Spring 6-4-2023

Sarcasm Detection in English and Arabic Tweets Using Transformer Models

Rishik Lad
Dartmouth College, rishik.lad.23@dartmouth.edu

SARCASM DETECTION IN ENGLISH AND ARABIC TWEETS USING TRANSFORMER MODELS

A Thesis

Submitted to the Faculty

in partial fulfillment of the requirements for the

degree of

Bachelor of Arts

in

Computer Science

by

Rishik Lad

DARTMOUTH COLLEGE

Hanover, New Hampshire

June 2023

Advised By

Professor Soroush Vosoughi

Department of Computer Science

Preface

A version of the work presented in this thesis has previously been published in the Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). I am the first author of the paper [14] in collaboration with Weicheng Ma and Soroush Vosoughi.

Abstract

This thesis describes our approach toward the detection of sarcasm and its various types in English and Arabic Tweets through methods in deep learning. There are five problems we attempted: (1) detection of sarcasm in English Tweets, (2) detection of sarcasm in Arabic Tweets, (3) determining the type of sarcastic speech subcategory for English Tweets, (4) determining which of two semantically equivalent English Tweets is sarcastic, and (5) determining which of two semantically equivalent Arabic Tweets is sarcastic. All tasks were framed as classification problems, and our contributions are threefold: (a) we developed an English binary classifier system with RoBERTa, (b) an Arabic binary classifier with XLM-RoBERTa, and (c) an English multilabel classifier with BERT. Pre-processing steps are taken with labeled input data prior to tokenization, such as extracting and appending verbs/adjectives or representative/significant keywords to the end of an input tweet to help the models better understand and generalize sarcasm detection. We also discuss the results of simple data augmentation techniques to improve the quality of the given training dataset as well as an alternative approach to the question of multilabel sequence classification. Ultimately, our systems place us in the top 14 participants for each of the five tasks in a sarcasm detection competition.

Acknowledgements

I'd like to first express my deepest gratitude to Professor Soroush Vosoughi for welcoming me into his Minds, Machines, and Society research group and granting me the opportunity to work on interesting natural language processing problems with amazing people I am honored to call my colleagues and my friends. I first met Professor Vosoughi through his machine learning course during my sophomore spring. At the time, I had yet to be convinced of a possible interest in computer science, but his infectious energy and passion for teaching were enough to lead me toward a major in computer science, two years of natural language processing research under him, and multiple deep learning publications. His expertise, encouragement, and guidance throughout my journey at Dartmouth have been instrumental in shaping my academic and professional interests for the years ahead. For that, I am truly grateful.

I would also like to extend my heartfelt appreciation to Weicheng Ma, a Ph.D. student advised by Professor Vosoughi. Weicheng offered continuous support and mentorship throughout my journey as an undergraduate researcher. His deep expertise, insightful discussions, and invaluable feedback have played a significant role in my growth as a student and researcher, and his commitment to his work leaves me in awe. I'm deeply appreciative of the guidance he has given me every step of the way.

I am, of course, indebted to Dartmouth and the Department of Computer Science for providing an enriching academic environment that fosters opportunity for its many faculty and undergraduate/graduate students. Resources provided by the College, such as the James O. Freedman Presidential Scholars program, were instrumental in seeding and enabling my interest in computer science research. Moreover, I am grateful for the experience of having learned from some of the most extraordinary professors anywhere, including Professors Tim Pierson and Deeparnab Chakrabarty.

Beyond the faculty, I'd also like to give a shout-out to the friends I made along the way. Special thanks to Brian Wang, without whom I would not be at Dartmouth or a researcher in the first place. Our conversations over the years have brought great clarity to the way I structure and approach different situations in my life, regardless of the domain, and I'm grateful to be friends.

Finally, I want to thank my parents for their unwavering love, support, and encouragement from across the country. Their decision to immigrate in hopes of a better life for me and all the associated sacrifices have been the driving force for much that I do, and I'm filled with gratitude for providing me with the opportunity to pursue my dreams.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Section 1.1 Motivation

As machine learning models become increasingly popular for natural language processing applications, so does the need for these models to be able to recognize sarcasm in online texts. Sarcasm is a form of irony that occurs when there is a discrepancy between the literal and intended meanings of a text or utterance. This discrepancy typically manifests itself in the form of dislike, contempt, or derogation [29]. Since sarcasm is prevalent on social media, online forums, and review websites, it is critical that models attempting to analyze text online be able to detect and understand it. Without this capability, the nature of sarcasm can interfere with the effectiveness of certain types of tasks.

As an example, a popular use case of current natural language processing models is sentiment analysis and opinion-mining to gauge public opinion about events, products, and more [19]. If a model is unable to identify which Tweets or Facebook comments contain sarcasm, this could incorrectly bias the results toward the incorrect sentiment. This has the effect of producing invalid information that could misinform a researcher or company's understanding of how people think about and perceive something [25]. To ensure knowledge about public opinion is high-quality, it's important for machine learning models to correctly identify and decipher the true meaning behind sarcastic texts.

As a second example, consider offensive speech: sarcasm is occasionally employed to conceal the true nature behind derogatory statements [10]. If a model takes each statement for its literal meaning, this could lead to hate speech making it past the filters of social media applications and online forums like Facebook, Reddit, and Twitter. Not being able to catch this kind of speech before it makes its way to an audience could cause considerable emotional damage to readers, while simultaneously damaging the user experience of people on these sites [10].

As such, issues like this strongly motivate the research and development of machine learning systems that are able to ingest large volumes of textual data, particularly from social media, and extract sequences containing sarcasm. Such systems could be used as building blocks for more complex social media language understanding, inference, and generation applications [20]. Before we discuss our approach, it's important we first understand precisely what constitutes sarcasm and its various types.

Section 1.2
What is Sarcasm?

Sarcasm is the use of words to convey a message that is the opposite of its literal meaning. It's generally used in a humorous way to criticize somebody or something, relying on context and linguistic signals to communicate the actual message. In regards to text, this could mean utilizing excessive punctuation marks or emojis to convey sarcasm [24]. Understanding and identifying sarcasm is complicated by the fact that it can be divided into multiple types: general sarcasm, irony, satire, understatement, overstatement, and rhetorical question [2]. Table 1.1 below provides an example of each category:

General sarcasm	"Oh, great. Another meeting. Just what I needed."
Irony	"I think the sushi we had a few hours ago gave me food poisoning - I've never felt healthier!"
Satire	"When I was a boy, I was told that anybody could become President of the United States. Now I'm beginning to believe it."
Understatement	"Oh it's just a Category 4 hurricane, no big deal."
Overstatement	"She left me a three-star review?! My life is over!"
Rhetorical question	"Do you think money grows on trees?"

Table 1.1: Examples of each category of sarcasm.

A sarcastic sentence will often belong to one or multiple categories, which tends to complicate the research question of classifying a sequence into none, one, or more types. It's worth mentioning that people themselves may have a difficult time catching on to sarcasm in text [16] or determining all the different categories a sarcastic text belongs to, which makes our research that much more important to creating safe and responsible online communities, as well as effective natural language processing tools.

As mentioned earlier, sarcasm is also intertwined with other kinds of communication, like hateful or offensive speech. This enlarges the breadth and scope of sarcasm, and naturally the importance of building high-quality machine learning models that can detect it. With this understanding of sarcasm, we're now better primed to understand the problems tackled in our research.

Section 1.3

Problem Formulation

Our research attempts to improve sarcasm detection in English and Arabic Tweets with the use of deep-learning models. Since much of sarcasm detection work has been centered around English sequences, it's unclear whether current models are able to generalize beyond English to other languages [21]. As such, we train and evaluate models on an Arabic dataset of Tweets provided by the sarcasm detection competition. More information regarding the structure of our English and Arabic datasets is provided in later sections.

There are five classification tasks that our research addresses with a variety of deep modeling techniques:

1. **Task A:** Determine whether a given English Tweet is sarcastic or not.
2. **Task B:** Determine whether a given Arabic Tweet is sarcastic or not.
3. **Task C:** Given an English Tweet, determine whether it belongs to zero, one, or multiple subcategories of sarcasm. These are (a) general sarcasm, (b) irony, (c) satire, (d) understatement, (e) overstatement, and (f) rhetorical question.
4. **Task D:** Given a sarcastic English text and its non-sarcastic rephrase, determine which is sarcastic.
5. **Task E:** Given a sarcastic Arabic text and its non-sarcastic rephrase, determine which is sarcastic.

This set of tasks provides an interesting breadth of problems to solve across the dimensions of both language and granularity. Success with Tasks A and B would imply that models are better able to pick up on signals of sarcasm with nothing but the context of the input sequence itself. Furthermore, success with Tasks D and E would suggest that models are able to pick up on subtle linguistic clues that distinguish a sarcastic text from its non-sarcastic rephrase, even if they have the same surface-level meaning. Finally, success with Task C would enable natural language processing applications to capture a higher level of granularity with respect to the nature of a sarcastic text. All of these have important implications for the effectiveness of natural language processing applications across sentiment analysis, information extraction, and conversational artificial intelligence, among others.

Section 1.4 Contributions

Our contributions to the problem of sarcasm detection vary across the specific models created for each task as well as techniques employed to improve performance outside of model architecture. As for task-specific models, we produced:

- An English multilabel classifier with $\text{BERT}_{\text{BASE}}$ for Task C.
- An English binary classifier with $\text{RoBERTa}_{\text{BASE}}$ for Tasks A and D.
- An Arabic binary classifier with $\text{XLM-RoBERTa}_{\text{BASE}}$ for Tasks B and E.

Furthermore, we examine pre-processing steps with labeled input data prior to sequence tokenization, such as extracting and appending verbs/adjectives or significant keywords to the end of an input Tweet to help the models better understand and generalize sarcasm detection. We also discuss the effects of simple data augmentation to improve the quality of the training dataset and find it does not meaningfully improve classification performance for any of the five tasks. Finally, we discuss an alternative approach to the question of multilabel sequence classification, namely creating and training six individual binary classifiers for the six categories of sarcasm.

Section 1.5

Organization of Chapters

The remainder of this thesis is organized as follows:

- **Chapter 2** discusses related work on sarcasm detection.
- **Chapter 3** describes the deep learning models created for the five tasks.
- **Chapter 4** discusses the experimental setup used to evaluate our models.
- **Chapter 5** analyzes and discusses the results of our experiments.
- **Chapter 6** concludes the thesis through a summary, discussion of limitations, and directions for future avenues of investigation.

Chapter 2

Related Work

Over the last decade, researchers have utilized a wide variety of methods and models to detect sarcasm online. Some have employed rule-based approaches that depend on finding explicitly-defined patterns of sarcasm in text, while others have deployed supervised, semi-supervised, and unsupervised machine learning models to identify instances of sarcasm. All have their own tradeoffs in terms of effectiveness, complexity, and scale. While not the most comprehensive review of available literature, this section will lay out interesting and representative examples of research for the following approaches: **rule-based, supervised, semi-supervised, and unsupervised.**

Section 2.1 Rule-Based Detection

Rule-based detection relies on defining explicit patterns of lexical, structural, and semantic features of sarcasm in English [12]. It is among the most explainable types of artificial intelligence since the rules are defined by researchers, but models of this nature are sometimes unable to generalize as well as other machine learning methods. The 2010 paper "Detecting Ironic Intent in Creative Comparisons" focuses on the simile as a framing device to deliver ironic intent (e.g. "sharp as a ball"). In their research, the authors analyze a large English corpus of web-harvested similes and discover that 18% of similes are ironic [27].

Other papers look for different signals, like a contradiction in emotion: Bharti et al. devised and proposed an algorithm that tries to find positive sentiment expressed for a negative situation, and vice versa [5]. They also draw upon the presence of hyperbolic words at the start or end of sentences that, when combined with the previous feature, strongly suggest sarcasm [5]. The sentence "Wow, I'm so glad I failed my test!" is a prime example of sarcasm that expresses positive feelings about a negative outcome, intensified by the word "wow" at the start.

A third paper, "Modelling Sarcasm in Twitter, a Novel Approach" by Barbieri et al., is notable for its sheer number of features and rules. Among other things, they consider the following seven feature sets to model sarcasm [3]:

- **Frequency** (rarity, and gap between rare and common words)
- **Written-Spoken** (written mean, spoken mean, written spoken gap)
- **Intensity** (intensity of adverbs and adjectives)
- **Structure** (length, punctuation, emoticons)
- **Sentiments** (gap between positive and negative terms)
- **Synonyms** (common vs. rare synonyms use)
- **Ambiguity** (measure of possible ambiguities)

This examination of the inner structure of sentences with rules to measure qualities like intensity, imbalance, and unexpectedness help contribute explainable methods of sarcasm detection to the field. Now, let us examine the approach and usefulness of supervised machine learning models in detecting sarcasm.

Section 2.2

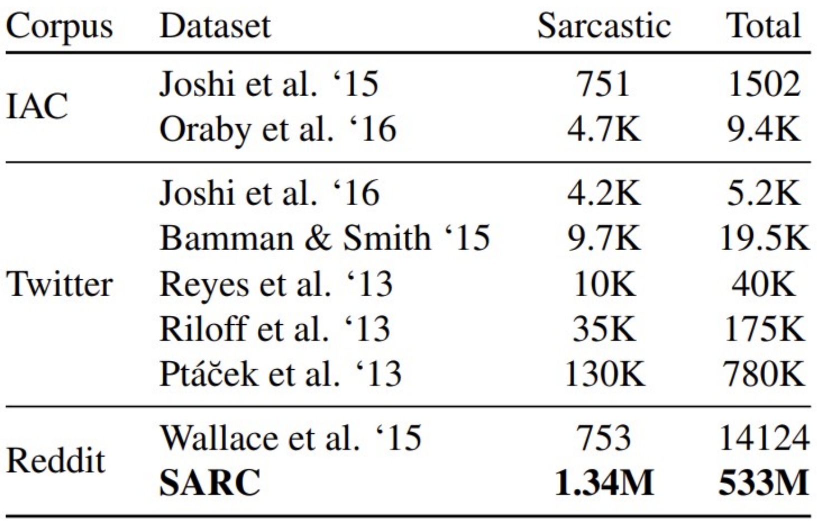
Supervised Learning

Models of rule-based detection depend only on pre-defined patterns and do not need to be trained; machine learning methods, however, need to learn from large amounts of data. A benefit to this modeling paradigm is that machine learning models can incorporate different kinds of input beyond simple text, such as contextual embeddings, semantic relatedness, and other patterns [30]. This can ultimately help the model pick up on features otherwise missed for improving sarcasm detection.

Supervised learning is a subset of machine learning that relies on labeled data for training, meaning that each data point contains features and an associated label. In the context of sarcasm, this might mean that the sequence "You're useful to me like a park bench is to a fish" is given a label of 1 to denote sarcasm, while the non-sarcastic sequence "Sunny weather is expected in the morning, with mild winds in the afternoon" is labeled 0. For problems formulated as binary classification, 1 and 0 are generally used to mark a datapoint as belonging to the positive and negative class, respectively. Common examples of supervised learning models include support vector machines (SVMs), linear and logistic regression (LR), random forests (RF), and Naive Bayes classifier (NBC).

As for supervised learning research in sarcasm detection, Riloff et al. trained an SVM classifier with unigram and bigram features that ultimately achieved 64% precision and 39% recall on a gold-standard dataset of manually annotated Tweets [23]. Another paper, 2013's "A Multidimensional Approach for Detecting Irony in Twitter" by Reyes et al., discusses an approach utilizing both Naive Bayes and decision tree classifiers to identify sarcastic Tweets in a set of 40,000 sequences. With the goal of determining (1) the most representative features of ironic Tweets and (2) the most effective classification strategies to distinguish ironic versus non-ironic, their machine learning systems achieve encouraging results in classification metrics like accuracy, precision, recall, and F-1 [22].

While these papers were still able to put forward meaningful progress, all supervised machine learning systems are limited by the quantity and quality of their labeled training data. The higher the quantity and higher the quality, the better these systems will perform. However, existing datasets for online sarcasm are limited in their quantity (<50K sarcastic utterances) or have questionable labeling quality due to practices like self-annotation or poor quality checking [13], which have negative downstream effects on machine learning models. Figure 2.1 below outlines key details about existing sarcasm datasets that are thematically centered around the Internet Argument Corpus (IAC), Twitter, and Reddit. It's worth noting that with the exception of the Joshi et al. 2016 Twitter dataset, the sarcastic sequences compose no more than 50% of the total, and frequently less.



Corpus	Dataset	Sarcastic	Total
IAC	Joshi et al. ‘15	751	1502
	Oraby et al. ‘16	4.7K	9.4K
Twitter	Joshi et al. ‘16	4.2K	5.2K
	Bamman & Smith ‘15	9.7K	19.5K
	Reyes et al. ‘13	10K	40K
	Riloff et al. ‘13	35K	175K
	Ptáček et al. ‘13	130K	780K
Reddit	Wallace et al. ‘15	753	14124
	SARC	**1.34M**	**533M**

Figure 2.1: Details of existing sarcasm corpora [13].

Section 2.3

Semi-Supervised Learning

Whereas supervised learning only utilizes labeled training data, semi-supervised learning uses a minimum quantity of labeled training data in addition to a much larger amount of unlabeled training data. The idea here is that the labeled datapoints are consumed as they are in supervised learning, namely that the model's prediction ability is tuned by minimizing loss through gradient descent. The unlabeled datapoints then improve this learned ability to consistently and confidently make predictions [4].

Such an approach is employed in "Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon," by Davidov et al., to identify sarcasm in Amazon product reviews and Twitter posts. Using datasets of 5.9 million Tweets and 66 thousand reviews, they initially seed their classification algorithm (similar to K-nearest neighbors) with a small number of labeled utterances and then use the remaining unlabeled datapoints to strengthen their model's classification confidence [8]. To account for varying levels of sarcastic certainty across labeled utterances, a score range of 1 (not sarcastic) through 5 (clear sarcasm) is adopted during the training phase to help the model distinguish between the most and least definitive signals of sarcasm. Evaluation ultimately shows their system achieving F-scores of 0.78 on the product reviews database and 0.83 on the Twitter dataset [8].

Our research extends semi-supervised research by finetuning Transformers (i.e. type of pre-trained large language model) like BERT, RoBERTa, and XLM-RoBERTa on labeled English and Arabic datasets to detect sarcastic Tweets. More information surrounding these models and our approaches is provided in Chapter 3.

Section 2.4

Unsupervised Learning

Unsupervised learning is a type of machine learning that learns patterns entirely from unlabeled data. These types of algorithms are able to engineer features without human intervention to help analyze and cluster new datapoints. It's a unique capability that can uncover hidden patterns within data, avoid the time and labor-intensive costs of data annotation, and create effective features that possibly have little to no interpretability for humans. While this can pose a problem for explainability, unsupervised learning methods are able to outperform their rule-based or supervised learning counterparts (with sufficient data). Examples include k-means clustering, principal component analysis, and hierarchical clustering.

While entirely unsupervised approaches for sarcasm detection are not nearly as popular as the others discussed earlier, there is still innovative research in the space. In 2016, researchers attempted to crack the question of domain-independent irony detection: they used word embeddings to get the ironic orientation of words and combined it with a probabilistic topic model to create the Topic-Irony Model (TIM) [17]. It's an unsupervised system that achieved an 85.81% accuracy on an unbalanced dataset, significantly outperforming existing supervised models at the time [17].

Chapter 3

Models

This chapter provides an overview of the foundational Transformer architecture utilized in our research, as well as a discussion of the architecture and application of BERT, RoBERTa, and XLM-RoBERTa for our five tasks.

Section 3.1

Transformer Architecture

Introduced in 2017 by Vaswani et al. in the foundational paper "Attention is All You Need," a transformer is a type of deep learning model that's become highly effective over the past five years for different needs in natural language processing. The main innovation behind the Transformer is the *self-attention mechanism*, which enables the model to weigh the importance of different tokens in an input text differently during processing and more effectively learn the relationships between input and output sequences [26]. It is inspired by the human thought processing mechanism of paying more attention to certain words and phrases when ingesting information, hence the aptly named "self-attention" mechanism.

With the aid of Figure 3.1, we can now examine the underlying structure of the Transformer. It consists of two types of components, an encoder and a decoder, which are the left and right halves of the visualization, respectively. Each of the encoder and decoder blocks consists of six layers that each contain a self-attention layer and feed-forward neural network.

The encoder first takes an input sequence and processes it by transforming each token into a vector called an embedding. The self-attention layer then receives these embeddings and calculates a weighted sum with them. By determining a weighted sum, the transformer model is able to focus on different parts of the input as is appropriate for the task at hand. This is akin to the idea of "paying more attention" to certain words and phrases in a sentence. The output produced through this self-attention mechanism is finally given to a feed-forward neural network that applies a non-linear transformation to produce a final output of dimension 512, which is the upper limit of the sequence length as allowed by the architecture. In short, the encoder maps a sequence of input tokens $(x_1, x_2, x_3, ..., x_n)$ to an embedding $(z_1, z_2, z_3, ..., z_n)$ that holds all the learned information of the input [28].

The decoder component of the transformer is structurally similar to the encoder, but it contains an additional self-attention layer that takes the encoder's output as input [28]. This is important because it enables the decoder to use context from the input text as it is creating the output sequence. Using the embedding $(z_1, z_2, z_3, ..., z_n)$ outputted from the encoder, the decoder auto-regressively generates the output sequence $(y_1, y_2, y_3, ..., y_m)$ one token at a time [28].

It's worth noting that the multi-head self-attention mechanism in each encoder/decoder layer computes the attention heads in parallel [26], which is more computationally efficient and improves model performance by identifying and capturing different types of dependencies. In particular, the Transformer does a better job than other models, like bi-directional long short-term memory networks (LSTMs), in extracting long-term dependencies.

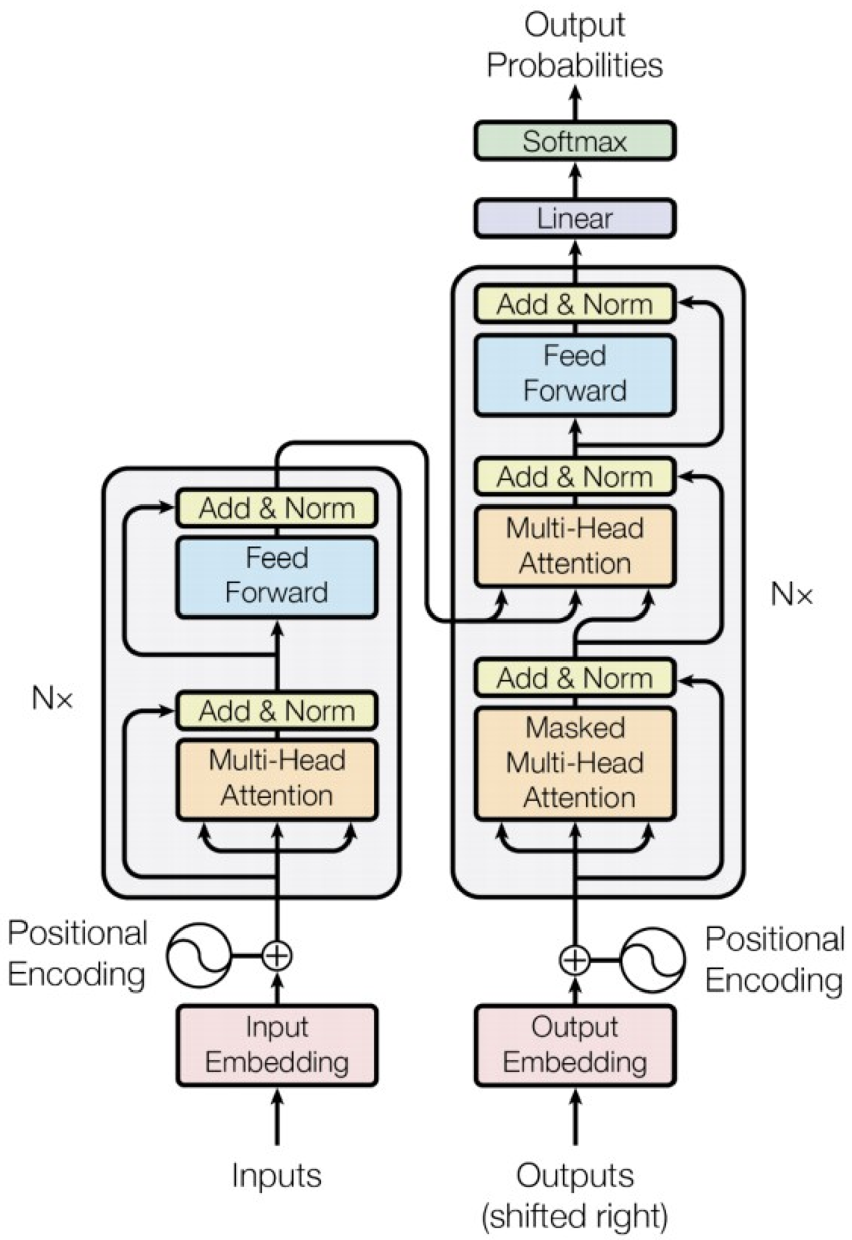


Figure 3.1: The Transformer architecture [26].

Section 3.2

BERT for Multilabel Classification

3.2.1. BERT Overview

Introduced in 2018 by Devlin et al., BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers) is a pre-trained large language model (LLM) that's based on the Transformer architecture [9]. In recent years, BERT has achieved or surpassed state-of-the-art results on a large number of natural language processing tasks, including question answering, text summarizing, named entity recognition, and text classification. Its unique ability to perform at a high level has rendered it an especially popular deep-learning model among researchers and casual users for all kinds of applications. It's worth noting that there are two original model implementations [9]:

1. $\textbf{BERT}_{\textbf{BASE}}$ containing 12 encoder blocks with 12 bidirectional self-attention heads that amount to 110 million parameters
2. $\textbf{BERT}_{\textbf{LARGE}}$ containing 24 encoder blocks with 16 bidirectional self-attention heads that amount to 340 million parameters

The massive popularity of BERT has since inspired a significant number of BERT-based variants, including the RoBERTa and XLM-RoBERTa models that we utilized for the binary classification of English and Arabic Tweets. Across all of our tasks, we opted for the base model implementations because the larger alternatives proved too computationally expensive.

The differentiating factor for BERT is the model's ability to capture contextual information within a sequence by considering the words on both the left and right sides of a given token *simultaneously* [9]. This approach creates vector representations of input sequences that more comprehensively reflect the underlying meaning of a text as well as the relationships and dependencies among the input's tokens.

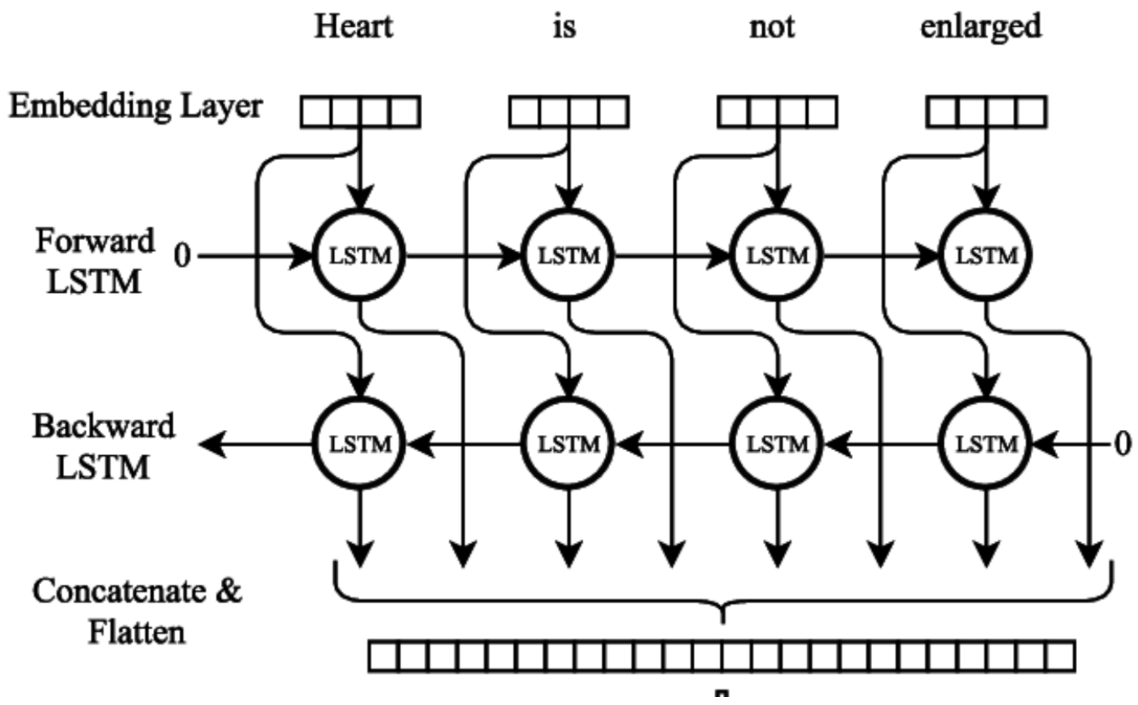


Figure 3.2: Architecture of a bidirectional LSTM [7]

Prior to BERT, researchers relied on bidirectional long short-term memory networks (LSTMs) to capture the relationships and dependencies within a given input. However, LSTMs are only able to process sequences in one direction at a time, either left-to-right (forward) or right-to-left (backward). When ingesting an input text, a bidirectional LSTM will produce a separate forward and backward output, and later "reconcile" the two by concatenating these outputs at each time step [31]. The architecture is pictured in Figure 3.2 and illustrates the process for the input sentence "Heart is not enlarged." While it is still an effective way of capturing context and dependencies, some quality loss occurs during the concatenation and flattening step that is avoided by BERT because it is able to consider the forward and backward contexts at the same time. This is the main reason why BERT is able to train deep bidirectional representations that lend it a more thorough, richer understanding of relationships, dependencies, and context between words in an input sequence.

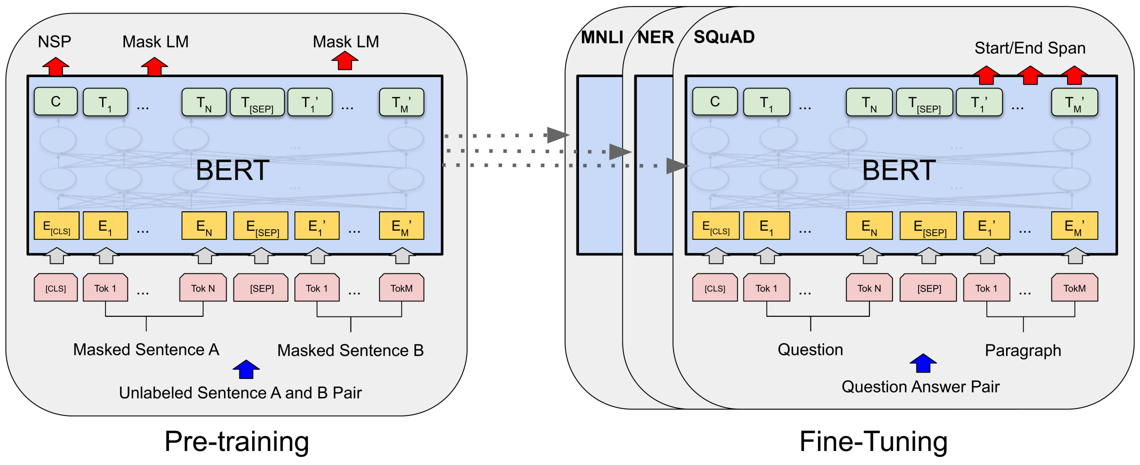


Figure 3.3: BERT pre-training and fine-tuning objectives.

There are two main parts to training a BERT model: it must first be pre-trained on a general corpus of text and then fine-tuned to do a particular task. Figure 3.3 visualizes these processes in greater detail.

Pre-training. The goal of pre-training a large language model such as BERT is to help it learn general-purpose language representations that give it a deep understanding of the structure and semantics of a language. This is done by training the model on massive amounts of unlabeled textual data. BERT, for example, is pre-trained on the English Wikipedia (2.5 billion words) and BookCorpus, a dataset composed of a little more than 11,000 unpublished books (800 million words) [9].

There are two training objectives for BERT's pre-training process: masked language modeling (MLM) and next sentence prediction (NSP). During masked language modeling, BERT is given an input sentence with one or more of its tokens randomly masked; BERT then has to predict what those masked tokens are using the sequence's surrounding context [9]. On the other hand, the next sentence prediction objective involves tasking BERT with determining whether the second sentence in a pair of sentences is an appropriate and valid continuation of the first [9]. Both training objectives can be conducted in an unsupervised manner and without labeled text data. Ultimately, joint training on MLM and NSP helps BERT learn how to predict and generate words or sentences based on context, which also gives BERT a deeper understanding of word-level and sentence-level relationships.

Fine-tuning. The learned representations that the model gains through pre-training can then be fine-tuned for a variety of natural language processing applications. Whereas pre-training is done on a large corpus of general, unlabeled textual data, fine-tuning requires that BERT is trained on task-specific labeled data. The labeled data vary in structure and form, depending on whether the task is question answering, text summarizing, binary/multilabel classification, or something else entirely. More layers are also added on top of BERT as needed, and this whole model is then fine-tuned on the labeled data to achieve the task at hand. We will explain our own fine-tuning approaches in the pages to come.

3.2.2. Multilabel Classification Approach (Task C)

Multilabel classification concerns predicting zero or more mutually non-exclusive class labels for a given sequence. For Task C, we utilize BERT to create a multilabel classifier to categorize an English Tweet into zero, one, or multiple subcategories of sarcasm. A high-level visual of the architecture is provided in Figure 3.4. To reiterate, there are six subcategories within the scope of this task: (a) general sarcasm, (b) irony, (c) satire, (d) understatement, (e) overstatement, and (f) rhetorical question. Examples of each are provided in Table 1.1.

The first step in our classifier-training pipeline was to process and normalize input Tweets. This included replacing hyperlinks, user tags, and emojis with a standardized token (`"@URL"` for hyperlinks and `"@USER"` for user tags, for example). Furthermore, contracted words were separated out to extract the key token. This was to ensure the model did not learn from random, irrelevant noise found in hyperlinks or user tags.

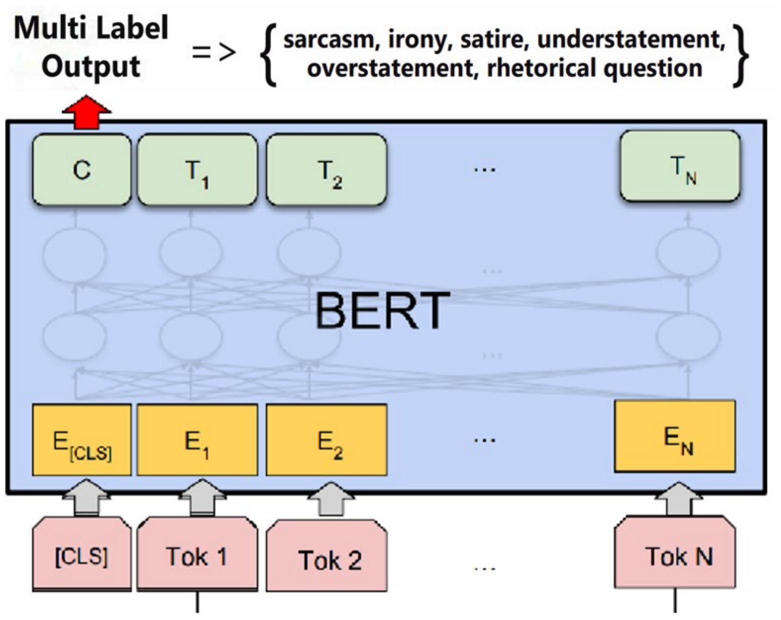


Figure 3.4: BERT architecture modified to reflect multilabel output.

Before feeding the Tweet into BERT's WordPiece tokenizer, however, we utilized an approach to improve the model's understanding of the keywords that might point towards a particular category of sarcasm. A TF-IDF vectorizer was used to extract the 15 most significant and representative keywords across sentences of each category. A sample of such keywords is provided in Table 3.1 for general sarcasm, understatements, and overstatements. Then, for each input training Tweet, all the keywords associated with the categories of sarcasm the Tweet belonged to were strung together to create a second sequence (Sequence B) that was appended to the original input Tweet (Sequence A) and separated with a separator token. Padding tokens were also added to make each input the same length for the BERT model. The maximum length used for this system was 256. Although the longest string of tokens from the training dataset was 111, we set it to 256 for the sake of safety. This ensured that all tensor inputs were set to equal the maximum sequence length used for batched parallelized training.

Sarcasm	"just", "like", "really"
Understatement	"good", "like", "sorta"
Overstatement	"hate", "love", "worst"

Table 3.1: Some examples of extracted, representative keywords.

This meant the ultimate input passed into the tokenizer looked like Table 3.2, where `[CLS]` is the classifier token, `[SEP]` is the separator token, `[PAD]` is the padding token, `Sequence-A` contains the original input sequence, and `Sequence-B` is a concatenated string of the 15 most representative keywords across each of the types of sarcasm applicable to the input. For example, if a Tweet were both an understatement and a rhetorical question, then `Seq-B` would be 30 tokens: a concatenation of the 15 most significant keywords across all understatement sentences in the training dataset, and the 15 keywords across all rhetorical questions. Similarly, a Tweet belonging to 3 categories would have a `Sequence-B` length of 45 tokens, and so on. This approach was used to help the model better seek out key phrases and words that might indicate a Tweet's sarcastic categorization.

Input	`[CLS] Sequence-A [SEP] Sequence-B [PAD]...[PAD]`

Table 3.2: Input structure prior to encoding.

The final element of this system was a sequence classification head containing a linear layer with 6 outputs on top of the final hidden-states output. Since we are working on multilabel classification, we used the sigmoid activation function for the final output and the binary cross-entropy loss function.

After training the multilabel classifier and generating six probabilities for a Tweet's likelihood of categorization in each of the six types of sarcasm, the Tweet would be marked as valid for a category if its percentage score was greater than 0.30. This was a relatively low threshold that we felt was necessary to account for the similarities across Tweets belonging to different categories. Furthermore, we needed to compensate for the lack of training data in categories like satire, understatement, and overstatement, which had 25, 10, and 40 training examples, respectively. By setting a lower threshold, we are able to ensure that we are not prohibitively preventing classifying any Tweets as satire, understatement, overstatement, or others. This technique also improved performance compared to the default decision threshold of 0.50.

Section 3.3

RoBERTa for English Binary Classification

3.3.1. RoBERTa Overview

RoBERTa (**R**obustly **O**ptimized **BERT A**pproach) is a variant BERT model that was introduced in 2019 by Liu et al. [15]. While RoBERTa and BERT share the same underlying Transformer architecture, RoBERTa is further optimized through a variety of techniques. Here are some key differences between BERT and RoBERTa:

- **Pre-Training Objective:** While the masked language modeling (MLM) task is used for both, next sentence prediction (NSP) is not utilized for RoBERTa [15]. It should be noted that this reduces noise during the pre-training process at the possible expense of RoBERTa's understanding of sentence-level relationships.
- **Volume of Data:** RoBERTa is trained on significantly more unlabeled text than BERT. RoBERTa is trained on the corpora used for BERT as well as CC-News, OpenWebText, and Stories [15]. These are datasets containing millions of English news articles, data used for training GPT-2, and a subset of CommonCrawl data, respectively. In total, the model is trained on 160GB of text.
- **Training Hyperparameters:** BERT is trained for 1M steps with a batch

size of 256, whereas RoBERTa is trained for 500K steps with a significantly larger batch size of 8,000 [15, 9]. Hence, despite fewer total training steps, RoBERTa better learns the finer nuances of English's linguistic structure. This is because training with larger batches improves accuracy for the downstream task as well as improving prediction capability for the masked language modeling objective [15].

The General Language Understanding Evaluation (GLUE) benchmark is a set of nine tasks in natural language processing that encompass classification, summarizing, and inference, among others. It's a popular standard by which existing and new models are measured against. It's shown that RoBERTa matches or exceeds BERT in all of the nine individual tasks in GLUE [15], and it's likely that this state-of-the-art performance is due to the model's larger training corpus, modified pre-training objective, and updated training hyperparameters.

3.3.2. Binary Classification Approach for English Tweets (Tasks A and D)

In our research, we employ RoBERTa to tackle Tasks A and D. As a reminder, Task A is to determine if a given Tweet is sarcastic, and Task D is to identify the sarcastic sequence in a pair of Tweets where one is sarcastic and the other is its non-sarcastic rephrase. Refer to Table 3.3 below for an example pair for Task D. We framed both of these challenges as problems of binary classification. Doing so is intuitive for Task A, though perhaps less so for Task D; our approach for using binary classification here was to evaluate the classifier on both Tweets of the pair separately and return the one with a higher likelihood of containing sarcasm.

Processing each input Tweet first began with changing sarcasm labels from 1 (sarcastic) to 0.8 and 0 (non-sarcastic) to 0.2, although the 0.2 was eventually changed back to 0. This was to account for random noise in the dataset and for training exam-

Sarcastic Tweet	"The villains in super hero films are awfully polite. They always confine their invasions to being above one particular city."
Non-Sarcastic Rephrase	"I would say that the invasions that take place are unrealistic and are only shown in this way for plot purposes, as opposed to realism."

Table 3.3: An example pair of Tweets for Task D.

ples that were of lesser quality than others. The next step was input normalization; we borrowed the techniques detailed earlier in Section 3.2.2, consisting of standardizing hyperlinks, user tags, emojis, and expanding out contractions to isolate key tokens.

Before passing each normalized Tweet into RoBERTa's tokenizer, we first extracted all verbs and adjectives from the original sequence (Sequence A) and strung them together with whitespace to create a new string (Sequence B). In turn, those verbs and adjectives were replaced with the `<mask>` token in the original Tweet. This was done in an attempt to help the model better learn the relationship between a sarcastic Tweet and any available verbs or adjectives in it. These two sequences were then joined together and fed into the tokenizer as a single sequence.

Input	`<s> Sequence-A with Masking </s> Sequence-B <pad>...<pad>`

Table 3.4: Sample input for encoding.

Padding tokens were also added to make each input the same length for RoBERTa. As with our multilabel classifier, the maximum length used for this binary classifier system was 256. The finalized input sequence looked like Table 3.4, where `<s>` is the classifier token, `</s>` is the separator token, `<pad>` is the padding token, `Sequence-A` is the input Tweet that now contains `<mask>` tokens where its adjectives and verbs originally were, and `Sequence-B` is a concatenated string of all verbs and adjectives.

The last step of constructing this system involved adding a sequence classification head containing a linear layer on top of the final hidden-states output. A label prediction of 1 denoted a sarcastic Tweet and 0 denoted a non-sarcastic Tweet. For standalone Tweets (Task A), the threshold to pass input as sarcastic was set to 0.40. We observed that, given the quality of the training dataset, changing the sensitivity to a slightly lower decision threshold improved our results marginally. For determining which of two Tweets is sarcastic (Task D), the one with the highest absolute score, regardless of its relation to a threshold, was marked as sarcastic.

Section 3.4
XLM-RoBERTa for Arabic Binary Classification

Before discussing our system development, it's worth mentioning why we chose to pursue the challenge of detecting sarcasm in Tweets from a different language, especially a lower-resource one like Arabic. Right now, English is the primary focus of many new machine learning and natural language processing models. At a surface level, this makes sense because many applications, exchanges, and interfaces are implemented in English. Indeed, English is the world's most common language. However, focusing exclusively on English for NLP tasks can cause researchers in academia and industry to miss out on opportunities to cater to different segments of the population, some of which are hundreds of millions or billions of people large [11]. Furthermore, rapid advancements in English modeling with artificial intelligence before other languages like Croatian, Farsi, and Portuguese, among others, have a chance to catch up, can widen the gap in performance for downstream NLP tasks in English versus other languages [21]. As such, there is a demonstrated need for academia to extend its research into other languages and build natural language processing systems that are increasingly global in nature.

3.4.1. XLM-RoBERTa Overview

XLM-RoBERTa (Cross-Lingual Language Model - RoBERTa) is a multilingual version of RoBERTa that was introduced in the 2020 paper "Unsupervised Cross-Lingual Representation Learning at Scale" by Conneau et al [6]. Since much of XLM-RoBERTa is shared with its monolingual cousin, we will skip its architecture and training hyperparameters. However, it's important to note that the pre-training data is different. XLM-RoBERTa is pre-trained on 2.5TB of filtered CommonCrawl data containing text from 100 languages [6]. RoBERTa, on the other hand, is pre-trained on 160GB of English text from sources including Wikipedia. Figure 3.5 shows the size difference, in gigabytes, between CommonCrawl and Wikipedia data across a plethora of languages. CommonCrawl contains more data for each by significant margins, often doubling or tripling that of Wikipedia. This difference in volume and linguistic breadth of data gives XLM-RoBERTa a good understanding of language-agnostic representations that can later be finetuned to provide competitive results across both high and low-resource languages. Indeed, XLM-RoBERTa excels in cross-lingual transfer tasks involving low-resource languages, whereas existing alternatives like mBERT (multilingual BERT) tend to perform better with high-resource languages like English [6]. As a result, XLM-RoBERTa has become especially popular for multilingual NLP tasks since it outperforms mBERT in standard performance benchmarks like GLUE.

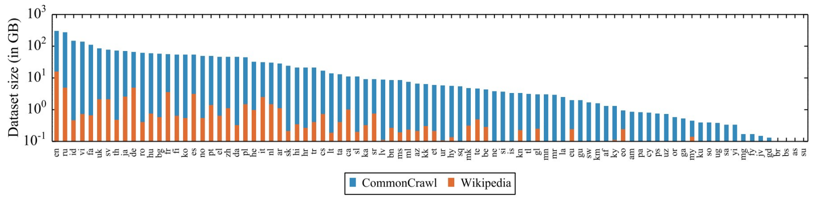
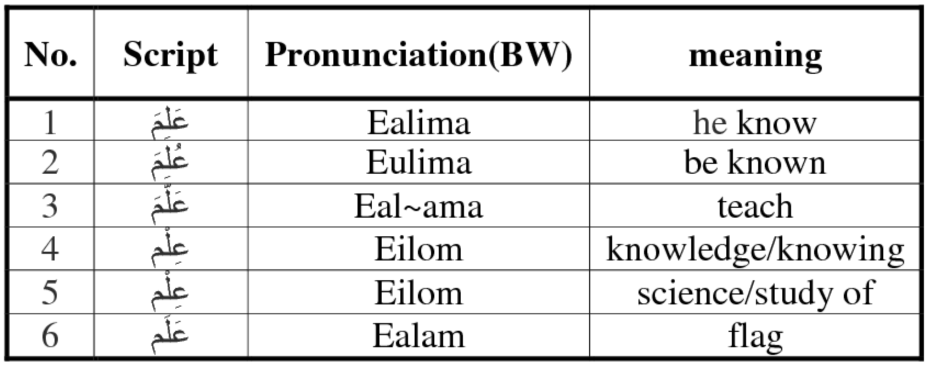


Figure 3.5: Comparison of dataset sizes between CommonCrawl and Wikipedia [6]
.

3.4.2. Binary Classification Approach for Arabic Tweets (Tasks B and E)

We created an Arabic binary classification system with XLM-RoBERTa to address Tasks B and E. These are to determine if an input is sarcastic and to identify the sarcastic sequence in a pair of semantically equivalent Tweets, respectively. Our approach mimics that of our English binary classification model with RoBERTa: we evaluate the Arabic classifier directly on single Tweets for Task B and evaluate the classifier on both Tweets in the pair separately to return the one with a higher likelihood of sarcasm for Task E.

In building our system, we first deal with input pre-processing. We start with changing the sarcasm confidence labels from 1 to 0.8. This is done to account for random noise as well as subpar training data examples that did not encapsulate sarcastic qualities nearly as well as others.



No.	Script	Pronunciation(BW)	meaning
1	عَلِمَ	Ealima	he know
2	عُلِمَ	Eulima	be known
3	عَلَّمَ	Eal~ama	teach
4	عِلْم	Eilom	knowledge/knowing
5	عِلْم	Eilom	science/study of
6	عَلَم	Ealam	flag

Figure 3.6: Diacritics in Arabic can change the meaning of the same word [1].

Note that regular Tweet normalization does not apply to Arabic. Certain qualities in the written form of Arabic, such as diacritization, further complicate this matter. The same word in two different diacritizations can have meanings that are seemingly completely unrelated. Figure 3.6 displays six examples of different diacritics changing the meaning of the same word, which introduces unique difficulty in extracting the true semantic meaning of an Arabic text.

To address this, we relied on CAMeL Tools [18], a Python library designed for the Arabic language to execute de-diacritization and remove any non-essential components from the input texts. Further normalization was also conducted with functions from this specialized library, such as removing orthographic ambiguity.

After processing input Tweets, we again employed the technique of extracting verbs and adjectives to form a second sequence. The extracted verbs and adjectives were replaced with `<mask>` tokens in the original input Tweet, and this was prepended to the second sequence of verbs and adjectives using a separator token. This combined sequence, which is identical in format to Table 3.4, was then fed into the XLM-RoBERTa's SentencePiece tokenizer to be encoded. In this particular case, a specialized part-of-speech tagger was used from CAMeL Tools to identify and extract each Tweet's set of verbs and adjectives. During tokenization, padding tokens were added to the right to make all the tensor inputs of uniform length. Although the longest examined length of any Arabic Tweet was 143, we utilized 256 to account for any unexpected inputs.

The last element of this system was a sequence classification head containing a linear layer that was applied on top of the final hidden-states output with a label of 1 predicting sarcasm and a label of 0 predicting otherwise. For sarcasm detection in standalone Tweets (Task B), those with a score that exceeded our modified decision threshold of 0.40 were marked as sarcastic while others were not. For determining the sarcastic sequence in a pair of semantically equivalent Tweets (Task E), the one with the highest absolute score was marked as sarcastic. Since the task guaranteed one of the two Tweets would be sarcastic, we did not have to consider the magnitude of the model's prediction relative to a decision threshold; we could simply return the more confident Tweet.

Section 3.5

Data Augmentation

To support our efforts for English multilabel classification (Task C), we explored three data augmentation techniques to improve our accuracy and classification metrics. We did not pursue data augmentation techniques for the binary classification tasks.

3.5.1. Simple Sequence Duplication

Our first technique was simply duplicating the satire, understatement, and overstatement categories to double the number of sentences in each of those categories. This involved copying and pasting each sentence back into the dataset to hopefully strengthen the model's understanding of sarcastic keywords, phrases, and qualities. We observed no meaningful improvement in results.

3.5.2. Manual Sequence Generation

The second technique involved the manual generation of sentences to expand the dataset. As mentioned before, a TF-IDF vectorizer was used to extract the 15 most relevant and representative keywords for sentences across each category. See Table 3.1 for some example keywords extracted through this technique. Using basic sentence templates, new training data examples were created with keywords for each satirical category being substituted into various parts of each new training sentence. 30-40 new training examples were created for each of the satire, understatement, and overstatement categories. However, this effort did not yield a meaningful or significant improvement in results.

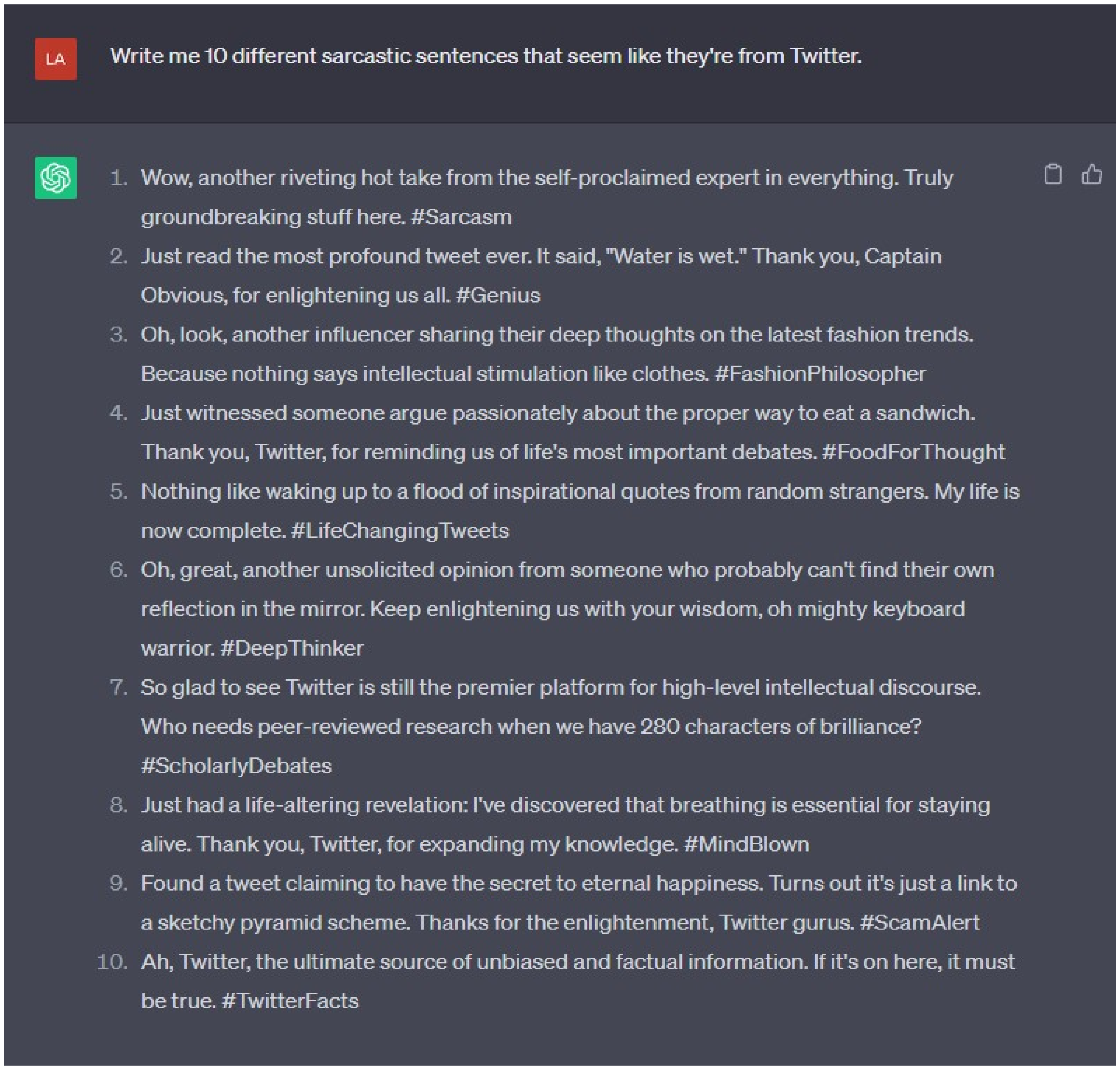
3.5.3. Automated Sequence Generation

A final technique involved utilizing GPT-3 for generating new sentences. A prompt like "Generate 10 sarcastic sentences" or "Create 15 rhetorical questions" was provided to the model. However, it was observed that the produced sentences were particularly repetitive with little variance in structure or style. The difference between most sentences produced by the model was a simple substitution in topic, object, or subject, especially as we asked the model to produce a larger number of new sentences. A lack of training data to properly fine-tune the GPT-3 model was likely an issue here, and this technique was eventually dropped.

I would like to note, however, that the recent introduction of more powerful models like ChatGPT and GPT-4 could make this technique viable again. While outside the direct scope of the research we conducted, I prompted ChatGPT with the following in late May 2023: "Write me 10 different sarcastic sentences that seem like they're from Twitter." The outputted sentences (Figure 3.7) were significantly more diverse than those produced by GPT-3 months ago. While the produced sentences shared some commonalities in tone of voice and structure, such as the hashtag at the end of each sentence, the results were nevertheless promising. It suggests that future research directions in sarcasm detection for low-resource languages could utilize newer and more powerful technology, like ChatGPT or GPT-4, for enhancing datasets through data augmentation. The implications for other downstream natural language processing tasks like question-answering, text summarization, and inference are exciting as well.

3.5.4. Results

Results for these techniques are provided in Table 3.5, with the highest-performing technique in bold for each metric. As observed, while most techniques seemed to perform better than the baseline, the improvements are marginal and unreliable.



Write me 10 different sarcastic sentences that seem like they're from Twitter.

1. Wow, another riveting hot take from the self-proclaimed expert in everything. Truly groundbreaking stuff here. #Sarcasm
2. Just read the most profound tweet ever. It said, "Water is wet." Thank you, Captain Obvious, for enlightening us all. #Genius
3. Oh, look, another influencer sharing their deep thoughts on the latest fashion trends. Because nothing says intellectual stimulation like clothes. #FashionPhilosopher
4. Just witnessed someone argue passionately about the proper way to eat a sandwich. Thank you, Twitter, for reminding us of life's most important debates. #FoodForThought
5. Nothing like waking up to a flood of inspirational quotes from random strangers. My life is now complete. #LifeChangingTweets
6. Oh, great, another unsolicited opinion from someone who probably can't find their own reflection in the mirror. Keep enlightening us with your wisdom, oh mighty keyboard warrior. #DeepThinker
7. So glad to see Twitter is still the premier platform for high-level intellectual discourse. Who needs peer-reviewed research when we have 280 characters of brilliance? #ScholarlyDebates
8. Just had a life-altering revelation: I've discovered that breathing is essential for staying alive. Thank you, Twitter, for expanding my knowledge. #MindBlown
9. Found a tweet claiming to have the secret to eternal happiness. Turns out it's just a link to a sketchy pyramid scheme. Thanks for the enlightenment, Twitter gurus. #ScamAlert
10. Ah, Twitter, the ultimate source of unbiased and factual information. If it's on here, it must be true. #TwitterFacts

Figure 3.7: Sarcastic sentences produced by ChatGPT in late May 2023.

Data Augmented Results	**Scores**		
	Weighted Recall	Weighted Precision	Macro-F1 Score
Baseline (No Augmentation)	0.549	0.091	0.156
Category Duplication	**0.642**	0.089	0.156
Manual Sentence Generation	0.561	**0.093**	**0.159**
GPT-3 Sentence Generation	0.552	0.082	0.151

Table 3.5: Performance of data-augmented systems for multilabel classification.

Section 3.6

Additional Techniques

It is worth mentioning that the implementation of another system was explored to conduct multilabel classification. Specifically, we attempted to create 6 individual binary classifiers (one for each category of sarcasm) with the intent of aggregating results across all binary classifiers to mimic a multilabel classifier's output. This was complicated however by the severe lack of training examples for some categories as well as issues with computational capacity and consumption on Google Colab Pro. This system was eventually dropped in favor of a single multilabel classifier built with BERT, as described earlier.

Chapter 4

Experimental Setup

The following chapter discusses our English and Arabic datasets, evaluation metrics, and model training and finetuning details.

Section 4.1

Datasets

The datasets we worked with were provided by a sarcasm detection competition, specifically *Task 6: iSarcasmEval - Intended Sarcasm Detection in English and Arabic* of the 2022 Semantic Evaluation (SemEval) competition. This organization provided two training data files, one for English and another for Arabic. In each case, the task organizers provided the sarcasm labels for each Tweet themselves. This avoided the need to rely on existing proxies like predefined tags or third-party labelers [2].

4.1.1. English Dataset

Within the English training file, there were nine pieces of information: the Tweet, a `0/1` value for the presence of sarcasm, a non-sarcastic rephrase of that Tweet, and a `0/1` value for each of the various categories of sarcasm (general sarcasm, irony, satire, understatement, overstatement, and rhetorical question). It contained 3466 training examples, of which 866 were sarcastic and the remaining 2600 were not. These 866 examples were further split into multiple labels as follows: 713 for sarcasm, 155 for irony, 25 for satire, 10 for understatement, 40 for overstatement, and 101 for rhetorical questions. The under-resourced nature of categories like satire, understatement, and overstatement introduced challenges for our multilabel classifier system in extracting and understanding the key characteristics belonging to those categories. It is worth noting that data quality is, at times, questionable, with training examples such as *"whoop diddy scoop poop"* adding random noise into an already scarce dataset. Of the available training utterances, we set apart 80% for training the model, 10% for validation, and the remaining 10% for testing our models. The official testing data was not released until the evaluation period of the competition.

4.1.2. Arabic Dataset

Within the Arabic training file, there were four pieces of information: the Tweet, a `0/1` value for the presence of sarcasm, a non-sarcastic rephrase of the Tweet, and a dialect label for that particular Tweet (e.g. Nile, Maghreb). This training file contained 3102 training examples, of which 745 were sarcastic and the remaining 2357 were not. Of the available training utterances, we set apart 80% for training the model, 10% for validation, and the remaining 10% for testing our models. The official testing data was not released until the evaluation period of the competition.

Section 4.2 Evaluation Metrics

For both the English and Arabic binary classification approaches, a confusion matrix was produced to determine accuracy, precision, recall, and F-1 scores. For the multilabel classification task, weighted precision/recall for each category, as well as macro F-1 scores were utilized. The macro F-1 scores were computed by taking the arithmetic mean of all the per-class F1 scores across the six sarcasm categories.

Section 4.3 Implementation Details

All three systems were developed with the PyTorch framework, HuggingFace's transformers library for the BERT, RoBERTa, and XLM-RoBERTa models, and Google Colab Pro using a single Tesla P100-PCIE-16GB GPU. For Tasks A, B, D, and E, the English and Arabic binary classifiers shared the same training hyperparameters: training and validation batch sizes of 16, a maximum sequence length of 256, 6 training epochs, and an AdamW optimizer with a learning rate of 3e-5 and epsilon value of 1e-8. For Task C, the English multilabel classifier's training policy utilized the following hyperparameter values: training and validation batch sizes of 16, a maximum sequence length of 256, 4 training epochs, and an AdamW optimizer with a learning rate of 1e-05 and epsilon value of 1e-12. All other hyperparameter values were set to their defaults according to the HuggingFace implementation. It should also be noted that for all systems, a random seed was set for the sake of reproducibility.

Chapter 5

Results

The following sections describe our model performance across all five tasks, some error analysis, and a case study of Task A.

Section 5.1

Task Performance

Our research placed us in the following ranks out of 50+ teams in the competition:

- **Task A:** 13th place
- **Task B:** 9th place
- **Task C:** 14th place
- **Task D:** 12th place
- **Task E:** 10th place

Table 5.1 displays a tabular summary of all official scores received on each of the five tasks, which vary from accuracy and precision to recall and macro F1 scores. As observed, our best-performing system for binary classification was the English classifier developed for Task A, whereas our worst performing for binary classification was the Arabic classifier developed for Task B. For the multilabel classification task,

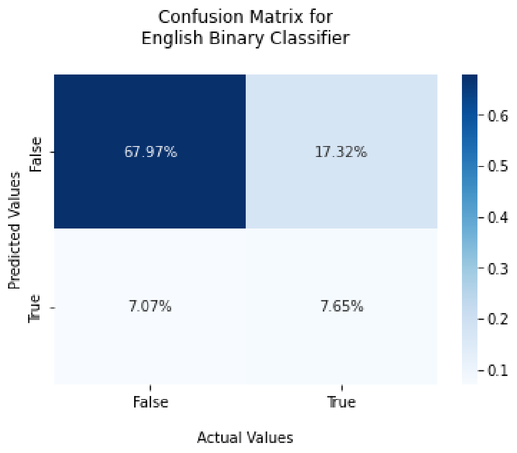
Binary Classification	**Scores**				
	F-1 Sarcastic	F1-Score	Precision	Recall	Accuracy
Task A (English)	0.386	0.635	0.625	0.648	**0.804**
Task B (Arabic)	0.350	0.529	0.581	0.665	0.597
Task D (English)	-	0.659	-	-	0.660
Task E (Arabic)	-	0.679	-	-	0.680

Multilabel	**F-1 Scores**						
	Macro	Sarcasm	Irony	Satire	Under	Over	Rhet-Q
Task C	0.0590	0.2293	0.0202	0.0824	0.0000	0.0077	0.0143

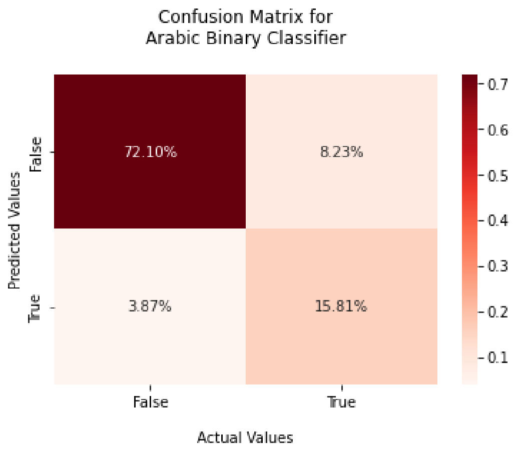
Table 5.1: A tabular summarization of the performance of all three systems across all five subtasks, reporting various metrics including accuracy, precision, recall, and regular/macro F-1 scores.

macro F-1 as well as weighted categorical scores are provided. We also discovered through iterative testing that our data augmentation techniques and alternative modeling approaches did not improve the classification performance of any of our BERT, RoBERTa, and XLM-RoBERTa models.

Confusion matrices displaying our best development metrics are provided in Figures 5.1 and 5.2. These reveal insights into the relatively imbalanced split of the training datasets, creating issues that were further compounded by the overall small number of training examples. It's interesting to note in Figure 5.1 that the false positive rate for the English binary classifier is more than double that of the Arabic binary classifier: 17.32% versus 8.23%. We suspect this may have been due to us modifying the decision thresholds from their default values of 0.50 to 0.40. The Arabic model responded well to this, as the true positive and true negative rates were higher at a threshold of 0.40 than 0.50. Further investigation with the English model would be needed to determine an optimal decision boundary that improves its true positive and true negative percentages.



(a) A confusion matrix of the RoBERTa binary classifier.



(b) A confusion matrix of the XLM-RoBERTa binary classifier.

Figure 5.1: Confusion matrices for binary classification models.

Section 5.2 Error Analysis

This section focuses on some of the factors and approaches that may have led our models to make erroneous predictions across all five tasks. We will do this through a brief case study that explores Task A to better understand our classification scores. To begin, it should be noted that certain inputs in the test dataset for this subtask were sometimes single Tweets like “Followed” or “Pinball!”, while other Tweets were random noise, such as a Tweet containing factual information about a particular day’s time, temperature, humidity, hourly rain, and pressure.

The prevalence of random noise such as the above in the test set can make it somewhat challenging for the model at hand to be able to relate the test input to what it has learned. There’s very little context to learn from in one-word Tweets like the ones mentioned, and this may bias the model towards marking such Tweets as non-sarcastic, when in reality they may be sarcastic (e.g. perhaps the one-word Tweet was a sarcastic remark towards another Tweet). Granted, this is the nature of text in short-form online social forums like Twitter, but it does contribute to a concrete decrease in model performance. Generally, such issues point to the more subjective nature of sarcasm and the various roadblocks involved in its detection with deep learning models.

Furthermore, our technique of masking verbs and adjectives in the original Tweet while simultaneously stringing those verbs and adjectives together into a second sequence to be fed into the tokenizer alongside the Tweet input may have overfit the model towards certain words and phrases from the training dataset. While this may have been helpful in identifying sarcastic Tweets which did include those words, it may also have caused the model to overlook other sarcastic sentences that did not include them in the test dataset.

As such, random noise, poor quality, and certain learning techniques may have been factors in contributing to the scores received by our binary and multilabel classification systems. The same observations apply to Tasks B, C, D, and E. In particular, our technique of extracting and appending the top 15 words for each subcategory a Tweet belongs to may have inadvertently overfit the model to overlook other textual signals that indicate a Tweet's sarcastic categorization in preference for certain words and phrases.

Chapter 6

Conclusion

Section 6.1

Summary

In this paper, we have laid out the motivation behind and problem formulation of detecting sarcasm in English and Arabic Tweets with a variety of deep learning models. We lay the foundation for understanding the foundational Transformer architecture and discuss BERT, RoBERTa, and XLM-RoBERTa. These three large language models are used to structure, train, and evaluate binary and multilabel sequence classification systems for five tasks, including detecting sarcasm in standalone Tweets, identifying the sarcastic sequence in a pair of semantically equivalent Tweets, and determining whether an input belongs to zero, one, or multiple sarcastic subtypes. We found that additional work to augment the training data with duplication of sentences and manually/automatically synthesizing new sarcastic sentences did not improve the results of our models. Furthermore, challenges were observed with the binary and multilabel classifiers in so far as our pre-processing approaches may have biased them toward seeking out a certain set of linguistic cues that prevented them from generalizing as well as they could have.

Section 6.2 Limitations

During our research, we came across a set of limitations that prevented us from exploring the full scope of our research ambitions and possibly hurt our models' classification scores. For one, our research was conducted entirely on Google Colab Pro – despite it being a premium service with improved capabilities, we would regularly time out or run out of RAM or disk space. This prevented us from fully exploring alternative methods to multilabel classification, such as creating six individual binary classifiers for each sarcastic category instead of a single multilabel classifier. These technical limitations also prevented us from utilizing bigger models that contain significantly deeper, richer inner representations of language like $\text{BERT}_{\text{LARGE}}$. With larger models that contain more encoder/decoder blocks, self-attention heads, and parameters, our classification metrics could've improved meaningfully.

Furthermore, the provided training datasets were severely underresourced in a few ways. For example, the English training file only contained around 3,500 labeled utterances, of which a nontrivial portion contained random noise or low-quality sequences. For pursuing approaches based on deep learning, this is a rather small dataset that doesn't lend itself to finetuning our LLMs to the highest quality.

Finally, it's worth mentioning that our approaches for the tasks we chose only address a small portion of the issue of sarcasm detection. We relied on training our models exclusively on textual data – sarcasm is also conveyed very well through audio and images, so a multi-modal approach with a much larger scope that can take into account these different cases would break new ground around sarcasm detection. Other methods, including classical machine learning or ensemble learning, could prove to do the same.

Section 6.3 Future Work

Further investigation could include implementing six binary classifiers instead of a single multilabel sequence classifier for Task C. Given enough training time, data, and resources, it could certainly be the case that aggregating results across specialized binary classifiers provides more concrete results than what has been produced. In particular, this could allow each of the binary classifiers to more deeply learn the unique characteristics, keywords, and structure of the sarcastic sentences it ingests. Alternatively, it could be interesting to use a one vs. all approach for multilabel labeling with, for example, support vector machines (SVMs). In general, using other machine learning paradigms like supervised learning, ensemble learning, or reinforcement learning could reveal new insights into the question of sarcasm detection.

As mentioned earlier, the advent of increasingly powerful technology like ChatGPT and GPT-4 could transform research methods for bolstering low-resource language datasets through data augmentation. Even beyond sarcasm detection, their ability to integrate with, enable, and improve upon other downstream natural language processing tasks is truly exciting. Determining the applications this technology has to other tasks like question-answering, summarization, and inference would be a worthwhile avenue of investigation.

Finally, it would be interesting to see how the results across all three classification systems change with sufficient training data, with perhaps tens of thousands of more valid examples that can allow the models to truly capture the essence of sarcasm across a wide and varied set of training examples.

Bibliography

[1] Gheith Abandah, Alex Graves, Balkees Al-Shagoor, Alaa Arabiyat, Fuad Jamour, and Majid Al-Taee, *Automatic diacritization of arabic text using recurrent neural networks*, International Journal on Document Analysis and Recognition (IJDAR) **18** (2015), 183–197.

[2] Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy, *SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic*, Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Association for Computational Linguistics, 2022.

[3] Francesco Barbieri, Horacio Saggion, and Francesco Ronzano, *Modelling sarcasm in Twitter, a novel approach*, Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (Baltimore, Maryland), Association for Computational Linguistics, June 2014, pp. 50–58.

[4] Avi Bewtra, *Semi-supervised learning: Techniques amp; examples*, Jul 2022.

[5] Drsantosh Bharti, Korra Babu, and Sanjay Jena, *Parsing-based sarcasm sentiment recognition in twitter data*, 08 2015.

[6] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer,

and Veselin Stoyanov, *Unsupervised cross-lingual representation learning at scale*, 2020.

[7] Savelie Cornegruta, Robert Bakewell, Samuel Withey, and Giovanni Montana, *Modelling radiological language with bidirectional long short-term memory networks*, 2016.

[8] Dmitry Davidov, Oren Tsur, and Ari Rappoport, *Semi-supervised recognition of sarcasm in Twitter and Amazon*, Proceedings of the Fourteenth Conference on Computational Natural Language Learning (Uppsala, Sweden), Association for Computational Linguistics, July 2010, pp. 107–116.

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

[10] Simona Frenda, *The role of sarcasm in hate speech. a multilingual perspective*, vol. Vol-2251, 09 2018.

[11] TELUS International, *Natural language processing is key for non-english languages*, Jan 2021.

[12] Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman, *Automatic sarcasm detection: A survey*, 2016.

[13] Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli, *A large self-annotated corpus for sarcasm*, Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018) (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.

[14] Rishik Lad, Weicheng Ma, and Soroush Vosoughi, *Dartmouth at SemEval-2022 task 6: Detection of sarcasm*, Proceedings of the 16th International Workshop

on Semantic Evaluation (SemEval-2022) (Seattle, United States), Association for Computational Linguistics, July 2022, pp. 912–918.

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

[16] Bleau Moores and Vijay Mago, *A survey on automated sarcasm detection on twitter*, 2022.

[17] Debora Nozza, Elisabetta Fersini, and Vincenzina Messina, *Unsupervised irony detection: A probabilistic model with word embeddings*, 11 2016.

[18] Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash, *CAMeL tools: An open source python toolkit for Arabic natural language processing*, Proceedings of the Twelfth Language Resources and Evaluation Conference (Marseille, France), European Language Resources Association, May 2020, pp. 7022–7032 (English).

[19] Alexander Pak and Patrick Paroubek, *Twitter as a corpus for sentiment analysis and opinion mining*, Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10) (Valletta, Malta), European Language Resources Association (ELRA), May 2010.

[20] Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij, *A deeper look into sarcastic tweets using deep convolutional neural networks*, 2017.

[21] Alaa Rahma, Shahira Azab, and Ammar Mohammed, *A comprehensive survey on arabic sarcasm detection: Approaches, challenges and future trends*, IEEE Access **11** (2023), 18261–18280.

[22] Antonio Reyes, Paolo Rosso, and Tony Veale, *A multidimensional approach for detecting irony in twitter*, Language Resources and Evaluation **47** (2013).

[23] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang, *Sarcasm as contrast between a positive sentiment and negative situation*, Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (Seattle, Washington, USA), Association for Computational Linguistics, October 2013, pp. 704–714.

[24] Jayashree Subramanian, Varun Sridharan, Kai Shu, and Huan Liu, *Exploiting emojis for sarcasm detection*, pp. 70–80, 06 2019.

[25] Yik Yang Tan, Chee-Onn Chow, Jeevan Kanesan, Joon Huang Chuah, and YongLiang Lim, *Sentiment analysis and sarcasm detection using deep multi-task learning*, Wireless Personal Communications **129** (2023), no. 3, 2213–2237.

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

[27] Tony Veale and Yanfen Hao, *Detecting ironic intent in creative comparisons*, 01 2010, pp. 765–770.

[28] Patrick von Platen, *Transformer-based encoder-decoder models*, Oct 2020.

[29] Deirdre Wilson, *The pragmatics of verbal irony: Echo or pretence?*, Lingua **116** (2006), no. 10, 1722–1743, Language in Mind: A Tribute to Neil Smith on the Occasion of his Retirement.

[30] Hamed Yaghoobian, Hamid R. Arabnia, and Khaled Rasheed, *Sarcasm detection: A comparative study*, 2021.

[31] Enes Zvornicanin, *Differences between bidirectional and unidirectional lstm*, Nov 2022.