	32	MR + SiS	crf
3	83.40	1×10^{-5}	0.06	32	MR + SiS	softmax
4	83.55	1×10^{-6}	0.1	32	MR + SiS	softmax
5	83.00	1×10^{-6}	0.06	32	MR	crf
6	83.70	5×10^{-6}	0.06	16	MR	crf
7	83.50	1×10^{-6}	0.06	32	MR	softmax

PAI at SemEval-2022 Task 11: Name Entity Recognition with Contextualized Entity Representations and Robust Loss Functions

Long Ma[*], **Xiaorong Jian**[*],**Xuan Li**
Pingan Life Insurance of China, Ltd
{MALONG633,JIANXIAORONG974,LIXUAN208}@pingan.com.cn

Abstract

This paper describes our system used in the SemEval-2022 Task 11 Multilingual Complex Named Entity Recognition, achieving 3rd for track 1 on the leaderboard. We propose Dictionary-fused BERT, a flexible approach for entity dictionaries integration. The main ideas of our systems are: 1) integrating external knowledge (an entity dictionary) into pre-trained models to obtain contextualized word and entity representations 2) designing a robust loss function leveraging a logit matrix 3) adding an auxiliary task, which is an on-top binary classification to decide whether the token is a mention word or not, makes the main task easier to learn. It is worth noting that our system achieves an F1 of 0.914 in the post-evaluation stage by updating the entity dictionary to the one of Meng et al. (2021), which is higher than the score of 1st on the leaderboard of the evaluation stage.

1 Introduction

Name entity recognition (NER) is a fundamental task in natural language processing. Processing complex and ambiguous Named Entities (NEs) and in low-context situations is a challenging NLP task in practical and open-domain settings, which was recently outlined by Meng et al. (2021). Other work has extended this to multilingual and code-mixed settings (Fetahu et al., 2021) since code-mixed queries, with entities in a different language than the rest of the query, pose a particular challenge in domains like e-commerce (e.g. queries containing movie or product names).

SemEval-2022 task 11 Multilingual Complex Named Entity Recognition (Malmasi et al., 2022b) focuses on dealing with the challenges above: detecting complex entities in short, low-context, and code-mixed settings.

For this task, we propose Dictionary-fused BERT to integrate external dictionaries into the NER model, and it is compatible with emerging entities and user-defined entities, without retraining the model.

2 Related Work

Named Entity Recognition (NER) is a core task in knowledge extraction and is important to various downstream applications such as question answering and dialogue systems. NER is the task of detecting mentions of real-world entities from text and recognizing their types (e.g., locations, persons).

However, the NER task is facing many challenges outlined by Meng et al. (2021), such as short texts like search queries (Wang et al., 2014), emerging entities (Craswell et al., 2020), long-tail entities, complex entities, and entities in code-mixed queries (Fetahu et al., 2021). For example, complex NEs, like the titles of creative works (movie/book/song/software names) are not simple nouns and are harder to recognize. The neural models driven by memorization perform well on "easy" entities (person names) but fail to recognize complex/unseen entities when entities overlap less between train/test sets. A lot of works have been proposed to address the challenges above.

Contextualized Word and Entity Representations There are some works focusing on obtaining good contextual word and entity representations such as KnowBERT (He et al., 2019) and LUKE (Yamada et al., 2020). KnowBERT (He et al., 2019) incorporates knowledge bases into BERT (Devlin et al., 2018) using Knowledge attention and recontextualization, which explores the joint learning of entities and relations. LUKE (Yamada et al., 2020) employs RoBERTa (Delobelle et al., 2020) as the base pre-trained model, trained on a large number of entity-annotated corpora retrieved

[*] The first two authors contributed equally.

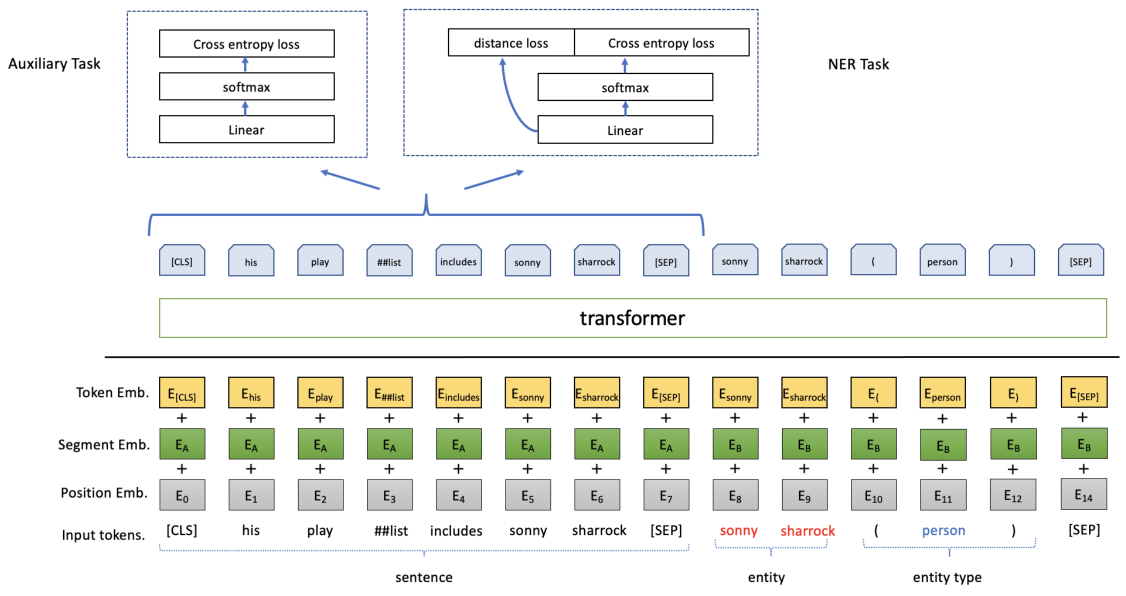


Figure 1: The overall architecture of our proposed system

from Wikipedia. LUKE treats both words and entities as separate tokens and computes the intermediate and output representations of all tokens through the Transformer. Since entities are also used as tokens, LUKE can model the relationship between entities.

Robust Loss Functions Other works like Generalized Cross-Entropy (Zhang and Sabuncu, 2018) and In-trust loss functions (Huang et al., 2021) focus on how to design a robust loss function to solve the NER problem under label noise, which is consistent with complex and ambiguous NER scenarios. Generalized Cross-Entropy (Zhang and Sabuncu, 2018) is actually a new evolutionary form of MAE and CCE, and can be easily applied with the DNN algorithm while yielding good performance in a lot of noisy label scenarios. In-trust loss functions (Huang et al., 2021) combines a CRF loss with a robust Distrust Cross-Entropy term and can effectively alleviate overfitting. What's more, it has been demonstrated that leveraging a logit matrix is an effective way to distinguish noisy samples from difficult samples.

Our proposed model combines the Contextualized-Entity-Representations method and the Robust-Loss-Functions method. Firstly our model incorporates the entity dictionary proposed by LUKE (Yamada et al., 2020) into the BERT-Whole-Word-Masking model and treats entities as separate tokens. Then we design a robust loss function that boosts cross-entropy loss with KL divergence over the logit matrix.

3 Data

We leverage LUKE entity dictionary (Yamada et al., 2020) since our model needs external knowledge. We train and test on MultiCoNER Dataset (Malmasi et al., 2022a).

MultiCoNER.Dataset for the SemEval-2022 Task 11, containing a training set of size 15300, a dev set of size 800, and a test set of size 217818, consisting of 6 entity types: PERSON (PER for short, names of people), LOCATION (LOC, locations/physical facilities), CORPORATION (CORP, corporations and businesses), GROUPS (GRP, all other groups), PRODUCT (PROD, consumer products), and CREATIVE-WORK (CW, movie/song/book/etc. titles). All data are uncased.

LUKE Entity Dictionary consists of 500k entities retrieved from English Wikipedia data.

4 Methodology

We propose Dictionary-fused BERT, which adopts a multi-layer bidirectional transformer (Vaswani et al., 2017) and has the ability to encode not only words of the sentence but also information of matched entities. We experiment with Transformer-

Preained-model	**dev f1**
BERT$_{base}$	0.854
BERT$_{large}$	0.871
RoBERTa$_{base}$	0.836
RoBERTa$_{large}$	0.877
DistilBERT$_{base}$	0.835
BERT-WWM$_{large}$	**0.883**
LUKE$_{base}$	0.856
LUKE$_{large}$	0.878

Table 1: Dev f1 score of different Transformer-based models.

based models which produce contextualized word representations. The overall architecture is shown in Figure 1, the components are detailed below.

4.1 Contextualized Word and Entity Representation

Entity representations are obtained in two steps: entity matching and contextual encoding.

Dictionary Entity Matching We denote the input sentence as $(w_1, w_2..., w_n)$, where w_i is the i-th word and n is the Length. Full string matching is used to match all entities in the input sentence and we choose the longer one while dealing with overlapping matches. $(m_1, m_2, ..., m_L)$ represents the matched entities where $m_j = (w_i, w_{i+1}.., w_{i+k})$ and L is the number of matched entities.

Contextual Encoding The input tokens consist of two segments: words of the sentence and matched entities including the entity itself and entity type. The two parts are separated by a special token [SEP] and different entities are separated by the token $. Then they are fed together into the encoder to get the contextualized representations as LUKE(Yamada et al., 2020) does.

4.2 On-top Binary Classification Task

On-top binary classifier (adding a binary classifier on the top of the encoder) is an auxiliary task to detect entity mention words of the input sentence and helps increase accuracy. For example, the target output of the auxiliary task is represented as $y_{aux} = (0, 0, 0, 1, 0, 0, 1, 1)$ whereas the NER task's is $y_{ner} = (O, O, O, B-CW, O, O, B-PER, I-PER)$ when the input sentence is "adaptation of the manga series by chiho saito".

4.3 Loss Function

As is shown in the Figure 1, the loss is divided into two parts: the cross-entropy loss of the auxiliary task ($L_{CE-auxiliaty-task}$) and the loss of the NER task that consists of the final cross-entropy loss (L_{CE}) and KL divergence loss over the logit matrix (L_{logit}).

4.3.1 Loss over Logit Matrix

The logit matrix is the output of models in the training process, which goes through the softmax layer and then gets into the final loss function. Huang et al. (2021) has demonstrated the use of logit matrix to distinguish noisy samples from difficult samples to prevent overfitting, which we use here to distinguish tag pairs such as (CW, PROD) and (CORP, GRP) pairs that are easily confused with each other by baseline model. These samples are shown below.

* {they have been used experimentally in [gyrobus]{PROD}{CW}}. In this sample, [gyrobus] is a product (PROD) but is mispredicted as a creativework (CW).

* {cockpit concept, developed by [astion martin racing]{GRP}{CORP}}. [astion martin racing] is a group (GRP) but mispredicted as a corporation (CORP).

We denote the logit matrix as $[Z_{ij}]^{N \times M}$ where N is the length of input tokens and M is the number of tags. The loss over logit matrix can be represented as $L_{logit} = -(KLdiv(z_{CW}, z_{PROD}) + KLdiv(z_{GRP}, z_{CORP}))$ where z_{CW}, z_{PROD}, z_{GRP} and z_{CORP} are column vectors corresponding to tag CW, tag PROD, tag GRP and tag CORP in the logit matrix and $KLdiv$ is the Kullback-Leibler divergence loss function. The negative signal $(-)$ indicates that the larger distance between tag pairs the better.

4.3.2 Forming Total Loss

The total loss can be represented as

$$L_{total} = \alpha \cdot L_{CE-auxiliaty-task} + \beta \cdot (L_{CE} + L_{logit})$$

where α and β are two decoupled hyper parameters, α regulates the recall issue while β aims to flexibly explore the robustness of L_{logit}.

	settings	dev f1
baseline	BERT-WWM	0.883
	BERT-WWM$_{entity}$	0.90
ours	BERT-WWM$_{entity}$+auxiliary task	0.904
	BERT-WWM$_{entity}$+auxiliary task+L_{logit}	**0.913**

Table 2: results of ablation experiments

dictionary settings	dev f1	test f1
LUKE (Yamada et al., 2020)'s	0.913	0.784
GEMNET (Meng et al., 2021)'s	0.945	0.914

Table 3: performance of our model with injecting different entity dictionaries

5 Results

In this section, we verify the advantages of our system. On the one hand, we select the pre-trained BERT-Whole-Word-Masking model (BERT-WMM) (Liu et al., 2020) as our baseline model by comparing five different kinds of Transformer-based models. On the other hand, we analyze the effects of the way injecting an entity dictionary, adding an auxiliary task, and the loss function leveraging the logit matrix.

5.1 Baseline Experiments

The baseline performance of BERT (Devlin et al., 2018), BERT-WMM (Liu et al., 2020), DistilBERT (Sanh et al., 2019), RoBERTa (Delobelle et al., 2020) and LUKE(Yamada et al., 2020) is tested by directly fine-tuning the MultiCoNER data(Malmasi et al., 2022a) on the pre-trained LMs. All models are uncased. The results are shown in Table 1. BERT-WWM outperforms the others by 1 to 5 percentage points, so we choose BERT-WWM as our encoder. It may be due to the whole-word-masking mechanism adopted by BERT-WMM. Most hyperparameters are set as before, especially the learning rate is 2e-5, the batch size is 32, max sequence length is 100, max epochs is 20. All the performance data is on the development set.

5.2 Ablation Experiments

Results of ablation experiments are shown in Table 2. Our model Dictionary-fused BERT is represented as BERT-WWM$_{entity}$ + auxiliary task + L_{logit} and achieves an F1 of 0.913 on dev set when integrating LUKE entity dictionary into the model .

Effect of Integrating Entity Dictionaries As Malmasi et al. (2022b) says the MultiCoNER task aims to recognize entities in open-world settings that entities with tag CW and PROD try to mimic, which reminds us of levering external knowledge to deal with such issues. We treat sentence words and matched entities which are obtained by matching the LUKE entity dictionary and the original sentence as separated tokens like Yamada et al. (2020) does. This experiment is called BERT-WWM$_{entity}$ and improves by 1.7% over the no-entity BERT-WWM baseline.

Effect of Auxiliary Task We also explore Multi-Task learning by conducting the experiment of adding a binary classifier on top of the encoder, which predicts whether the token is a mention word or not. The auxiliary task is intended to increase the overall recall of mentions, which indeed brings a tiny improvement by 0.4% over BERT-WWM$_{entity}$.

Effect of L_{logit} As mentioned in the previous chapter, we propose L_{logit} using the logit matrix to distinguish some entities due to they are easily confused with each other. The total loss function boosting with L_{logit} improves by 0.9% . Finally, our model achieves 0.913 F1 on the dev set.

5.3 Evaluation Results

In the stage of evaluation, in order to fully utilize the training and dev sets, we corporate 10-fold cross-validation training trick and ensemble ten results to obtain the final one. Concretely, we divide the origin training and dev sets into 10 mutually exclusive subsets of the same size. Each time the union of 9 subsets is treated as the training set while the remaining subset is treated as the dev set. We do training and predicting 10 times to get 10 models and 10 results of the test set. The final result is a weighted vote of 10 results, where the weight is the F1 score of each label of each model.

We also compare the performance of our model injecting different entity dictionaries on dev and test sets. We incorporate the LUKE entity dictionary into our model in the evaluation stage. In the post-evaluation stage, by updating LUKE(Yamada et al., 2020) entity dictionary to the one of Meng et al. (2021), our system achieves an F1 of 0.914, which is higher than the score of 1st on the leaderboard of the evaluation stage. Results are shown in Table 3.

6 Conclusion

In this paper, we focused on how to integrate external dictionaries into NER models to deal with NER challenges in open-world settings and design a robust loss function to prevent overfitting. We proposed Dictionary-fused BERT, a flexible and simple approach that includes a contextualized entity representation encoder and a robust loss function leveraging a logit matrix. More importantly, our approach can infuse entity dictionaries into any Transformer-based pre-trained models and is compatible with any emerging entities and user-defined entities without retraining the model.

Although we have already succeeded in injecting entity information into Transformer-based models, entity spans are implicitly informed through a special token $. In the future, we will attempt to explore other ways to explicitly encode entity span information.

References

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 18 million clicked query-document pairs for analyzing search.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. Robbert: a dutch roberta-based language model. *arXiv preprint arXiv:2001.06286*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

B. He, D. Zhou, J. Xiao, X. Jiang, Q. Liu, N. J. Yuan, and T. Xu. 2019. Integrating graph contextualized knowledge into pre-trained language models.

Xiusheng Huang, Yubo Chen, Shun Wu, Jun Zhao, Yuantao Xie, and Weijian Sun. 2021. Named entity recognition via noise aware training mechanism with data filter. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4791–4803, Online. Association for Computational Linguistics.

Chao Liu, Cui Zhu, and Wenjun Zhu. 2020. Chinese named entity recognition based on bert with whole word masking. In *Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence*, pages 311–316.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: a Large-scale Multilingual dataset for Complex Named Entity Recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 Task 11: Multilingual Complex Named Entity Recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

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

Kuansan Wang, Evgeniy Gabrilovich, David Carmel, Bo June Hsu, and Ming Wei Chang. 2014. Erd'14: Entity recognition and disambiguation challenge. *ACM SIGIR forum*.

I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto. 2020. Luke: Deep contextualized entity representations with entity-aware self-attention.

Zhilu Zhang and Mert R. Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels.

SemEval 2022 Task 12: Symlink
Linking Mathematical Symbols to their Descriptions

[1]Viet Dac Lai, [1]Amir Pouran Ben Veyseh, [2]Franck Dernoncourt, [1]Thien Huu Nguyen
[1]Dept. of Computer and Information Science, University of Oregon, Eugene, Oregon, USA
{*vietl,apouranb,thien*}@*cs.uoregon.edu*
[2]Adobe Research, Seattle, Washington, USA
franck.dernoncourt@adobe.com

Abstract

We describe Symlink, a SemEval shared task of extracting mathematical symbols and their descriptions from LaTeX source of scientific documents. This is a new task in SemEval 2022, which attracted 180 individual registrations and 59 final submissions from 7 participant teams. We expect the data developed for this task and the findings reported to be valuable for the scientific knowledge extraction and automated knowledge base construction communities. The data used in this task is publicly accessible at https://github.com/nlp-uoregon/symlink.

1 Introduction

The exponential growth of published articles may exceeds many readers' ability to keep track of the development of their field of interest. Hence, automatic reading comprehension of scientific documents has attracted the attention of researchers across various domains such as Drug Discovery, Knowledge Base Construction, and Natural Language Processing. A crucial aspect of understanding scientific literature is understanding terminologies and formulae because they offer an explicit and precise interface to present the relation between scientific concepts (Schubotz et al., 2018). As such, a reading comprehension machine needs to (i) identify their descriptions and formulae, (ii) segment them into primitive terms and symbols, and (iii) link the associated terms and corresponding symbols.

Working with mathematical formulae is arduous due to two fundamental reasons. First, common text encodings such as ASCII and Unicode do not fully support typing mathematical symbols. As a result, complex mathematical formulae are rarely written using either ASCII or Unicode. Rather, a higher level encoding (or typesetting) is often used to encode the content of scientific documents, in particular LaTeX. Second, most scientific documents are stored in one of two forms: photos or Portable Document Format (PDF). Scientific documents that were published prior to the graphical computer era are printed and now scanned and distributed as photos. Nowadays, scientific documents are often composed in some text editors or word processing software, then exported and shared a PDF file. Unfortunately, analyzing textual information in photo images or PDF files is extremely difficult, and most of the natural language processing tools are not developed to handle this format. As such, to facilitate the understanding of scientific literature, documents should be stored using a universal easy-to-process text-like encoding. In this paper, we use LaTeX as the typesetting to facilitate document analysis. Thanks to recent advances in text processing and image recognition, a LaTeX document can often be restored to some extent from either a photo or a PDF file (Deng et al., 2017).

This paper introduces the Symlink shared task for the extraction of mathematical symbols and their descriptions from English scientific documents using their LaTeX source. Figure 1 visualizes an example of the task. This paper also presents an analysis of the results of participant systems on the task. The rest of the paper is organized as follows. Section 2 presents related work in extracting formulae and their related information from scientific documents. Section 3 describes the subtasks of this Symlink shared task. Section 4 presents the data creation process including data sources, preprocessing, annotation guidelines, annotation, and data format. An analysis of the created data set is provided in Section 5. The evaluation method is presented in Section 6, while the descriptions of the submitted systems are presented in Section 7.

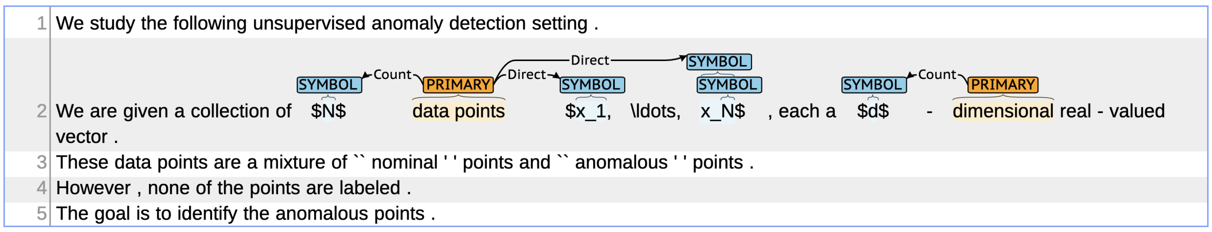

1 We study the following unsupervised anomaly detection setting .
2 We are given a collection of N data points $x_1, \ldots, x_N$, each a d - dimensional real - valued vector .
3 These data points are a mixture of `` nominal ' ' points and `` anomalous ' ' points .
4 However , none of the points are labeled .
5 The goal is to identify the anomalous points .

Figure 1: Example of the Symlink tasks.

2 Related Work

Early studies for scientific literature link formulae to Wikipedia page (Nghiem Quoc et al., 2010; Kristianto et al., 2016). Even though this can provide additional information regarding the mathematical expression, a reader might find it harder to understand the Wikipedia page as it is presented in many unrelated forms. Linking to the description in the same document is more practical (Kristianto et al., 2014; Alexeeva et al., 2020) as the descriptions are dedicated to the symbols and the context presented in the document.

Previous studies on symbol-description extraction rely on pattern matching (Yokoi et al., 2011; Nghiem Quoc et al., 2010) and rule-based algorithms (Alexeeva et al., 2020). These methods might work for observed patterns with an assumption of close proximity between symbol and description. They may fail to capture distant symbol-description pairs and symbols in very complex structures such as algorithms in computer science literature.

Most of the previous studies have attempted to extract and link at formula level (Nghiem Quoc et al., 2010; Kristianto et al., 2014, 2016). In reality, understanding mathematical formulae requires details of atomic symbols e.g. superscript, subscript, function arguments. We believe that addressing the problem at this fine-grain level is crucial to drive future research toward a better understanding of the complex symbol-description extraction task.

Prior to this shared task, some studies have created datasets for similar tasks (Yokoi et al., 2011; Schubotz et al., 2016; Alexeeva et al., 2020). However, one of them is created for publications written in Japanese (Yokoi et al., 2011), making it nearly impossible to transfer to English literature. While two other datasets (Schubotz et al., 2016; Alexeeva et al., 2020) only annotate small-scale golden datasets for evaluation purposes. As the result, no training data is available for training deep neural network models. In this shared task, we provide a large-scale dataset for English literature that we believe will provide enough supervision for the promising deep neural network-based models.

Definition extraction from scientific document is close to the task presented in SemEval Task 12. The Scientific Document Understanding workshop has hosted the Acronym Extraction and Acronym Disambiguation Shared Tasks, namely *Acronym Extraction and Acronym Disambiguation Shared Tasks*(Veyseh et al., 2021a, 2022). The prior studies in this research direction considers extracting definitions from the text (Spala et al., 2019, 2020; Veyseh et al., 2020), or together with acronyms, and acronyms sense disambiguation (Pouran Ben Veyseh et al., 2020, 2021).

3 Task Description

The ultimate goal of Symlink shared task is to extract pairs of mathematical symbols and descriptions from scientific documents. As such, Symlink shared task is a combination of an entity recognition and an entity linking task.

Given a LaTeX source of a paragraph from a scientific document:

- **Named Entity Recognition:** For each paragraph, identify all spans containing mathematical symbols and terminology descriptions.
- **Relation Extraction:** For each pair of entities, identify the relationships between them if it is available among symbols and descriptions using Coref-Description, Coref-Symbol, Direct, Count relation types.

4 Data Annotation

4.1 Data source

We obtain the documents from `arXiv.org`, a repository for preprint scientific articles due to the broad coverage of subjects in scientific articles published in ArXiv. In particular, ArXiv offers articles in physics, mathematics, quantitative biology, computer science, quantitative finance, statistics, electrical engineering, and economics. As such, our obtained papers contain a large number of mathematical symbols and equations, allowing a higher yield of extracted symbol-description relations. Among these subjects, we choose five subjects of mathematics, physics, biology, economics, and computer science for annotation.

4.2 Data preparation

ArXiv open-sources the LaTeX version of their articles, when available. In order to make our Symlink dataset open-access to the whole community, we crawled the metadata of these articles and only selected articles under the CC BY license. Once obtained the LaTeX project, we extracted all the paragraphs from the **.tex** files. We filtered out all short paragraphs with less than 50 words and paragraphs without symbols. Since a formula can be composed in multiple ways such as inline formulae (between \$ \$), displayed formulae (between \$\$ \$\$), or using commands e.g. *array*, to keep the original TeX format of the formulae, all of these math objects are masked before tokenization. Then, we used the SciBERT tokenizer (Beltagy et al., 2019) to tokenize the text. The original math object is then restored. As we observed that many papers have nested math objects, we deleted all the nested objects, hence, having non-nested LaTeX data. This is helpful as it makes the LaTeX documents more similar to the ones generated by the PDF-to-LaTeX tools, which do not contain nested objects.

4.3 Taxonomy

To prepare for the annotation, we designed a taxonomy with 3 general entity types and four relation types. In particular, mathematical symbols are annotated under the tag **SYMBOL**, whereas descriptions are tagged under two labels **PRIMARY**, for single standalone definitions, and **ORDERED**, for the description of multiple terms, whose mentions are not separated without creating non-contiguous mentions. Due to the quadratic numbers of combinations of descriptions and complex math expressions, we only tagged an entity if and only if there is a second entity that pairs with the first entity to form a relationship. For relation, we are particularly interested in two main types of relations: **DIRECT**, linking a symbol with its definition, and **COUNT**, linking a description of a concept with a symbol that is the number of instances of the concept. Due to the sheer number of repetitions and coreferences of both descriptions and symbols, we also annotated **COREF-SYMBOL** relation, linking co-referred symbols, and **COREF-DESCRIPTION** relation, linking co-referred descriptions. Detailed annotation guidelines with examples are presented in Appendix A.

4.4 Annotation

We recruited 10 annotators from the crowdsourcing platform `upwork.com` to annotate scientific papers in the five mentioned domains (each subject was annotated by two annotators). The annotators are explicitly selected based on their demonstrated experiences in reading and writing scientific documents in their expertise field(e.g., holding an M.S. or Ph.D. degree). Detailed annotation guidelines with many examples and explanations are provided to train the annotators. Overall, we annotated 102 papers, accounting for 3,690 paragraphs, and 595K tokens. Our annotators for each domain co-annotate the documents in their domain and achieve Cohen's Kappa scores of (averaged) 0.79. This inter-agreement score thus indicates substantial agreements between our annotators. Eventually, the annotators engage in discussions to resolve any conflict to produce a final consolidated version of our Symlink dataset.

4.5 Data Format

The participants are provided with preprocessed in JSON format. Each paragraph is stored in a JSON object with its id, topic, original LaTeX source, set of entities, and set of relations. An example of the data object is presented in Figure 2.

5 Data Analysis

Table 1 presents the statistics for the dataset including the number of articles, distribution of entities, and distribution of the relations. Overall, our dataset offers more than 31K entities, 20K pairs of relations, which is one order of magnitude larger than existing datasets for a similar task.

Figure 3 presents the distribution of the span lengths of both symbols and descriptions of up to 15 tokens. As can be seen from the figure, the majority of entities have a length of 1-3 tokens. However, overall, the span lengths of both symbols and descriptions vary significantly from 1 up to 47 tokens (note that Figure 3 only illustrates the spans

```
{
  "id": "1503.01158v2...",
  "phase": "test",
  "topic": "cs.ai",
  "document": "1503.01158v2...",
  "paragraph": "paragraph_48",
  "text": "... with a covariance
  matrix of $I$ ; that is , ...",
  "entity": {
   "T1": {
    "eid": "T1",
    "label": "SYMBOL",
    "start": 325,
    "end": 326,
    "text": "I"
   },
   "T2": {
    "eid": "T2",
    "label": "PRIMARY",
    "start": 303,
    "end": 320,
    "text": "covariance matrix"
   }
  },
  "relation": {
   "R1": {
    "rid": "R1",
    "label": "Direct",
    "arg0": "T2",
    "arg1": "T1"
   }
  }
}
```

Figure 2: An example of a paragraph in Symlink dataset.

with up to 15 tokens). This demonstrates a key challenge of the Symbol-Description Linking task in this paper where symbols and descriptions with long spans might introduce confusion for extraction models.

To further understand the dataset, we present the distances between the entities and relations annotated in Symlink by different relation types in Figure 4. The distributions can be grouped into two categories. The first category involves the symbol-description relations while the second group involves the coreference relations. The distributions of symbol-description relations have long tails, indicating that symbols and descriptions tend to appear in close proximity. On the other hand, the distributions of coreference relations are quite flat, suggesting that the coreference relations appear in both short and long distances.

Table 1: Statistics and label distribution of the Symlink dataset. *The texts are tokenized by SciBERT.

	Train	**Dev**	**Test**	**Total**
Statistics				
#Documents	91	6	5	102
#Paragraphs	3,120	270	300	3,690
#Sentences	25,070	1,765	2,286	29,121
#Tokens*	522K	35K	38K	595K
Entity types				
#SYMBOL	18,547	1,504	1,864	21,915
#PRIMARY	7,953	678	907	9,538
#ORDERED	14	3	1	18
Relation types				
#Direct	8,200	731	867	9,798
#Count	1,484	17	221	1,722
#Coref-Symbol	6,821	759	690	8,270
#Coref-Description	612	97	154	863

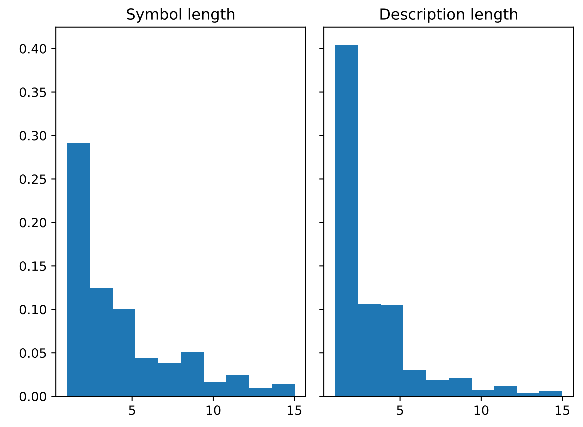


Figure 3: Length of symbols and descriptions in Symlink

6 Evaluation

The results are evaluated separately for the Named Entity Recognition (NER) task and the Relation Extraction (RE) task. For NER, we use the entity-based partial/type from SemEval 2013 Task 9.1. For RE, we use standard precision, recall, F-score metrics. Relations output by the participating system is correct if the prediction label strictly matches the gold standard.

During the 21-day evaluation period (January 10 through 31, 2022), 7 CodaLab users submitted a total of 59 submissions with 37 submissions passing the validation and being scored. Given the complexity of the task, we allow unlimited submissions

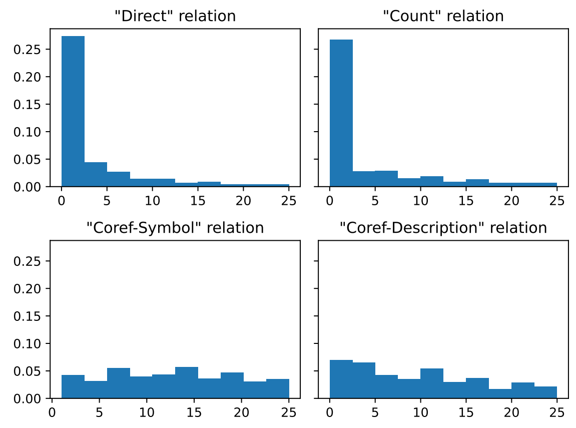


Figure 4: Distribution of distances between entities in Symlink by relation type.

during the evaluation. As such the top submitter tried up to 18 times.

Table 2 shows the performances of the successful submissions. Asterisk denotes teams with system descriptions submitted for review. Among the participated teams, 6 teams performs both Named Entity Recognition and Relation Extraction subtasks while one team tried the Named Entity Recognition subtask only. Figure 5 presents the timelines of submissions and high scores over the evaluation period.

7 Summary of Participating Systems

The Symlink track at SemEval-2022 received 4 system description paper submissions presented in Table 2. Overall, all submitted systems are based on BERT architecture (Devlin et al., 2019). Among those, two out of four systems use SciBERT (Beltagy et al., 2019), while two remaining systems use other variants of BERT such as original BERT (Devlin et al., 2019) and mBERT (Devlin et al., 2019).

7.1 System Specifics

Lee and Na (2022) (JBNU-CCLab) achieved their state-of-the-art performance using SciBERT (Beltagy et al., 2019). Their entity model consists of an MRC-based model (Li et al., 2020), simplifying the tasks as binary classification problems whether span is valid using entity type information as input features. They proposed a simple rule-based Symbol Tokenizer to predict accurately the complex symbols appearing in scientific documents. The relation model exploits entity span information and entity type information as input features using typed entity marker. Additionally, the paper exploited many regularization techniques to improve the model performance such as regularized dropout (Wu et al., 2021) and representational collapse prevention (Aghajanyan et al., 2020) and traditional ensemble techniques.

Popovic and Laurito (2022) (AIFB-WebScience) proposed an end-to-end joint entity and relation extraction approach based on transformer-based language models. Unlike traditional entity and relation extraction methods, which perform the task in sequence, this system incorporates information from relation extraction into entity extraction. As such, the system can be trained even on partially annotated datasets where only a subset of all valid entity spans is annotated.

Ping and Chi (2022) (AN(L)P) participated in the Entity Extraction only. They finetuned a BERT-large model (Devlin et al., 2019) for each domain. For cs.ai domain, they used data from cs.ai only, whereas, for the other domain, they augmented the in-domain data with the data from cs.ai.

der Goot (2022) (MaChAmp) proposed to pretrain a language model and re-finetune after multitask learning for a pre-defined set of semantically focused NLP tasks. They trained a multi-task model for all text-based SemEval tasks that include annotation on the word, sentence, or paragraph level. They compared the performance with models using mBERT (Devlin et al., 2019). The pretrained multi-task embedding showed a consistent improvement across many tasks against the mBERT embedding.

7.2 Symbol tokenizer and detection

In this shared task, the uniqueness of the task is detecting mathematical symbol span. Symbol span in LaTeX source is comprised of both human language and machine language, i.e. LaTeX language. Further, mathematical formulae in LaTeX sources are written in both linear and hierarchical manners. Therefore, a system must consider not only human language modeling but also a highly systematic syntax system of LaTeX source. As such, fundamental tasks such as tokenization is a huge contributor to the robustness of the model.

Among four submitted systems, MaChAmp (der Goot, 2022) and AN(L)P (Ping and Chi, 2022) teams used the default tokenizer from either BERT or mBERT, which are not designed for scientific documents. Consequently, they are unable to correctly segment the mathematic source, hence, they

Table 2: Results for each team/user, ordered by F1-score on Relation Extraction. Team with $*$ submitted their system description paper to SemEval 2022.

Team	Variant	Entity		Relation		
		F1 (partial)	F1(type)	Precision	Recall	F-score
JBNU-CCLab*	Base	47.61	47.70	32.09	38.56	35.03
	+RDrop	47.61	47.70	33.40	38.66	35.84
	+R3F	47.61	47.70	33.77	38.56	36.00
	+R3F,Ensemble	**47.61**	**47.70**	38.20	**36.23**	**37.19**
ZQ	-	39.39	39.51	**57.25**	23.29	33.11
AIFB-WebScience*	Max/Original	37.83	37.88	45.80	20.96	28.66
	Mean/Original	41.21	41.23	42.25	26.55	32.28
	Max/LaTex2Text	38.33	38.38	46.09	21.64	29.45
	Mean/LaTex2Text	34.53	34.64	47.02	18.20	26.24
LingZing	-	33.87	33.93	13.45	10.92	12.05
MaChAmp*	Single mBERT	-	-	-	-	2.67
	Multi RemBERT	25.17	25.25	13.11	5.17	7.42
iyerke	-	6.67	6.46	0.10	0.62	0.17
AN(L)P*	-	-	16.30	-	-	-

achieved the lowest Named Entity Recognition performance. Whereas AIFB-WebScience (Popovic and Laurito, 2022) and JBNU-CCLab (Lee and Na, 2022) achieved much higher performances thanks to SciBERT tokenizer because it is trained on scientific literature. However, the SciBERT tokenizer is far from perfect such that JBNU-CCLab further proposed to tokenize the mathematical formulae using a customized rule-based tokenizer based on capital letters, numbers, and special characters(e.g. %, \$, {, }). Hence, they achieved state-of-the-art performance on both NER and RE subtasks.

8 Conclusion

In this paper, we present the task description, the data annotation, the evaluation, the results, and the descriptions of four submitted systems for Symlink at SemEval 2022. The Symlink shared task is challenging given the complexity of the LaTeX source and partly due to the difference of the domains involved in the data. In this shared task, it is hard to separate the NER and RE subtasks due to their constraints.

The submitted systems employed variants of contextualized embedding BERT for encoding the text. In general, the task can be formatted into similar sequence labeling and relation extraction task. However, special treatments are needed to process LaTeX sources. For instance, a LaTeX-source-trained tokenizer or a customized tokenizer is essential to tokenize the text. Some unique characteristics of the dataset have not been investigated such as the syntax of the LaTeX source, and the hierarchical structure of formulae. These suggest future research directions to improve the robustness of the model.

References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*.

Maria Alexeeva, Rebecca Sharp, Marco A. Valenzuela-Escárcega, Jennifer Kadowaki, Adarsh Pyarelal, and Clayton Morrison. 2020. MathAlign: Linking formula identifiers to their contextual natural language descriptions. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2204–2212, Marseille, France. European Language Resources Association.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M Rush. 2017. Image-to-markup generation with coarse-to-fine attention. In *International Conference on Machine Learning*, pages 980–989. PMLR.

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

Rob Van der Goot. 2022. Machamp at semeval-2022 tasks 2, 3, 4, 6, 10, 11, and 12: Multi-task multilingual learning for a pre-selected set of semantic datasets. In *OpenReview*.

Giovanni Yoko Kristianto, Akiko Aizawa, et al. 2014. Extracting textual descriptions of mathematical expressions in scientific papers. *D-Lib Magazine*, 20(11):9.

Giovanni Yoko Kristianto, Goran Topić, and Akiko Aizawa. 2016. Entity linking for mathematical expressions in scientific documents. In *International Conference on Asian Digital Libraries*, pages 144–149. Springer.

Sung-Min Lee and Seung-Hoon Na. 2022. Jbnu-cclab at semeval-2022 task 12: Fusing maximum entity information for linking mathematical symbols to their descriptions. In *OpenReview*.

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

Minh Nghiem Quoc, Keisuke Yokoi, Yuichiroh Matsubayashi, and Akiko Aizawa. 2010. Mining coreference relations between formulas and text using Wikipedia. In *Proceedings of the Second Workshop on NLP Challenges in the Information Explosion Era (NLPIX 2010)*, pages 69–74, Beijing, China. Coling 2010 Organizing Committee.

Annie Ping and Ethan Chi. 2022. Team an(l)p at semeval-2022 task 12: Building a lightweight symbol recognition system. In *OpenReview*.

Nicholas Popovic and Walter Laurito. 2022. Aifb-webscience at semeval-2022 task 12: Relation extraction firstusing relation extraction to identify entities. In *OpenReview*.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Walter Chang, and Thien Huu Nguyen. 2021. MadDog: A web-based system for acronym identification and disambiguation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 160–167, Online. Association for Computational Linguistics.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, and Thien Huu Nguyen. 2020. What does this acronym mean? introducing a new dataset for acronym identification and disambiguation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3285–3301, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Moritz Schubotz, André Greiner-Petter, Philipp Scharpf, Norman Meuschke, Howard S Cohl, and Bela Gipp. 2018. Improving the representation and conversion of mathematical formulae by considering their textual context. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*, pages 233–242.

Moritz Schubotz, Alexey Grigorev, Marcus Leich, Howard S Cohl, Norman Meuschke, Bela Gipp, Abdou S Youssef, and Volker Markl. 2016. Semantification of identifiers in mathematics for better math information retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 135–144.

Sasha Spala, Nicholas Miller, Franck Dernoncourt, and Carl Dockhorn. 2020. SemEval-2020 task 6: Definition extraction from free text with the DEFT corpus. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 336–345, Barcelona (online). International Committee for Computational Linguistics.

Sasha Spala, Nicholas A. Miller, Yiming Yang, Franck Dernoncourt, and Carl Dockhorn. 2019. DEFT: A corpus for definition extraction in free- and semi-structured text. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 124–131, Florence, Italy. Association for Computational Linguistics.

Amir Veyseh, Franck Dernoncourt, Dejing Dou, and Thien Nguyen. 2020. A joint model for definition extraction with syntactic connection and semantic consistency. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9098–9105.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Thien Huu Nguyen, Walter Chang, and Leo Anthony Celi. 2021a. Acronym identification and disambiguation shared tasks for scientific document understanding. In (Veyseh et al., 2021b).

Amir Pouran Ben Veyseh, Franck Dernoncourt, Thien Huu Nguyen, Walter Chang, and Leo Anthony Celi, editors. 2021b. *Proceedings of the AAAI 2021 Scientific Document Understanding Workshop*, number 2831 in CEUR Workshop Proceedings. Aachen.

Amir Pouran Ben Veyseh, Nicole Meister, Viet Dac Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2022. Acronym extraction and acronym disambiguation shared tasks at the scientific document understanding workshop 2022. In *AAAI 2022 Workshop on Scientific Document Understanding*, CEUR Workshop Proceedings, Aachen.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

Keisuke Yokoi, Minh-Quoc Nghiem, Yuichiroh Matsubayashi, and Akiko Aizawa. 2011. Contextual analysis of mathematical expressions for advanced mathematical search. *Polibits*, (43):81–86.

A Annotation guidelines

This section summarizes some rules that we use to make our annotations more consistent.

Description tagging: A description is usually a noun or a noun phrase that expresses a concept. These are the overall rules for entity annotations:

- We only tag a description if the corresponding symbol presents in the text.
- A description usually is a noun or a noun phrase. Sometimes, a verb, an adverb, or an adjective describes an operation, it is also considered a description.
- Descriptions should be short but it must cover the elements in the corresponding symbol, esp. in case of complex symbols, such as superscript, subscript, arguments, and limits.

Symbol tagging: A mathematical symbol can present an operand, an operator, an expression, or combination of these.

- An atomic symbol in PDF format has to be a character, that means, if we have Y hat, neither Y nor hat is considered an atomic symbol, instead "Y hat" is a symbol. In latex format, \hat{Y} should be annotated.
- A complex symbol is a combination of multiple symbols and brackets, for example: "P(x)", "Wx"
- An annotated symbol has to be a complete symbol e.g. "P(x)" is good, "P(x" is not because of lacking the closing parenthesis.
- A complex formula can be segmented into atomic symbols, we will annotate at all levels of the complex symbol as long as there are appropriate descriptions available.

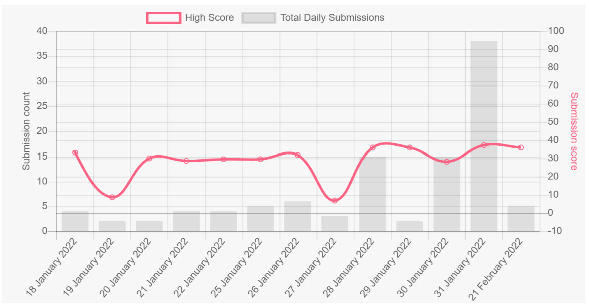


Figure 5: Submission counts and top performances during the evaluation period. The submission score is the F1-score of the RE task.

Relation annotation:

- Every annotated symbol/description has to have at least one relation linking to its description/symbol.
- If there are multiple mentions of a single symbol/description, use coreference relation to link them. A direct relation or a count relation is used to link the closet pair of symbol and description.

B Timeline of submissions

Figure 5 presents the number of submissions over the evaluation of the task.

JBNU-CCLab at SemEval-2022 Task 12: Machine Reading Comprehension and Span Pair Classification for Linking Mathematical Symbols to Their Descriptions

Sung-Min Lee and **Seung-Hoon Na**
Division of Computer Science and Engineering, Jeonbuk National University, South Korea
`{cap1232,nash}@jbnu.ac.kr`

Abstract

This paper describes our system in the SemEval-2022 Task 12: 'linking mathematical symbols to their descriptions', achieving first on the leaderboard for all the subtasks comprising named entity extraction (NER) and relation extraction (RE). Our system is a two-stage pipeline model based on SciBERT that detects symbols, descriptions, and their relationships in scientific documents. The system consists of 1) machine reading comprehension(MRC)-based NER model, where each entity type is represented as a question and its entity mention span is extracted as an answer using an MRC model, and 2) span pair classification for RE, where two entity mentions and their type markers are encoded into span representations that are then fed to a Softmax classifier. In addition, we deploy a rule-based *symbol tokenizer* to improve the detection of the exact boundary of symbol entities. Regularization and ensemble methods are further explored to improve the RE model. [1]

1 Introduction

Mathematical symbols and descriptions appear in various forms across document section boundaries without explicit markups, and mathematical symbols appear in the form of long texts. Thus, linking mathematical symbols and their descriptions is challenging.

SemEval 2022 task 12: 'linking mathematical symbols to their descriptions (Lai et al., 2022a)', is a relation extraction task targeted at scientific documents divided into two sub-tasks: sub-task A is a **named entity recognition** (**NER**) task that aims to predict the span of symbols and descriptions, and sub-task B is a **relation extraction** (**RE**) task that aims to predict relations between symbols and descriptions.

[1]Our code is publicly available at `https://github.com/ZIZUN/symlink`.

Extracting these entities and relations is done to discover relational facts from unstructured texts. This problem can be decomposed into NER (Tjong Kim Sang and De Meulder, 2003; Ratinov and Roth, 2009) and RE (Zelenko et al., 2002; Bunescu and Mooney, 2005). Early works employed a two-stage relation extraction system, training one model to extract entities (Florian et al., 2004) and another model to classify relations between these entities (Zhou et al., 2005; Chan and Roth, 2011). To reduce the error propagation of NER or better capture the interactions between NER and RE, joint models have been proposed as a promising approach that are based on an end-to-end method or on the setting of multi-task learning using shared representations (Wadden et al., 2019; Lin et al., 2020; Wang and Lu, 2020).

Recently, it has been observed that RE based on the shared encoder is suboptimal, but the use of separated encoders for NER and RE has shown improved performance compared to shared encoders, reexamining the effectiveness of the simple pipelined two-stage approach (Zhong and Chen, 2021; Ye et al., 2021). From these results, we hypothesize that whereas separated encoders for NER and RE can learn customized representations useful for each task, joint models may include irrelevant information in the learned representation for NER or RE tasks, lowering the performance of the model.

These results of using distinct encoders (Zhong and Chen, 2021) encourage us to adopt the aforementioned two-stage approach for NER and RE tasks, consisting of 1) MRC-based NER and 2) span pair classification for RE, as follows:

1. **MRC-based NER using a symbol tokenizer**: Unlike the PURE system of (Zhong and Chen, 2021) that exploits the standard span-based NER of (Lee et al., 2017; Wadden et al., 2019), our NER model is based on an MRC-based model (Li et al., 2020), which treats NER as

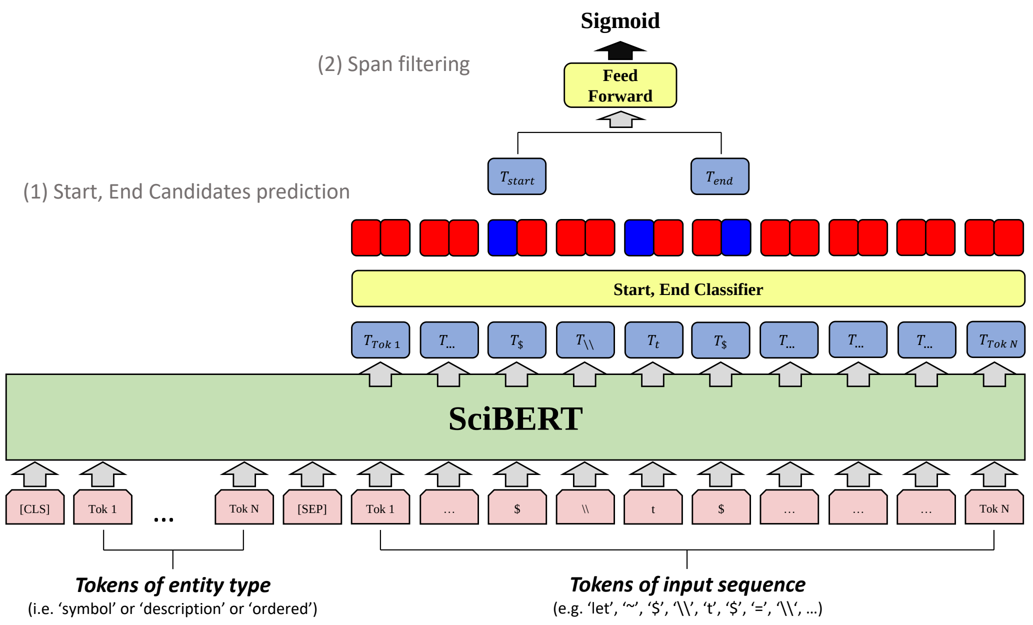


Figure 1: Our NER model architecture based on MRC

an MRC problem by providing an entity type as a question and using an MRC model to extract its entity mentions as answers. As in (Li et al., 2020), an MRC model is based on two binary classification models: first, the position classifier predicts the start and end indexes to create a set of valid answer spans, second, the span classifier determines whether each of the valid spans is an answer. As a pretrained encoder for the NER model, SciBERT of (Beltagy et al., 2019) is used. Before presenting to SciBERT's tokenizer, we apply a rule-based *symbol tokenizer* to precisely predict the span boundary of mathematical symbols that appear in scientific documents,

2. **Span pair classification for RE with solid markers**: Similar to the PURE system of (Zhong and Chen, 2021), a pair of spans resulting from the NER model is given as an input but with *solid markers*, i.e., using a typed entity marker, as in the works of (Wu and He, 2019; Zhou and Chen, 2021). The SciBERT encoder then uses this marked input to generate contextualized representations, which are then transformed to a pair of span representations and fed into a Softmax classifier. To define a set of relation types (or classes), the RE model explicitly adds a *NIL*-type class as a relation type to refer to the case in which a pair of spans has no relationship. It should be noted that the NER model's symbol tokenizer is not used in the RE model. Regularization methods such as *RDrop* (Wu et al., 2021) and *R3F* (Aghajanyan et al., 2020), as well as traditional ensemble techniques, are used to improve the performance of RE models[2].

The remainder of this paper is organized as follows: Section 2 presents our system architecture in detail, Sections 3-5 describe the experimental setting, results, and ablation studies, and Section 6 contains our concluding remarks and future works.

2 System Overview

In this section, we first describe the models of the proposed system for each sub-task.

2.1 MRC-based NER with a symbol tokenizer

Figure 1 shows our MRC-based NER model, that extracts mathematical symbols and descriptions from scientific documents.

2.1.1 Symbol tokenizer as a pre-tokenizer

We discovered in our preliminary experiment that SciBERT's tokenizer is not optimal for extracting the boundaries of mathematical symbols, because non-alphanumeric characters are important in the mentions of symbol-type entities. We perform a

[2]Regularization and ensemble methods were not adopted in the NER model.

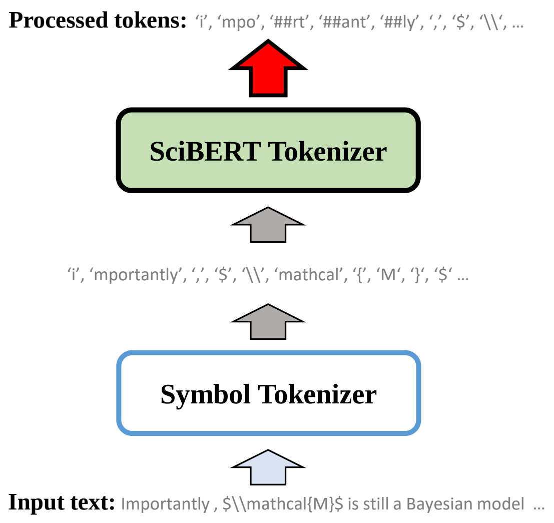

Processed tokens: ‘i’, ‘mpo’, ‘##rt’, ‘##ant’, ‘##ly’, ‘,’, ‘$’, ‘\\’, ...
SciBERT Tokenizer
‘i’, ‘mportantly’, ‘,’, ‘$’, ‘\\’, ‘mathcal’, ‘{’, ‘M’, ‘}’, ‘$’ ...
Symbol Tokenizer
Input text: Importantly , $\\mathcal{M}$ is still a Bayesian model ...

Figure 2: Two-step tokenization process for NER model with a symbol tokenizer

rule-based *symbol tokenizer* as a pre-tokenizer before applying SciBERT's tokenizer to precisely detect the boundary of symbol-type entities. Figure 2 presents this two-step tokenization adopted for the NER model.

The symbol tokenizer seperates mathematical symbols based on capital letters, numbers, and special characters (e.g., %, \$, }, {). Our symbol tokenizer's rules are derived heuristically from a training dataset[3].

2.1.2 MRC models for nested NER

Given that the dataset addresses nested entities, we use the MRC model of (Li et al., 2020), that takes a *question-augmented* input. Specifically, suppose that $X = [x_1, \cdots, x_n]$ is a sequence of tokens in a scientific document, where n is the length of the sequence. Given a target entity type t, its natural language form $Q_t = [q_1, \cdots, q_m]$ is provided as a question based on Table 1, where q_i is the i-th token of Q_t and m is the length of the question. The question-augmented input X' is formulated as follows:

$$X' = \text{[CLS]}, q_1, \cdots, q_m, \text{[SEP]}, x_1, \cdots, x_n$$

Type	Text
SYMBOL	symbol
PRIMARY	description
ORDERED	ordered

Table 1: Natural language forms mapped for entity types

[3]This rule-based symbol tokenizer is also included in our codes.

Then, as a pre-trained language model, we apply SciBERT's encoder trained from scientific domain documents to obtain contextualized representations $T \in \mathbb{R}^{n\times d}$ over n tokens in a given document X, where d is the dimensionality of SciBERT's hidden representation.

The NER model predicts the probability of each token being a start or end index as follows:

$$\begin{aligned} P_{start} &= Sigmoid(FFN^{(start)}(T)) \in \mathbb{R}^n \\ P_{end} &= Sigmoid(FFN^{(end)}(T)) \in \mathbb{R}^n \end{aligned} \tag{1}$$

where $FFN^{(start)}$ ($FFN^{(end)}$) is a feed-forward neural network layer for predicting the start position and P_{start} (P_{end}) represents the probability of each index being the start (end) position of an entity, given a question entity type.

Based on Eq. (9), we obtain sets of predicted start and end indices as follows:

$$\begin{aligned} I_{start} &= \left\{ i \middle| \mathbb{1}(P^{(i)}_{start} > 0) \right\} \\ I_{end} &= \left\{ i \middle| \mathbb{1}(P^{(i)}_{end} > 0) \right\} \end{aligned} \tag{2}$$

where $P^{(i)}_{start}$ ($P^{(i)}_{end}$) is the i-th element of P^{start} (P^{end}) and $\mathbb{1}$ is an indicator function that gives 1 if an element is true, and 0 otherwise.

For any start index $i_{start} \in I_{start}$ and $i_{end} \in I_{end}$, a binary classifier is applied to predict whether the span of (i_{start}, i_{end}) becomes an answer, as follows:

$$\begin{aligned} &P_{i_{start}, i_{end}} = \\ &Sigmoid\left(FFN^{(span)}(T_{i_{start}}; T_{i_{end}})\right) \end{aligned} \tag{3}$$

where ; is the concatenation operator and $FFN^{(span)}$ is an additional feed-forward neural network layer for the span prediction.

Training As in (Li et al., 2020), the loss function for predicting the start and end positions is based on the cross-entropy term, which is formulated with probabilities of indexes being the start and end positions, as follows:

$$\begin{aligned} \mathcal{L}_{start} &= CE(P_{start}, Y_{start}) \\ \mathcal{L}_{end} &= CE(P_{end}, Y_{end}) \end{aligned} \tag{4}$$

where $Y_{start} \in \{0, 1\}^n$ and $Y_{end} \in \{0, 1\}^n$ represent the gold start and end positions, respectively of input tokens. The loss function for span probability is formulated as follows:

$$\mathcal{L}_{span} = CE(P_{start,end}, Y_{start,end}) \tag{5}$$

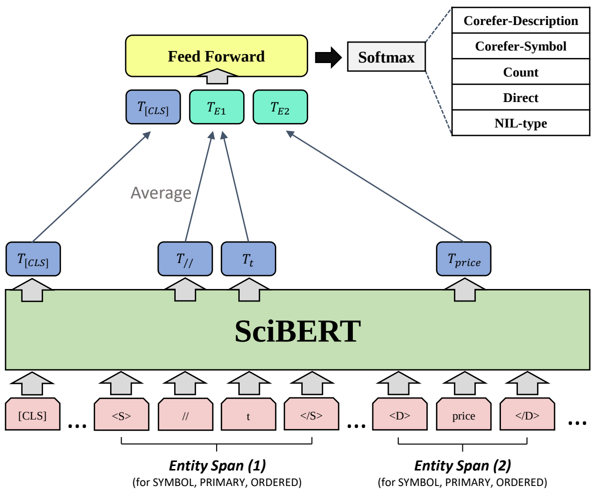


Figure 3: Span pair classification models for RE

where $Y_{start,end}$ represents the gold span of the input tokens. The overall loss is formulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{start} + \lambda_2 \mathcal{L}_{end} + \lambda_3 \mathcal{L}_{span} \quad (6)$$

where $\mathcal{L}_{start}$, $\mathcal{L}_{end}$, and $\mathcal{L}_{span}$ are the loss functions for predicting the start, end positions, and for the span prediction task, respectively, and λ_i is the weight for each loss function.

2.2 Span pair classification for RE with solid markers

Figure 3 represents the RE model based on the span pair classification of (Wu and He, 2019; Zhou and Chen, 2021), that classifies a pair of entity spans extracted from the MRC-based NER model in Section 2.1.2.

Like the NER model, we use SciBERT as a pre-trained language model for the RE model, but keep separate parameters that are not shared with the NER's encoder, following the work of (Zhong and Chen, 2021).

In the RE model, a *type-marked* document is provided as input. Specifically, suppose that e_1 and e_2 are a pair of entity spans (i.e., sequences of tokens), and their types are t_1 and t_2, respectively. Then, a type-marked document $\hat{X}$ is defined by prepending and appending *type markers* before and after each entity span, as follows:

$\hat{X}$ =[CLS] $\cdots$ <t_1> e_1 </t_1> $\cdots$ <t_2> e_2 </t_2> $\cdots$ where <t_i> and </t_1> are type markers.

Given $\hat{X}$, we apply SciBERT's encoder to obtain contextualized representations $T^{(rel)}$. We then obtain span representations for e_1 and e_2 by mean-pooling over their contextual representations, as follows:

$$H_{e_i} = \frac{1}{(end_{e_i} - start_{e_i} + 1)} \sum_{j=start_{e_i}}^{end_{e_i}} T_j^{(rel)} \quad (7)$$

where $start_{e_i}$ and end_{e_i} represent the start and end positions of e_i in the type-marked document $\hat{X}$, respectively. Finally, the model predicts the $P_{relation} \in \mathbb{R}^{l+1}$ probabilities over the relation types of e_1 and e_2 as follows:

$$P_{relation} = softmax(FFN^{(rel)}(T_{[CLS]}^{(rel)}; H_{e_1}; H_{e_2})) \quad (8)$$

where l is the number of relation types, NIL-type of relation is presented as $l+1$-th type, $FFN^{(rel)}$ is an additional feed-forward neural network for relation classification, and $T_{[CLS]}^{(rel)}$ (i.e., the contextual representation of the [CLS] token of $\hat{X}$) is concatenated to provide a global context over the entire document.

During training, the loss function for relation classification uses a cross-entropy function, which is formulated as follows:

$$\mathcal{L} = CE(P_{relation}, Y_{relation}) \quad (9)$$

where $Y_{relation} \in \{0,1\}^{l+1}$ represents a one-hot vector for the gold-relation label of a given pair of entities.

Tokenization Unlike the NER model in Section 2.1.2, only SciBERT's WordPiece tokenizer is exploited.

2.2.1 Automatic creation of examples for NIL-type class

Because we do not have explicit training examples for the NIL-type class, we use a simple negative sampling method to train the RE model. When a pair of entities appear in the context within the maximum length of tokens, they are considered *negative* samples (i.e., examples for the NIL-type class) when they do not have any relationship. The number of NIL-type samples collected in this manner, however, was more than 10 times that of normal samples. To correct the data imbalance, we use *oversampling* of (Chawla et al., 2002) on normal positive samples. Oversampling is also used to balance the positive, negative examples in the development set.

Model	NER				RE		
	Strict	**Exact**	**Partial**	**Type**	**Precision**	**Recall**	**F1 score**
Our System	-	-	47.61	47.70	32.09	38.56	35.03
+ *RDrop*	-	-	-	-	33.40	38.66	35.84
+ *R3F*	-	-	-	-	33.77	38.56	36.00
+ *R3F*, Ensemble	-	-	-	-	38.20	36.23	37.19

Table 2: Performances of final submission runs of our NER and RE models on the test dataset

3 Experimental setup

3.1 Dataset

We use the SemEval-2022 Task 12 dataset(Lai et al., 2022b) in our experiments.

The NER dataset contains three entity types: SYMBOL, PRIMARY, ORDERED. SYMBOL is a mathematical symbol, PRIMARY is a primary description, and ORDERED is a description of multiple terms.

The RE dataset contains four relation types: DIRECT, COUNT, COREFER-DESCRIPTION, and COREFER-SYMBOL. The dataset annotation guidelines[4] state that the relations should be directional; however, some of the relations, such as COREFER-SYMBOL, are *unidirectional*. COREFER-SYMBOL($\mathbf{E_1}$, $\mathbf{E_2}$) is the same as COREFER-SYMBOL($\mathbf{E_2}$, $\mathbf{E_1}$),

The directions of the other relations are defined based on the entity types. Such examples include COUNT($\mathbf{E_1}$, $\mathbf{E_2}$) and DIRECT($\mathbf{E_1}$, $\mathbf{E_2}$) where $\mathbf{E_1}$ is a symbol-type entity and $\mathbf{E_2}$ is a description-type entity. In postprocessing, the directions of these relations are automatically determined based on entity types in a post-processing manner. In other words, for the RE model, the order of the two entity spans e_1 and e_2 is determined based on their corresponding entity types.

Our system is evaluated separately for the NER and RE tasks. For NER, we use the entity-based strict/exact/partial/type from SemEval 2013 Task 9.1 (Segura-Bedmar et al., 2013). We use the standard precision, recall, f1-score metrics for RE.

3.2 Regularization and ensemble for RE model[5]

We use two regularization methods to improve the performance of the RE model: *RDrop* (Wu et al., 2021) and *R3F* (Aghajanyan et al., 2020). *Rdrop* is a regularization method that reduces the difference between representations at inference and training time caused by dropout, and *R3F* is a regularization method that maintains more generalizable representations of the pretrained language model during fine-tuning.

We train 10 RE models using different random seeds for the ensemble inference and and then perform maximum voting for each entity pair.

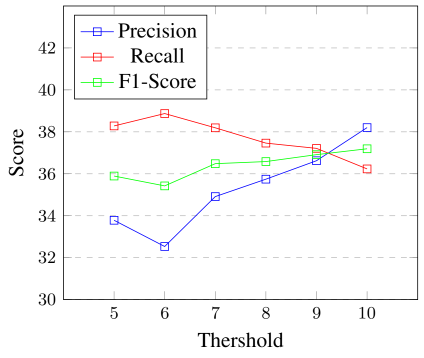


Figure 4: Performance comparison of ensembles of 10 RE models varying thresholds

4 Experimental results

Table 2 presents the final results on the blind test dataset[6].

As shown in Table 2, using the regularization method improves performance over the baseline model. Among the two methods, *R3F* is better than *RDrop*; thus, we use *R3F* for the submission of the RE model.

Overall, the recall is relatively higher than the precision for the RE model. In our preliminary experiments, we observed a similar tendency with high recall for the ensemble method, despite the fact that the ensemble method was shown to be effective in terms of F1 score.

We use a *voting threshold* to increase the pre-

[4]Official annotation guidelines are available at `http://nlp.uoregon.edu/download/symlink/guideline.pdf`

[5]These regularization and ensemble methods were not applied to NER model.

[6]Due to a submission error, we do not report strict/exact scores at the NER task.

cision of the ensemble RE model and adjust the ensemble inference so that a NIL-type class is assigned either when the size of the majority votes from the models does not exceed the voting threshold or when the class from the majority votes is NIL-type.

Figure 4 shows the performance of the ensemble method across different voting threshold values. In this case, we observe that as the voting threshold is raised, the F1 score gradually increases, while precision increases and recall decreases. As a result, when voting threshold is 10, the best performance of the F1 score is obtained. This run was finally submitted.

5 Analysis

In this section, we examine the effects of some of the components of our system as well as additional trials.

5.1 Effect of symbol tokenizer on NER task

Symbol Tokenizer	**Exact**	**Not Exact**	**Recall**
Used	18668	85	99.54
Unused	18412	341	98.18

Table 3: Frequencies and recalls of SYMBOL-type entities whose sequences of tokens are exact gold spans, with and without *symbol tokenizer*.

To examine the effect of the symbol tokenizer, Table 3 compares the frequencies and recalls of SYMBOL-type entities whose exact gold spans are correctly obtained when and without the symbol tokenizer. In this case, recall is defined as the ratio of the number of symbol-type entities whose exact span boundaries are *extractable* using a given tokenizer to the total number of symbol-type entities.

5.2 Effect of removing non-relational entities

Method	NER			
	Strict	**Exact**	**Partial**	**Type**
Not Excluded	-	-	47.61	47.70
Excluded	-	-	47.18	47.31

Table 4: Performances of NER models when including or excluding non-relational entities that have no relationship with other entities.

Assuming that mathematical symbols and descriptions must have one or more relations according to the annotation guideline, our additional trial is to exclude *non-relational* entities that have no relationship with other entities. Table 4 shows the performance of our NER models when those non-relational entities are included or excluded. However, it is observed that removing non-relational entities reduces the performance of the NER model.

5.3 Analysis of RE model

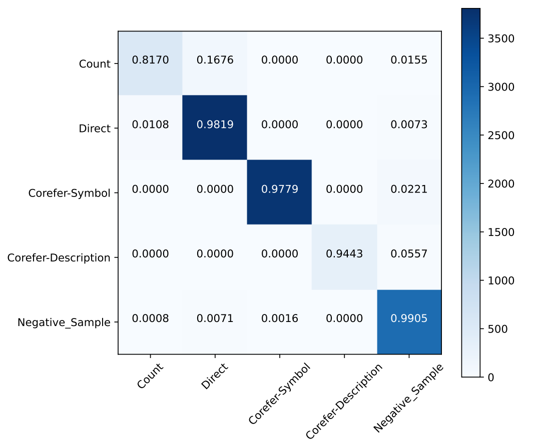


Figure 5: Confusion matrix of RE model on development dataset

Figure 5 shows the confusion matrix of the RE model. It is observed that the discrimination between COUNT and DIRECT is particularly challenging, and the effectiveness of COREFER-DESCRIPTION is relatively low. For this reason, there may be a low number of examples for COUNT and COREFER-DESCRIPTION labels. Given our assumption that these weak performances come from a lack of sufficient number of examples, data augmentation may need to be necessary to improve the performances of these relation labels.

6 Conclusion

Our system shows first for all subtasks of SemEval-2022 Task 12: 'linking mathematical symbols to their descriptions'. MRC-based NER and span pair classification for NER are part of our system that uses SciBERT as a backbone encoder. To improve the performance, the symbol tokenizer for NER model, regularization, and ensemble methods, for RE model are used.

To improve the performance further, future work should look into data augmentation and mathematical symbol and description-aware pretraining.

Acknowledgement

This work was supported by Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-02068, Artificial Intelligence Innovation Hub)

References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 551–560, Portland, Oregon, USA. Association for Computational Linguistics.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. Smote: Synthetic minority over-sampling technique. 16(1):321–357.

R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Viet Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Huu Nguyen. 2022a. Semeval 2022 task 12: Symlink: Linking mathematical symbols to their descriptions. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Viet Dac Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Huu Nguyen. 2022b. Symlink: A new dataset for scientific symbol-description linking.

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

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

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. 2013. SemEval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA. Association for Computational Linguistics.

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

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online. Association for Computational Linguistics.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364.

Deming Ye, Yankai Lin, and Maosong Sun. 2021. Pack together: Entity and relation extraction with levitated marker. *arXiv preprint arXiv:2109.06067*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 71–78. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 427–434, Ann Arbor, Michigan. Association for Computational Linguistics.

Wenxuan Zhou and Muhao Chen. 2021. An improved baseline for sentence-level relation extraction. *arXiv preprint arXiv:2102.01373*.

A Comparison of regularization methods

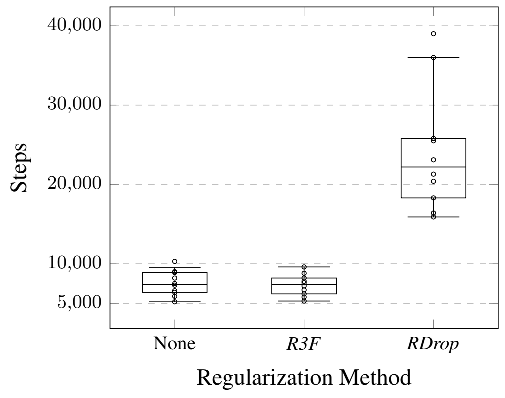


Figure 6: Comparison of the number of steps required for regularization methods in RE models.

We tried *RDrop* and *R3F* as regularization methods, and there were differences in terms of not only performance but also the *training time*. To compare training time, we measured the number of steps required for training our RE model. The results are shown in Figure 6.

B Hyper-parameters

NER model	
Sliding window	100
Dropout rate	0.1
Learning rate	3e-5
$\lambda_1, \lambda_2, \lambda_3$	1, 1, 0.1
Warmup steps	1000
Scheduler	OneCycle
Optimizer	AdamW
Max length	512
Batch size	2
Accumulation steps	5
Span classifier Inter hidden	2048
RE model	
Dropout rate	0.1
Learning rate	4e-5
Warmup steps	1000
Scheduler	Cosine
Optimizer	AdamW
Max length	512
Batch size	32
Accumulation steps	2

Table 5: Hyper-parameter settings

Table 5 shows the setup of hyper-parameters of our NER and RE models. We ran the experiments using 4 TITAN RTX(24GB) GPUs.

AIFB-WebScience at SemEval-2022 Task 12: Relation Extraction First - Using Relation Extraction to Identify Entities

Nicholas Popovic
Karlsruhe Institute of Technology
popovic@kit.edu

Walter Laurito
FZI Research Center for Information Technology
laurito@fzi.de

Michael Färber
Karlsruhe Institute of Technology
michael.faerber@kit.edu

Abstract

In this paper, we present an end-to-end joint entity and relation extraction approach based on transformer-based language models. We apply the model to the task of linking mathematical symbols to their descriptions in LaTeX documents. In contrast to existing approaches, which perform entity and relation extraction in sequence, our system incorporates information from relation extraction into entity extraction. This means that the system can be trained even on data sets where only a subset of all valid entity spans is annotated. We provide an extensive evaluation of the proposed system and its strengths and weaknesses. Our approach, which can be scaled dynamically in computational complexity at inference time, produces predictions with high precision and reaches 3rd place in the leaderboard of SemEval-2022 Task 12. For inputs in the domain of physics and math, it achieves high relation extraction macro F_1 scores of 95.43% and 79.17%, respectively. The code used for training and evaluating our models is available on GitHub[1].

1 Introduction

Information extraction systems are a key component in making scientific literature more consumable. With the large amount of scientific works which are constantly being published (e.g., more than 60,000 machine learning papers per year (Färber, 2019)), indexing techniques that go beyond keyword searches are becoming more important. While many efforts have focused on the processing of abstracts as a way of building representations of publications (Gábor et al., 2018; Luan et al., 2018), methods processing full text documents will be needed to accurately capture their contents for use cases such as academic search and recommender systems and scientific impact quantification.

The task tackled in this paper (Lai et al., 2022), consisting of linking mathematical symbols to their descriptions in LaTeX documents, is a *joint entity and relation extraction* task. While earlier work tackled both subtasks sequentially via separate models, more recent approaches tend to use a single joint model (Luan et al., 2018; Bekoulis et al., 2018; Nguyen and Verspoor, 2019; Eberts and Ulges, 2021). In contrast to early approaches, which are based on Bi-LSTMs (Luan et al., 2018; Bekoulis et al., 2018; Nguyen and Verspoor, 2019), more recent approaches (Wadden et al., 2019; Eberts and Ulges, 2021) make use of transformer-based language models, such as BERT (Devlin et al., 2019). A key challenge in joint models is the computational complexity stemming from pairwise comparisons between entity spans required for relation extraction. Previous works tackle this using a span scoring mechanism based on a feed forward neural network, which produces a score indicating the likelihood that a span is in a relation (Luan et al., 2018; Wadden et al., 2019). Relation extraction is then performed on only those spans with the highest scores. For data sets which include span annotations even for entities which are not in any relation, such as DocRED (Yao et al., 2019), as examined by Eberts and Ulges (2021), such a scoring mechanism is not necessary, because the entity extraction component of the model can be trained on these annotations. For the task tackled in this paper, complete annotations for entity spans are not provided, making the use of a span scoring mechanism necessary.

In this paper, we propose an end-to-end approach for joint entity and relation extraction. The approach is based on a transformer-based language model, following previous work (Eberts and Ulges, 2020, 2021), but is peculiar in the sense that it incorporates a span scoring mechanism based on dot product similarity which is learned via triplet loss rather than cross entropy loss, making it applicable to datasets which contain annotations only for a subset of all valid entity mention spans.

[1] https://github.com/nicpopovic/RE1st

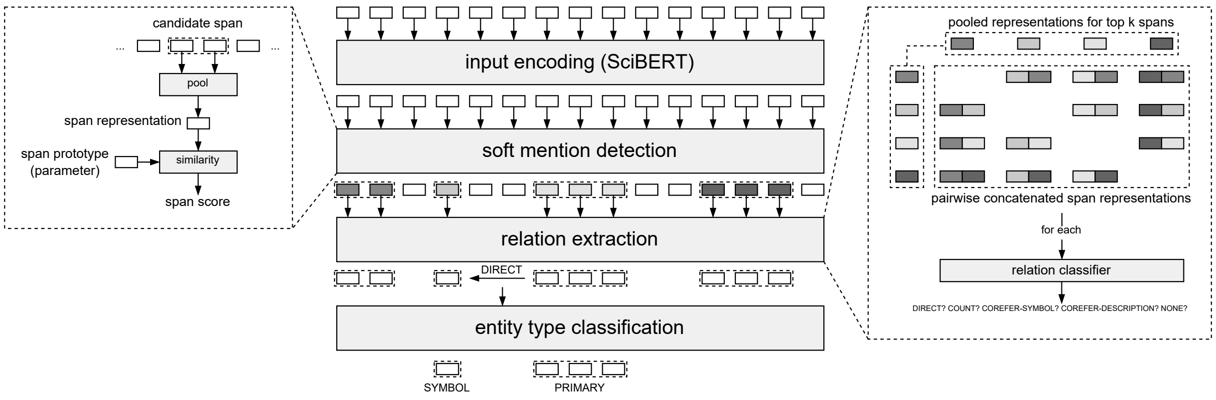

candidate span
pool
span representation
span prototype (parameter)
similarity
span score
input encoding (SciBERT)
soft mention detection
relation extraction
DIRECT
entity type classification
SYMBOL
PRIMARY
pooled representations for top k spans
pairwise concatenated span representations
for each
relation classifier
DIRECT? COUNT? COREFER-SYMBOL? COREFER-DESCRIPTION? NONE?

Figure 1: Architecture overview with detail illustrations for the soft mention detection (left) and relation extraction (right) modules. The layout of this figure was inspired by a similar figure found in (Eberts and Ulges, 2021).

2 Task Description

The task tackled in this paper is one of joint entity and relation extraction. This means, given an unannotated text as input, a system needs to (1) return annotations of relevant entity mention spans, (2) perform coreference resolution, (3) entity type classification, and finally (4) relation extraction on the identified spans. The specific task at hand has a number of key features that separate it from similar settings.

First, regarding entity extraction, the annotations and, thus, the final scoring are restricted to those entities which participate in relations. This means that a system which correctly identifies all symbols and descriptions in the input will score poorly even on the entity extraction portion of the final benchmark if the relation extraction is incorrect. More importantly from an engineering perspective, the resulting span annotations are incomplete in that they only include a partial set of valid spans for each document. In the entity extraction step we can, therefore, only reliably identify true positives and false negatives, not, however, false positives and true negatives.

Second, while coreference resolution (i.e., the linking of multiple mentions to a single entity) is part of the task, relation extraction is to be performed on a mention-level rather than the entity-level. This means that although a system may correctly identify a text span as being the description of a certain symbol, this classification will only be deemed correct in the evaluation if linked to the correct mention of said symbol. As a result, coreference links are interpreted as relations between mentions and thereby as part of the relation extraction subtask, rather than as part of the entity extraction subtask.

Third, entity types can be reliably inferred from the relations between them, meaning that instances of relations are only found between certain entity types. This feature can be used to inform the design of a system in two ways: Either, the task of relation extraction can be simplified by reducing the choices given to a classifier based on the entity types of two spans (i.e., a symbol cannot be the description to another symbol, therefore any such prediction can be disregarded), or the entity type classification can be informed by the relation extraction (i.e., if we identify a span A as the description of another span B, span A must be a description, while span B must be a symbol).

3 Approach

We propose an end-to-end entity and relation extraction system using a transformer-based language model, as illustrated in figure 1. The system consists of 4 modules: (1) The *input encoding module* tokenizes the input text and produces contextualized embeddings for each token, (2) the *soft mention detection module* ranks possible token spans by the likelihood with which they contain an entity mention, (3) the *relation extraction module* extracts relations on a subset of the highest ranked spans from the previous step, and finally (4) the *entity type classification module* assigns entity types to spans based on the relations detected between them.

3.1 Input Encoding

We examine two separate options of encoding the input: For the first option, we pass the input text to the language model without prior modification, whereas for the second option, we perform preprocessing on the input to remove LaTeX code from the text portions of the input. Any input in LaTeX math mode is passed to the model unchanged.

Since our approach uses a transformer-based language model, the input needs to be tokenized. As a result of the tokenization, there are instances of relations which cannot be matched correctly by our model, due to the annotated span boundaries being contained within a token. For the training and development sets, this occurs in 1.99% and 2.84% of relation instances, respectively, and in these cases we adjust the labels accordingly.

3.2 Soft Mention Detection

Given that we cannot reliably identify false positives and true negatives from our labeled data, a mention detection strategy based on cross-entropy loss cannot be used for this task. Instead of following previous approaches in using feed-forward neural networks (Luan et al., 2018; Wadden et al., 2019), we propose a linear similarity based approach which ranks possible spans based on their similarity to multiple *prototype* embeddings (one prototype per entity type).

We begin by computing the set of all possible continuous spans up to a maximum length n and produce a fixed-size embedding e_s for each span by pooling the contextualized embeddings of all tokens within it. As pooling strategies we use either mean or max pooling. For each span embedding e_s we compute a span score X_s:

$$X_s = \max_{a_i \in A}(sim(e_s, a_i)) \qquad (1)$$

where A is the set of prototype embeddings which contains an embedding for each entity type and $sim(a, b)$ is the dot product similarity of two vectors. We select the k spans with the highest values for X_s as our *candidate mentions* M for relation extraction. We compute the *mention loss* as the mean triplet loss (Schroff et al., 2015) across all prototype embeddings in A and all mentions in M.

3.3 Relation Extraction

For relation extraction, we use the document-level relation extraction model DL-MNAV (Popovic and Färber, 2022). We use the concatenation of two span representations as a representation for the relation between them (Wang et al., 2019). The resulting relation representations are compared to a single relation prototype embedding per relation type, as well as m additional prototypes representing the none-of-the-above class (this follows the MNAV model (Sabo et al., 2021)). The relation type corresponding to the prototype resulting in the highest dot product similarity for a relation representation is used as the predicted type. As loss function for the relation classification we use *adaptive thresholding loss* (Zhou et al., 2021) as it is capable of handling the large imbalance between positive and negative training examples present in document-level relation extraction tasks.

Due to quadratic scaling of the pairwise comparisons it is not feasible to perform relation extraction on all possible continuous spans. We, therefore, perform relation classification on the top k spans[2] with the highest span scores, meaning that we have to classify a maximum of $k(k-1)$ relation representations for a given input text. The computational complexity of the system can, therefore, be adjusted dynamically at inference time by changing k, for example to be run on GPUs with smaller memory capacity or on GPUs with higher memory capacity to improve the quality of predictions.

As a result of the soft mention detection, it is possible that some of the k spans are overlapping and correspond to the same target (see appendix A.2 for examples). This means that the relation classifier may output multiple predictions for the same relation instance with slightly different mention spans. For predictions in which both the head and tail entity overlap, we therefore output only the prediction with the highest classification score.

3.4 Entity Type Classification

Finally, we use a simple mapping to determine the entity type of the spans which participate in the relations predicted by the relation classifier. The mapping used can be found in appendix A.1. For spans classified as "PRIMARY" we additionally change the predicted type to "ORDERED", if they are the head entity of more than one "Direct" relation.

4 Experimental Setup

For our language model we use SciBERT (Beltagy et al., 2019), which is trained on scientific

[2]During training we add annotated spans which are not among the top k spans.

		Entity Extraction				Relation Extraction		
pooling	preprocessing	F_1 strict [%]	F_1 exact [%]	F_1 partial [%]	F_1 type [%]	precision [%]	recall [%]	F_1 [%]
				Development set				
max	None	**59.79** $\pm$ **0.99**	**60.19** $\pm$ **0.99**	**69.20** $\pm$ **0.68**	**67.45** $\pm$ **1.05**	**65.86** $\pm$ **1.34**	44.14 $\pm$ 0.97	**52.86** $\pm$ **1.10**
mean	None	58.02 $\pm$ 3.67	58.49 $\pm$ 3.70	69.14 $\pm$ 2.02	66.98 $\pm$ 2.31	61.27 $\pm$ 5.09	**44.58** $\pm$ **1.56**	51.61 $\pm$ 2.87
max	LaTeX2Text	58.90 $\pm$ 0.79	59.30 $\pm$ 0.87	68.69 $\pm$ 1.00	66.88 $\pm$ 1.09	64.54 $\pm$ 2.61	43.77 $\pm$ 1.76	51.66 $\pm$ 0.77
mean	LaTeX2Text	54.59 $\pm$ 15.07	54.99 $\pm$ 14.44	65.62 $\pm$ 11.28	64.22 $\pm$ 14.22	63.89 $\pm$ 33.24	40.59 $\pm$ 15.90	49.64 $\pm$ 23.63
				Test set				
max	None	-	-	37.83 $\pm$ 0.85	37.88 $\pm$ 0.85	45.80 $\pm$ 5.80	20.96 $\pm$ 0.08	28.66 $\pm$ 1.19
mean	None	-	-	**41.21** $\pm$ **1.18**	**41.23** $\pm$ **1.19**	42.25 $\pm$ 3.19	**26.55** $\pm$ **1.19**	**32.28** $\pm$ **0.20**
max	LaTeX2Text	-	-	38.33 $\pm$ 1.57	38.38 $\pm$ 1.57	46.09 $\pm$ 0.77	21.64 $\pm$ 1.60	29.45 $\pm$ 1.41
mean	LaTeX2Text	-	-	34.53 $\pm$ 11.02	34.64 $\pm$ 11.13	**47.02** $\pm$ **20.70**	18.20 $\pm$ 8.10	26.24 $\pm$ 11.64

Table 1: Entity and relation extraction scores for 4 different models on both the development and the test set. NER metrics strict and exact were not produced by the test set evaluation script on the competition site and the test set is not publicly available at the time of writing.

text, via Huggingface's Transformers library (Wolf et al., 2020). For LaTeX preprocessing (see section 3.1) we use Pylatexenc[3]. As our optimizer, we use AdamW (Loshchilov and Hutter, 2019) with learning rates $\in [3\mathrm{e}{-5}, 5\mathrm{e}{-5}, 7\mathrm{e}{-5}]$, a linear warmup of 1 epoch followed by a linear decay to zero, for a total of 60 epochs[4], a batch size of 4, and apply gradient clipping with a max norm of 1. During training, we randomly downsample the amount of candidate spans for soft mention detection to 1000, while ensuring that all labeled spans are included. During training and development set evaluation, we set k, the number of spans to perform relation classification on, to 50, as preliminary experiments showed this value to yield a good compromise between model performance and training time. For test set evaluation we increase k to 400. Training takes approximately 10 hours on a single NVIDIA V100 GPU using mixed precision. We perform early stopping based on the micro F_1 score for relation extraction on the development set. We train each hyperparameter configuration 3 times using different random seeds and report the median and standard deviation for each metric. As a result of the different combinations of preprocessing and mean-/max-pooling, we examine the performance of 4 configurations on the test set. For our evaluation, we report the micro F_1 scores for NER metrics as used in SemEval-2013 Task 9.1 (Segura-Bedmar et al., 2013)[5]. For relation extraction we report *micro* precision, recall and F_1 scores, unless otherwise indicated.

5 Results

5.1 Overview

The results of the 4 model configurations on the test set are reported in table 1. In comparison to the other approaches taking part in SemEval-2022 Task 12, our system ranks in place 3/9 in terms of relation extraction F_1 score.[6]

In general, we find that our model produces predictions with significantly higher precision than recall.

5.2 Impact of Preprocessing

With respect to the preprocessing procedure, we observe no clear performance impact. We conclude that SciBERT appears to cope well with LaTeX code and preprocessing, as described in this paper, is not required.

5.3 Impact of Pooling Procedure

Regarding the pooling procedures we find that mean pooling tends to cause higher variability in the classification performance of the models. For the models trained using mean pooling and preprocessing, 1 of 3 models performed significantly worse than the others, causing the large standard deviation in the results.

5.4 Impact of Domain

In table 2, we show the relation extraction F_1 scores for a model across the 4 different domains covered by the development set paired with the distribution of training data across domains. We observe large performance differences depending on the domain with math and physics showing very high macro

[3]`https://github.com/phfaist/pylatexenc`

[4]The length of one epoch is dictated by the number of training examples, which is 3119.

[5]We use the following implementation: `https://github.com/davidsbatista/NER-Evaluation`

[6]Scores for other metrics are not publicly visible on the leaderboard at the time of writing.

	domain			
	cs	econ	math	physics
% of training corpus	16.63	27.08	12.82	32.08
relation type				
Direct	33.12	21.05	63.82	85.71
Count	-	-	84.62	100.00
Corefer-Symbol	21.05	20.47	91.30	100.00
Corefer-Description	3.51	0.00	76.92	96.00
macro	19.23	13.84	79.17	95.43
micro	25.93	19.49	78.77	88.77

Table 2: F1 scores for relation extraction across different domains and relation types on the development set. cs and econ do not contain any instances of "Count".

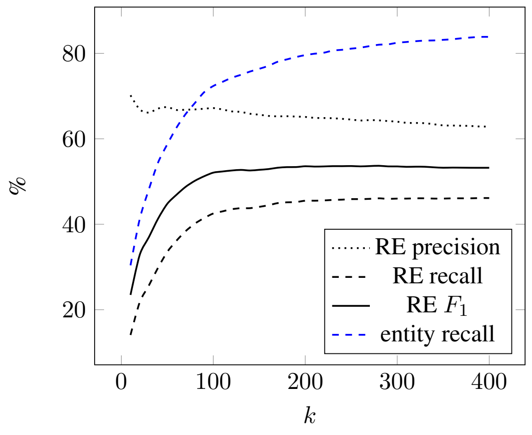


Figure 2: Plot of the impact of increasing values of k on precision, recall, and F1 scores on the development set.

F_1 scores (79.17% / 95.43%) and computer science and economics performing poorly (19.23% / 13.84%). While physics content does represent the majority of training examples, the distribution of domains across training examples does not fully explain the disparity.

5.5 Impact of k

In figure 2, we show the change in relation extraction performance across different values for k. We also include in the plot the percentage of entity spans in the top k ranked spans (entity recall). While the relation extraction performance improves proportional to the entity recall for $k \leq 100$ the improvement slows down for higher k. We hypothesize that this is due to the limiting of $k = 50$ and the candidate span downsampling during training, which prevents the model from seeing some of the more difficult cases. In appendix A.2, we show examples of detected spans.

matching	precision	recall	$F_{1,micro}$
strict	55.85	44.01	49.23
partial	62.87	46.13	53.22

Table 3: Comparison of strict and partial matching requirements with respect to classification scores on the development set.

5.6 Impact of Tokenization

In order to measure the impact of tokenization errors produced by adjusting labels during training, we perform a partial matching of relation labels as follows: For predicted relation triples which are false positives, we accept them as true positives for an annotated instance if the *intersection-over-union* (IOU) scores of both head and tail entities are greater than 67% and the predicted relation type matches the label. In table 3 we show the results of both strict and partial matching for our best model on the development set. We find that the relaxed requirements for span accuracy result in an increase in the F_1 score of 3.99%. We conclude that tokenization errors, while measurable, do not account for the majority of errors of our model.

6 Conclusion

In this paper, we present an end-to-end joint entity and relation extraction approach for linking mathematical symbols to their descriptions in LaTeX documents. Our model appears to be sensitive to the domain of the input documents, achieving high macro F_1 scores of 95.43% and 79.17% for physics and math content, respectively, while achieving macro F_1 scores of only 19.23% and 13.84% for computer science and economics related content. We find that the model's predictions are higher in precision than in recall. We perform a detailed error analysis and identify cross-domain generalization as the most critical problem to tackle in future work.

Acknowledgements

The authors acknowledge support by the state of Baden-Württemberg through bwHPC.

References

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34–45.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

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

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2006–2013. IOS Press.

Markus Eberts and Adrian Ulges. 2021. An end-to-end model for entity-level relation extraction using multi-instance learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 3650–3660. Association for Computational Linguistics.

Michael Färber. 2019. The Microsoft Academic Knowledge Graph: A Linked Data Source with 8 Billion Triples of Scholarly Data. In *Proceedings of the 18th International Semantic Web Conference*, ISWC'19, pages 113–129.

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haïfa Zargayouna, and Thierry Charnois. 2018. SemEval-2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 679–688, New Orleans, Louisiana. Association for Computational Linguistics.

Viet Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Huu Nguyen. 2022. Semeval 2022 task 12: Symlink: Linking mathematical symbols to their descriptions. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *Proceedings of the International Conference on Learning Representations 2019*, page 18.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Dat Quoc Nguyen and Karin Verspoor. 2019. End-to-End Neural Relation Extraction Using Deep Biaffine Attention. In *Advances in Information Retrieval*, Lecture Notes in Computer Science, pages 729–738, Cham. Springer International Publishing.

Nicholas Popovic and Michael Färber. 2022. Few-Shot Document-Level Relation Extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ofer Sabo, Yanai Elazar, Yoav Goldberg, and Ido Dagan. 2021. Revisiting Few-shot Relation Classification: Evaluation Data and Classification Schemes. *Transactions of the Association for Computational Linguistics*, 9:691–706.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823. IEEE Computer Society.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. 2013. SemEval-2013 Task 9 : Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019. Extracting Multiple-Relations in One-Pass with Pre-Trained Transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1371–1377, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara

Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A Large-Scale Document-Level Relation Extraction Dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

A Appendix

A.1 Entity Type Classification Map

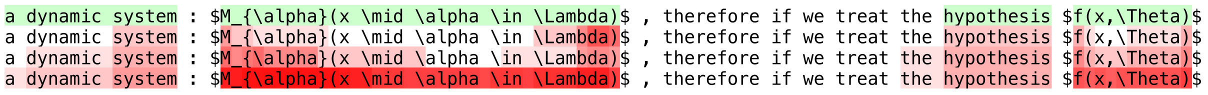
Table 4 shows the classification map used for determining entity types based on relations for SemEval-2022 Task 12.

Relation	**head entity**	**tail entity**
Direct	PRIMARY*	SYMBOL
Count	PRIMARY	SYMBOL
Corefer-Symbol	SYMBOL	SYMBOL
Corefer-Description	PRIMARY	PRIMARY

Table 4: Classification map for entity types based on relations in which the spans participate. *In a postprocessing step, entity types of spans which are the head entity of multiple "Direct" relations are adjusted to "ORDERED".

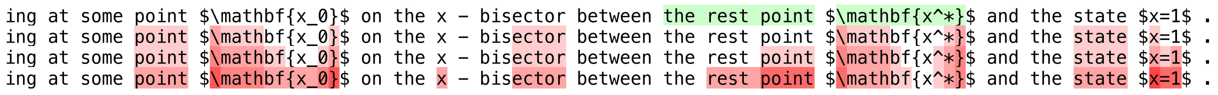
A.2 Examples of Spans Detected for Different Values of k

Examples of spans detected via soft mention detection are shown in figures 3, 4, 5, and 6.



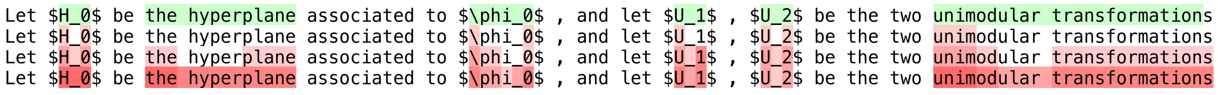
```
a dynamic system : $M_{\alpha}(x \mid \alpha \in \Lambda)$ , therefore if we treat the hypothesis $f(x,\Theta)$
a dynamic system : $M_{\alpha}(x \mid \alpha \in \Lambda)$ , therefore if we treat the hypothesis $f(x,\Theta)$
a dynamic system : $M_{\alpha}(x \mid \alpha \in \Lambda)$ , therefore if we treat the hypothesis $f(x,\Theta)$
a dynamic system : $M_{\alpha}(x \mid \alpha \in \Lambda)$ , therefore if we treat the hypothesis $f(x,\Theta)$
```

Figure 3: An example of spans detected in the domain of computer science. The top row shows ground truth labels in green, while the rows below are spans detected at $k = 50, 100, 150$.



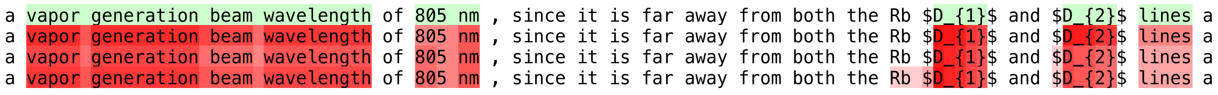
```
ing at some point $\mathbf{x_0}$ on the x - bisector between the rest point $\mathbf{x^*}$ and the state $x=1$ .
ing at some point $\mathbf{x_0}$ on the x - bisector between the rest point $\mathbf{x^*}$ and the state $x=1$ .
ing at some point $\mathbf{x_0}$ on the x - bisector between the rest point $\mathbf{x^*}$ and the state $x=1$ .
ing at some point $\mathbf{x_0}$ on the x - bisector between the rest point $\mathbf{x^*}$ and the state $x=1$ .
```

Figure 4: An example of spans detected in the domain of economics. The top row shows ground truth labels in green, while the rows below are spans detected at $k = 50, 100, 150$.



```
Let $H_0$ be the hyperplane associated to $\phi_0$ , and let $U_1$ , $U_2$ be the two unimodular transformations
Let $H_0$ be the hyperplane associated to $\phi_0$ , and let $U_1$ , $U_2$ be the two unimodular transformations
Let $H_0$ be the hyperplane associated to $\phi_0$ , and let $U_1$ , $U_2$ be the two unimodular transformations
Let $H_0$ be the hyperplane associated to $\phi_0$ , and let $U_1$ , $U_2$ be the two unimodular transformations
```

Figure 5: An example of spans detected in the domain of mathematics. The top row shows ground truth labels in green, while the rows below are spans detected at $k = 50, 100, 150$.



```
a vapor generation beam wavelength of 805 nm , since it is far away from both the Rb $D_{1}$ and $D_{2}$ lines a
a vapor generation beam wavelength of 805 nm , since it is far away from both the Rb $D_{1}$ and $D_{2}$ lines a
a vapor generation beam wavelength of 805 nm , since it is far away from both the Rb $D_{1}$ and $D_{2}$ lines a
a vapor generation beam wavelength of 805 nm , since it is far away from both the Rb $D_{1}$ and $D_{2}$ lines a
```

Figure 6: An example of spans detected in the domain of physics. The top row shows ground truth labels in green, while the rows below are spans detected at $k = 50, 100, 150$.

MaChAmp at SemEval-2022 Tasks 2, 3, 4, 6, 10, 11, and 12: Multi-task Multi-lingual Learning for a Pre-selected Set of Semantic Datasets

Rob van der Goot
IT University of Copenhagen
robv@itu.dk

Abstract

Previous work on multi-task learning in Natural Language Processing (NLP) often incorporated carefully selected tasks as well as carefully tuning of architectures to share information across tasks. Recently, it has shown that for autoregressive language models, a multi-task second pre-training step on a wide variety of NLP tasks leads to a set of parameters that more easily adapt for other NLP tasks. In this paper, we examine whether a similar setup can be used in autoencoder language models using a restricted set of semantically oriented NLP tasks, namely all SemEval 2022 tasks that are annotated at the word, sentence or paragraph level. We first evaluate a multi-task model trained on all SemEval 2022 tasks that contain annotation on the word, sentence or paragraph level (7 tasks, 11 sub-tasks), and then evaluate whether re-finetuning the resulting model for each task specificially leads to further improvements. Our results show that our mono-task baseline, our multi-task model and our re-finetuned multi-task model each outperform the other models for a subset of the tasks. Overall, huge gains can be observed by doing multi-task learning: for three tasks we observe an error reduction of more than 40%.[1]

1 Introduction

Recently, language models have become the de-facto standard in Natural Language Processing (NLP), where we first train a set of parameters on raw data, which are then finetuned on the task at hand. This in itself is a multi-task setup (language-modeling + target task). However, traditionally, multi-task learning was mainly done between multiple NLP tasks with gold annotation. In this setup, many questions arise: not only how to share the information between different tasks, but also when to share and even which tasks to use, as it is non-trivial to decide which auxiliary tasks are beneficial for a certain target task (Ruder, 2017; Crawshaw, 2020). Early work on multi-task learning in NLP often used up to a handful of tasks, carefully curated dataset/task combinations, and carefully tuned how to share the information between these tasks (e.g. Hashimoto et al. (2017); Søgaard and Goldberg (2016)).

A line of recent work has shown that an intermediate step can be used to finetune the language model on a set of NLP tasks, which leads to a model that is more apt for learning other NLP tasks. This is also called Supplementary Training on Intermediate Labeled-data Tasks (STILT), and was introduced by Phang et al. (2018). Phang et al. (2018) train on three classification tasks from the GLUE benchmark (Wang et al., 2018), and then retrain for all GLUE tasks, showing a 1.4 point of improvement over all GLUE tasks. Phang et al. (2020) shows that this positive transfer also holds cross-lingually when using multilingual language models and only doing intermediate training on English tasks. Wang et al. (2019). Similar as with earlier models, it remains an open question for STILT models which tasks transfer well to which tasks (Vu et al., 2020; Pruksachatkun et al., 2020; Chang and Lu, 2021). It should be noted that most work on STILS for autoencoder language models is done on text (i.e. sentence) classification only.

Later work used autoregressive language models, which learn to generate texts (as opposed to the autoencoding models, which learn to predict one token at a time, used by the previously mentioned STILT papers). These language models are commonly used for different types of tasks, namely generation tasks (e.g. question answering, machine translation, summarization), whereas autoencoding models are commonly used for classification and word-level tasks (text classification, pos-tagging, parsing etc.). Recent work has shown that many NLP tasks can be converted to generation tasks, and

[1]code available at: https://bitbucket.org/robvanderg/semeval2022

SemEval Task	Included sub-tasks	Languages	Citation
2: Multilingual Idiomaticity Detection	Idiomaticity detection (1-shot)	EN, PT, GL	Tayyar Madabushi et al. (2022, 2021)
3: PreTENS	1: Binary acceptability	EN, IT, FR	Zamparelli et al. (2022)
	2: Regression acceptability	EN, IT, FR	
4: Patronizing and Condescending Language Detection	1: Binary PCL detection	EN	Pérez-Almendros et al. (2022); Perez Almendros et al. (2020)
	2: Multi-label PCL classification	EN	
6: iSarcasmEval	1: Sarcasm detection	EN, AR	Abu Farha et al. (2022)
	2: Irony-labeling	EN	
	3: Paraphrase sarcasm detection	EN, AR	
10: Structured Sentiment Analysis	Expressions, entities and relations	CA, EN, ES, EU, NO	Barnes et al. (2022)
11: MultiCoNER - Multilingual Complex Named Entity Recognition	Named Entity Recognition	BN, DE, EN, ES, FA, HI, KO, MI, NL, RU, TR, ZH	Malmasi et al. (2022)
12: Symlink	Entities and relations	EN	Dac Lai et al. (2022)

Table 1: Overview of all tasks we participate in. Original source of the data of task 10 are Øvrelid et al. (2020); Barnes et al. (2018); Agerri et al. (2013); Wiebe et al. (2005); Toprak et al. (2010).

can then directly be used to (re-)train an autoregressive language model in a multi-task setup (Aribandi et al., 2022; Sanh et al., 2022). In this setup it is easier to exploit a variety of task-types and a much higher amount of datasets (~50-100 datasets) is used compared to previous work.

In this paper, we set out to examine whether we can obtain performance improvements with multi-task learning and re-finetuning after multi-task learning (i.e. STILT) for a pre-defined set of semantically focused NLP tasks. More precisely, we will use the pre-defined set of SemEval 2022 tasks, and train a multi-task model for all text-based SemEval tasks that include annotation on the word, sentence or paragraph level.[2] We compare a strong single task baseline to a default multi-task learning model, where the encoder is shared, each task has its own decoder, and training is done on all tasks simultaneously (shuffled batches). Finally, we use the parameters from the multi-task model to train a task-specific model for each task again. We seek to answer the following research question:

- Can we exploit a pre-selected combination of NLP tasks in a multi-task setup to improve the ability of an autoencoder language model to learn NLP tasks?

To the best of our knowledge, we are the first to participate in more than 2 SemEval tasks simultaneously, by participating in 7 tasks and a total of 11 tasks including sub-tasks. In our multi-task model, we model a total of 19 tasks if we train on the full data from the tasks (some tasks are modeled as multi-task by themselves), and 54 tasks if we separate them by language or dataset. We will release the finetuned multi-task language models on the hugginface hub (Wolf et al., 2020) for future use, which we dub: Sem-mmmBERT (SemEval MaChAmp multi-task multi-lingual BERT)[3] based on mBERT (Devlin et al., 2019) and Sem-RemmmBERT (SemEval Rebalanced MaChAmp multi-task multi-lingual BERT)[4] based on RemBERT (Chung et al., 2021).

2 Datasets

An overview of the datasets for the tasks included in our setup is shown in Table 1. For task 2, the regression task has no gold training data, so it was left out. Furthermore, we did not participate in any constrained tracks, as we are mainly interested in setups where we also trained on other data. The languages used in the tasks have some overlap, but also some unique languages. English is present in all tasks.

[2]document level annotation is excluded, as it is non-trivial to model in current autoencoder language models

[3]https://huggingface.co/robvanderg/Sem-mmmBERT

[4]https://huggingface.co/robvanderg/Sem-RemmmBERT

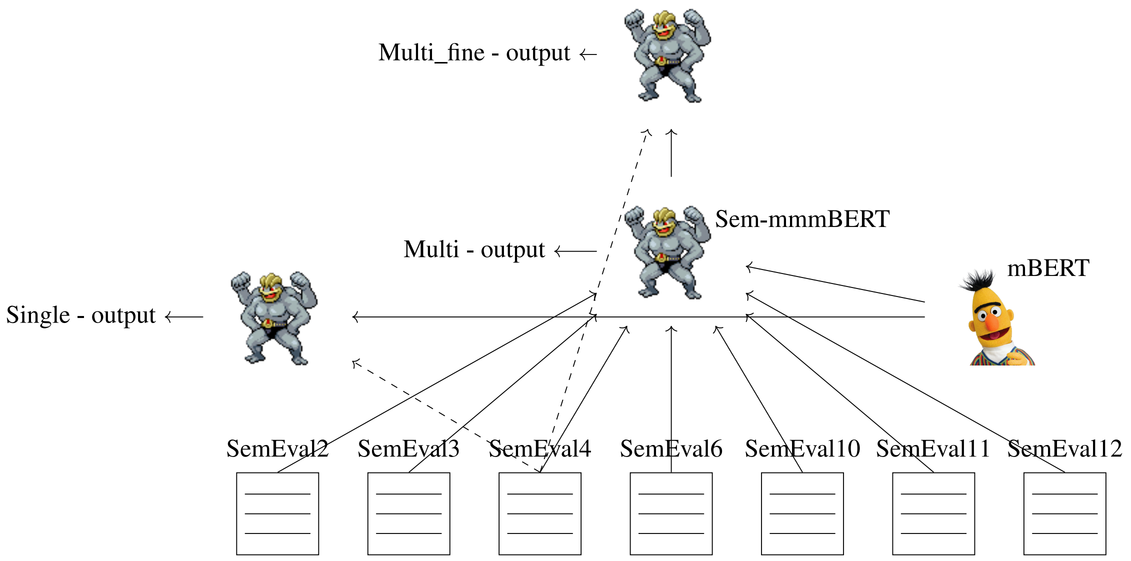


Figure 1: Schematical overview of our proposed multi-task models and the mono-dataset baseline. Sem-mmmBERT is the BERT model which can also be useful for other downstream tasks, and thus the one we release on the huggingface hub. In this example we show the usage of data when training models for task 4 (dashed arrows). For illustrational purposes we left out the sub-tasks in this figure. The boxes with the lines represent annotated data, and = a trained MaChAmp model.

Task	MaChAmp task-type	#words	#sents	#sents smoothed
2-a1	classification	10,199	139	2,742
3-1	classification	99,044	11,669	25,131
3-2	regression	4,761	785	6,518
4-1	classification	399,376	8,369	21,283
4-2	classification	135,750	2,202	10,917
6-a	classification	83,266	5,254	16,863
6-b	classification*6	12,183	691	6,115
6-c	classification	29,242	1,287	8,346
10	seq seq seq	1,109,260	58,799	56,413
11	seq_bio	2,768,898	171,300	96,288
12	seq seq	944,176	3,120	12,994

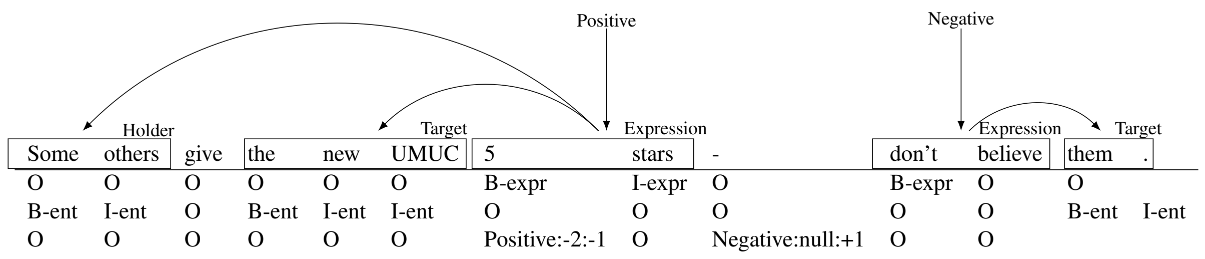
Table 2: The task-types used within machamp for each of the (sub-)tasks, and the data size before and after smoothing.

Table 2 reports the sizes of the datasets, we see that there is a large variety. We attempt to overcome this with dataset smoothing, which is described in more detail in Section 3.8.

3 Model

We implemented all of our models in MaChAmp v0.3beta (van der Goot et al., 2021). MaChAmp is a toolkit that focuses on multi-task learning for NLP task-types based on AllenNLP (Gardner et al., 2018) and the transformers library (Wolf et al., 2020). It supports a wide variety of tasks, and a variety of options for multi-task learning (for within as well as cross-dataset multi-task learning). A typical MaChAmp model consists of a shared encoder (i.e. language model), with multiple decoders on top (one for each task), which all share the same encoder. We use all default hyperparameters of MaChAmp for our experiments, except for the dataset smoothing (Section 3.8). Our general setup is shown in Figure 1. As baseline, we take the data of a single SemEval task, and finetune a MaChAmp model with all default hyperparameters (SINGLE). The first multi-task setup, is a MaChAmp model trained on a combination of all SemEval tasks we consider (MULTI), where each task has its own decoder. Finally we take the hyperparameters from the MULTI model, and refinetune them for a single task at a time (MULTI-FINE).

For the relation extraction tasks (task 10 and 12), we first converted the data to a word-level sequence labeling task, and we contributed a regression task-type in MaChAmp, to be able to tackle task 3-2. For all sub-tasks with multiple languages/datasets, we evaluate also whether learning these in separate decoders is useful (so we split the datasets, and learn them as separate tasks). Below, we describe the choices we made for each of the tasks (the MaChAmp task-types can be found in Table 2), after which we describe our two multi-task setups (Section 3.8 and Section 3.9).



Some	others	give	the	new	UMUC	5	stars	-	don't	believe	them	.
O	O	O	O	O	O	B-expr	I-expr	O	B-expr	O	O	
B-ent	I-ent	O	B-ent	I-ent	I-ent	O	O	O	O	O	B-ent	I-ent
O	O	O	O	O	O	Positive:-2:-1	O	Negative:null:+1	O	O		

Figure 2: Example of conversion of sentiment graph to sequence labeling for task 10, showing a gold annotated sentence (top of line) and the three layers of annotation that the model is supposed to predict (below the line): expression identification, entity identification and relations.

3.1 Task 2

We only participate in the supervised task (one-shot task a), which is a binary task where the goal is to identify whether a sentence contains an idiomatic expression. We use the classification task-type in MaChAmp, and include the multiword expression (MWE) as well as all 3 sentences (target + context) as input. Note that they will automatically be separated by a special separation token in MaChAmp, and their segment ID's will be all 0's for the MWE and third sentence, and 1's for the second and fourth sentence (for language models supporting segment ID's). We use macro-f1 for model picking as well as for the results we report in Section 4.

3.2 Task 3

Subtask 1 is a binary classification task: is a sentence (semantically) acceptable or not. We use the classification task type in MaChAmp and the macro-f1 metric. The data is divided in folds by the organizers, we use fold 1 and 2 as train data, and 3 as dev data. For subtask 1 we use macro-f1 for model picking, and report macro-f1s from the official evaluation script in Section 4.

Subtask 2 is a regression task, where we predict an acceptability score between 1 and 7. We contributed a regression decoder to MaChAmp, which uses a simple linear layer and mean square error loss. We use fold 0 for training and fold 1 as dev data. For subtask 2 we use pearson correlation for model picking.

3.3 Task 4

Subtask 1 concerns a binary classification task: does an utterance contain patronizing or condescending language or not. Subtask 2 identifies one out of 7 sub-categories of patronizing and condescending language. We model both tasks as classification task in MaChAmp, and split the data for both sub-tasks in 80% train and 20% dev data. Following the official metrics, we use accuracy for task 1 and macro-f1 score for task 2 for model picking.

3.4 Task 6

We use an 80:20 split for each of the tasks. Task A is binary sarcasm detection, task B is a multi-class classification task, in which we model each category as a separate task, so that multiple classes can be predicted. Task C is paraphrase detection between sentence-pairs. We follow the official metrics and use macro-f1 for task A and B, and accuracy for C for model picking.

3.5 Task 10

Task 10 is fine-grained sentiment analysis, in which sentiment graphs are predicted. Each opinion is annotated as a tuple consisting of: an expression, which has a polarity (positive/negative) a link to the target, and potentially a link to a holder/source (the person expressing the sentiment). Inspired by Ramponi et al. (2020), we convert this task to three sequence labeling tasks (see also Figure 2). The first task is expression identifiction, which we model as BIO encoded spans. It should be noted that the spans can overlap. The second task is the identification of the source and targets, which are also encoded as BIO spans, which can also overlap. For each token that is the beginning of an expression (B-label), we include a label describing the relations (the third task), which are triples containing of: polarity, link to holder, link to target. The links to targets are simply counts of the directions to the next identified entities (i.e. +1 for the next identified entity), similar to the relative POS strategy of Strzyz et al. (2019), and the relation extraction implementation of Ramponi et al. (2020). There can be multiple relations for a given expression. We concatenate the overlapping labels (which all three sub-tasks have) and model these three tasks in MaChAmp as sequence labeling task. We compared this against using the "multiseq" task type, which can output multiple labels per token. However, performance was better when simply doing sequence labeling, in contrast to Ramponi et al. (2020). Perhaps tuning the threshold of the prediction confidence to include labels could lead to better results (we used the default of 0.5), which we leave for future work. After prediction, we convert the data back to the official json format.

For model picking, we take the average over the accuracies of the concatenated labels, in Section 4 we report the official metric, sentiment graph F1 (Barnes et al., 2022).

3.6 Task 11

Task 11 is multi-lingual named entity recognition. We compared running with and without a CRF-layer, and found that the CRF layer is beneficial. We use span-f1; the implementation of AllenNLP for model picking, and the output of the `conlleval.pl` script for results reported in Section 4, because there is no official evaluation script available

3.7 Task 12

Task 12 is the linking of mathematical symbols, which consists of two steps: 1) detect mathematical symbols 2) identify links between them, which are directed and labeled. We use a similar strategy as we used in task 10, where we convert the task to sequence labeling. In contrast to task 10, the data in task 12 is not pre-tokenized, and some of the spans do not allign with the whitespaces. We tokenize with the `_is_punctuation` function from the transformers library (Wolf et al., 2020) to circumvent this, and save where it splits so that we can undo it after prediction. Similar as for task 10, a token can have multiple labels, we attempt to model this with the "multiseq" task-type in MaChAmp, which can predict multiple labels, but obtain better results by concatenating the labels and predict them as one label per token. We used accuracy for both tasks, as the official metric was not released. The results reported in Section 4 are the average of these two sub-tasks.

3.8 MULTI

We compare the single task baselines to models where we exploit multi-task learning (see also Figure 1). In the first setup, we finetune MaChAmp on all tasks simultaneously, for which we enable the multinomial smoothing in MaChAmp with $\alpha = 0.5$, so that the distribution between tasks becomes more similar (see also Table 2. Note that some SemEval tasks consist of multiple sub-tasks, and some single tasks are modeled as multiple tasks in MaChAmp, we have a total of 19 tasks in the final setting. We evaluate the output of each decoder/task separately for this model.

Task	COMBINED	SEPARATE
task2-a1	**66.00**	61.28
task3-1	**66.77**	66.71
task3-2	**74.07**	72.14
task4-1	42.59	—
task4-2	25.67	—
task6-a	**31.27**	31.25
task6-b	17.05	—
task6-c	90.74	**91.67**
task10	**35.10**	28.90
task11	**79.86**	79.48
task12	96.06	—

Table 3: Scores (dev) of single-task models with mBERT. SEPARATE means that the data from each language (or dataset for task10) has its own decoder. An empty cell (—) means that the task did not consist of multiple datasets/languages, so SEPARATE equals COMBINED.

3.9 MULTI_FINE

After the multi-task model is trained, we save the parameters of the shared encoder, so that they can be re-used for the next step. Finally, we re-finetune the resulting model for each task seperately again, to see whether the multi-task model constitutes a better initialization than the vanilla language model.

4 Results

For all the tasks where the shared task organizers released an evaluation script, we used the official script for the results reported in this section (for the model-picking we used internal equivalent metrics, see Section 3 for the details per task); for task11 we used `conlleval.pl`, and for task12 we used an average of accuracy over our converted data.

Task	SINGLE	MULTI	MULTI_FINE
task2-a1	**66.00**	64.67	63.64
task3-1	66.77	66.82	**66.87**
task3-2	74.07	84.82	**85.37**
task4-1	42.59	52.81	**80.00**
task4-2	25.67	**28.86**	27.65
task6-a	31.27	**59.75**	43.08
task6-b	17.05	**21.64**	19.22
task6-c	90.74	89.20	**95.37**
task10	35.10	**37.70**	25.70
task11	**79.86**	75.73	79.52
task12	**96.06**	95.10	95.31
avg.	56.83	61.55	**61.98**

Table 4: Scores of multi-task settings versus the single task baselines for mBERT.

	MULTI	MULTI_FINE
task2-a1	**78.79**	67.38
task3-1	66.85	**66.86**
task3-2	85.41	**85.80**
task4-1	65.57	**71.07**
task4-2	28.77	**29.54**
task6-a	**69.74**	51.66
task6-b	**29.82**	18.77
task6-c	**96.30**	91.05
task10	**47.70**	45.60
task11	80.45	**82.94**
task12	95.67	**96.33**
avg.	**67.73**	64.27

Table 5: Scores of multi-task settings for RemBERT.

4.1 Single task results

On the single task level, we compared for all datasets consisting of multiple languages or sub-datasets whether it is useful to train them as a single task (with one decoder: COMBINED), or as separate tasks (with N decoders: SEPARATE). For computational efficiency, these tests are only done with mBERT.

Table 3 shows that modeling the languages in separate decoders is only beneficial for task 6 c. We hypothesize that this is because this dataset only contains two languages (English and Arabic), which are relatively distant, so sharing the decoder leads to performance drops. For all further experiments, we will use the COMBINED setup.

4.2 Multi-task results

We first evaluate the results with mBERT, as we also have the single-task results with mBERT (due to computational constraints we do not have them for RemBERT). Table 4 shows the scores for the single-task (SINGLE) baseline, the multi-task model (MULTI), and the intermediate multi-task with finetuning per task setup (MULTI_FINE). Interestingly, each of the three models perform best for 3 or 4 different tasks, and it is thus highly dependent on the task which setup is most beneficial. Differences between scores can be huge though, and when looking at the averages it is clear that the multi-task setups are beneficial over single task models and competetive to each other. The smallest and largest datasets (Table 2) score best with the single task model, as well as task 12, which can be considered an outlier. The multi-task setup without additional finetuning (MULTI) seems to be mostly beneficial for classification tasks. The additional finetuning (MULTI_SEQ) is especially flourishing for task 4-1 and 6c, which are small to medium sized classification tasks, and it is unclear why their trends differ so much compared to task 3-1, 4-2 and 6-a. It should be noted that MULTI is computationally more attractive as well as much smaller to store, as we only need one model for all tasks.

Before the deadline for the SemEval task, we managed to also train the final model with RemBERT (Chung et al., 2021) as language model, however, we do not have the single task baselines. Unfortunately, here only the model with separate decoders for each language/dataset fit on our largest GPU (40gb), so we submitted results with these.

Table 5 shows that also for the RemBERT embeddings, there is no clear single best strategy. The best multi-task strategy sometimes differs compared to the mBERT results (Table 4): task4-2, task11 and task12 differ, where the latter two where the tasks where the single task was the best performing for the mBERT embeddings. On average, the MULTI setup performs more than 3 absolute points higher, but this is mainly due to task 6, which has 3 subtasks (and thus weights heavier in the average).

4.3 Test data

We submitted the results of the mBERT single task baseline and the RemBERT MULTI_FINE setup for the official test evaluations. In Table 6 we show the obtained results and rankings for each task.

Task	Single mBERT	Multi_fine RemBERT	Ranking Ranking
task2-a1	—	66.07	NA
task3-1	78.78	86.42	11/21
task3-2	0.6792	-0.164	17/17 (3/17)
task4-1	0.4172	0.4211	56/78
task4-2	0.0772	0.1546	34/49
task6-a	0.3639	0.3187	31/43 & 12/32
task6-b	0.0919	0.0851	3/22
task6-c	0.2400	0.2250	16/16 & 13/13
task10	0.472	0.501	13/22
task11	0.6027	0.6768	18/26
task12	2.67	7.42	—

Table 6: Official test set results. The — indicate results we could not obtain, and the NA is because we trained on data that was not allowed for that task, so we participated without ranking. For some tasks, multiple rankings are given per sub-track,for task3-2, the single mBERT based model would have ranked 3th.

We note that there are some disrepancies between scores on the test data (Table 6) and the previously reported dev scores (Table 4), these are probably the result of differences in implementation for the metric (when the official code was not released), and could sometimes be the result of uploading the data in the wrong format (e.g. task3-2). Perhaps surprisingly, the single task mBERT model sometimes outperforms RemBERT. This leads to the conclusion that we should not always blindly use the latest, larger language model. Furthermore, we see that for most tasks we rank somewhere in the middle. It should be noted that little to no tuning is done (except for MaChAmp task-type for three of our tasks: 10, 11, 12), as our focus was mostly on comparing our own models to each other and answering our research question. Results can be expected to still increase by selecting the architecture (SINGLE, MULTI, MULTI-FINE), selecting language model per task, tuning the multi-task setup (loss weighing, combination of tasks, smoothing α etc.) or the other hyperparameters (learning rate, scheduler, batch size etc.) of MaChAmp.

5 Conclusion

We have compared three setups in this work: SINGLE: single task finetuning of language models, MULTI: multi-task finetuning of language models, MULTI_FINE: using the output of MULTI and finetuning on single target tasks again. Our setup is both multi-lingual and uses pre-defined set of tasks with a large variety in types of tasks. Our results confirm the findings of recent and concurrent work (Phang et al., 2018; Aghajanyan et al., 2021), showing that for some task combinations, we can benefit from an intermediate task-trained model (MULTI_FINE). However, we also show that all three evaluated setups perform well for certain tasks. We hypothesize that this is an effect of using a pre-defined set of tasks. In our setup the differences between the setups are in some cases extremely large (error reductions larger than 40% compared to the single task baseline have been obtained for three tasks), whereas for some other tasks our single task baseline performed best. This leads to a positive answer to our research question, and the conlusion that intermediate finetuning can be beneficial. However, care should be taken, as our results also show that MULTI_FINE does not outperform MULTI nor SINGLE in all situations, which raises the question: how can we predict whether the intermediate model is better or we need to finetune one more time on the target task?

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. *arXiv preprint arXiv:2101.11038*.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. Ext5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Andrey Kutuzov, Laura Ana Maria Oberländer, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Ting-Yun Chang and Chi-Jen Lu. 2021. Rethinking why intermediate-task fine-tuning works. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 706–713, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2021. Rethinking embedding coupling in pre-trained language models. In *International Conference on Learning Representations*.

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

Viet Dac Lai, Amir Ben Veyseh, Thien Huu Nguyen, and Franck Dernoncourt. 2022. Semeval-2022 task 12: Symlink - linking mathematical symbols to their descriptions. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

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

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Copenhagen, Denmark. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jason Phang, Iacer Calixto, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. English intermediate-task training improves zero-shot cross-lingual transfer too. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 557–575, Suzhou, China. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. 2020. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online. Association for Computational Linguistics.

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

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon

Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. 2019. Viable dependency parsing as sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 717–723, Minneapolis, Minnesota. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

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

Roberto Zamparelli, Absar Chowdhury Shammur, Brunato Dominique, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. SemEval-2022 Task3 (PreTENS): Evaluating neural networks on presuppositional semantic knowledge. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Author Index