Assessing the Efficacy of the ELECTRA Pre-Trained Language Model for Multi-Class Sarcasm Subcategory Classification

Imogen Jones

MSc Master of Science in Computer Science
The University of Bath
2023

Assessing the Efficacy of the ELECTRA Pre-Trained Language Model for Multi-Class Sarcasm Subcategory Classification

Submitted by: Imogen Jones

Copyright

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of Bachelor of Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Abstract

Sarcasm detection remains a challenging task in the discipline of natural language processing, primarily due to the large levels of nuance, subjectivity, and context-sensitivity in expression of the sentiment. Pre-trained large language models have been employed in a variety of sarcasm detection tasks, including binary sarcasm detection and the classification of sarcastic speech subcategories. However, such models remain compute-hungry solutions and thus there has been a recent trend towards attempting to mitigate this through the creation of more lightweight models - including ELECTRA. This dissertation seeks to assess the efficacy of the ELECTRA pre-trained large language model, known for its computational efficiency and performant results in various natural language processing tasks, for multi-class sarcasm subcategory classification. This research proposes a partial fine-tuning approach to generalise on sarcastic data before the model is applied in several manners to the task while employing feature engineering techniques to remove overlap between hierarchical data categories. Preliminary results yield a macro F1 Score of 0.0787 for 6-class classification and 0.2363 for 3-class classification, indicating potential for further improvement and application within the field.

Contents

1 Introduction 1

2 Literature and Technology Survey 4
- 2.1 Natural Language Processing . . . 4
  - 2.1.1 Sentiment analysis . . . 5
  - 2.1.2 Sarcasm in text . . . 5
- 2.2 Sarcasm detection benchmark . . . 6
  - 2.2.1 Semeval . . . 6
  - 2.2.2 Semeval 2022 Task 6 . . . 6
- 2.3 Sentiment analysis for sarcasm detection in NLP . . . 7
  - 2.3.1 The Transformer . . . 8
- 2.4 LLMs and progression . . . 10
  - 2.4.1 ELECTRA . . . 11
  - 2.4.2 Why ELECTRA? . . . 11

3 Methodology 13
- 3.1 System diagram . . . 13
  - 3.1.1 Multi-label vs multi-class . . . 14
- 3.2 Data . . . 14
  - 3.2.1 Primary project dataset: Semeval 2022 Subtask B . . . 14
  - 3.2.2 Data preprocessing for the Semeval dataset . . . 15
  - 3.2.3 Supplementary project data: the News Headlines Dataset . . . 18
- 3.3 Tokenization for LLMs . . . 19
  - 3.3.1 ELECTRA tokenization . . . 19

4 Implementation and Testing 21
- 4.1 System implementation overview . . . 21
- 4.2 Data . . . 21
  - 4.2.1 Analysis of primary dataset . . . 21
  - 4.2.2 Analysis of secondary dataset . . . 24
- 4.3 ELECTRA Classifier . . . 24
  - 4.3.1 Fine-tuning . . . 25
  - 4.3.2 Scheduling learning rates . . . 26
  - 4.3.3 Evaluation metrics . . . 26
- 4.4 Custom ELECTRA Classifier . . . 27
  - 4.4.1 Learning rates . . . 27
  - 4.4.2 Evaluation Metrics . . . 28

4.4.3 6-class classification versus 3-class classification 29

5 Results 32

5.1 Custom ELECTRA Classifier quantitative results 32

5.2 Custom-ELECTRA Classifiers trend description 35

6 Reflections 37

6.1 Evaluation 37

6.1.1 Model performance 37

6.1.2 Evaluation of overall implementation techniques 38

6.2 Reflections: the Data Problem 38

6.2.1 Consideration of data augmentation techniques 38

6.2.2 Data availability conclusions 39

6.3 Future Work 39

Bibliography 41

A Packages and Extensions 47

A.0.1 Hugging Face 47

A.0.2 Pytorch Lightning 47

A.0.3 TextAttack 48

B Raw Results Output 49

B.0.1 Results Screenshots 49

B.0.2 Loss Graphs 50

C Semeval 2022 Subtask B Full Results 52

List of Figures

2.1 Scaled Dot-Product Attention 8
2.2 Multi-head Attention 9
2.3 Architecture of the Transformer 9

3.1 Fine-tuning ELECTRA for Sarcasm Subcategory Classification 13

B.1 Custom-ELECTRA (6-Class) Classifier Results 49
B.2 ELECTRA-no-FT Results 49
B.3 Aggregate-ELECTRA (3-Class) Results 49
B.4 Custom-ELECTRA Classifier (7-class) Results 50
B.5 Fine-tuned ELECTRA Classifier Results 50
B.6 Custom-ELECTRA (6-Class) Classifier Training Loss 50
B.7 Custom-ELECTRA (6-Class) Classifier Validation Loss 50
B.8 ELECTRA-no-FT Training Loss 51
B.9 ELECTRA-no-FT Validation Loss 51
B.10 Aggregate-ELECTRA (3-Class) Training Loss 51
B.11 Aggregate-ELECTRA (3-Class) Validation Loss 51

C.1 Semeval 2022 Subtask B Results (All Participants) 52

List of Tables

3.1 Overlap between ‘Sarcasm’ class and subcategories: 16
3.2 Overlap between ‘Irony’ class and subcategories: 16
3.3 Semeval training dataset overlap removal 17
3.4 Semeval training dataset values following overlap removal 17
3.5 Semeval test dataset values prior to overlap removal 17
3.6 Semeval test dataset values following overlap removal 18

4.1 Semeval Test Dataset Values Following Overlap Removal 22
4.2 Initial Semeval Training, Validation, and Test Dataset Sample Distribution . 22
4.3 Semeval Training Dataset Positive Values Following Overlap Removal and Dataset Split 23
4.4 Semeval Training Dataset Negative Values Following Overlap Removal and Dataset Split 23
4.5 Total Sample Distribution 23
4.6 News Headlines Dataset Training and Validation samples 24
4.7 Semeval training dataset samples (6-class) Custom-ELECTRA 30
4.8 Aggregate-ELECTRA Semeval Training Dataset Samples Before Augmentation 31
4.9 Aggregate-ELECTRA Semeval Training Dataset Samples After Augmentation 31

5.1 Custom-ELECTRA (6-Class) Classifier Results 33
5.2 Custom-ELECTRA (6-Class) Classifier Results Breakdown 33
5.3 ELECTRA-no-FT results without fine-tuning 33
5.4 ELECTRA-no-FT Results Breakdown 34
5.5 Aggregate-ELECTRA (3-Class) Classifier Results 34
5.6 Aggregate-ELECTRA (3-Class) Classifier Results Breakdown 34
5.7 Custom ELECTRA (7-Class) Classifier Results 35

Acknowledgements

Firstly I would like to thank my supervisor, Nadejda Roubtsova, who has simultaneously pushed me to become the best version of my academic self whilst also supporting and guiding me through even the most bizarre ideas I have had during the course of this project. Secondly, I would like to thank my family for their patience and particularly George, who has supported me even through the hardest aspects of this challenging project. I am wholly grateful for all of your support and encouragement, and would not have made it to the end without you all.

Chapter 1

Introduction

Sentiment analysis, or opinion mining, is part of the Natural Language Processing (NLP) discipline, and focuses primarily on extracting meaningful information from data relating to subjective nuances within text utilising machines Liu (2020). Humans are inherently complex creatures, whose ability to form sentences and express opinions is based on a plethora of circumstances which help to create the a priori cultural and linguistic knowledge which contributes to expressing sentiment. The identification of subjectivity in text is particularly challenging for both humans and machines, with machines often requiring context to identify subjective nuances such as sarcasm Wallace et al. (2014). Machine Learning approaches to sentiment analysis have been common in the field since their resurgence in the mid-2000s, culminating in applications of the now seemingly commonplace pre-trained large language models (LLMs) such as GPT Radford et al. (2019) and BERT Devlin et al. (2019) to a variety of sentiment analysis tasks.

Sarcasm, often described as the lowest form of wit and the greatest sign of intelligence, is defined by the Cambridge Dictionary as "the use of remarks that clearly mean the opposite of what they say, made in order to hurt someone's feelings or to criticise something in a humorous way" Cambridge English Dictionary (2023). Sarcasm is identified as being a communicative form of irony, in which a situation occurs which was the opposite of the intended result Joshi, Bhattacharyya and Carman (2018). Sentiment analysis for sarcasm detection has posed a challenge to NLP practitioners for decades, with early researchers applying stochastic or rule-based approaches in order to identify particular instances of words, syntaxes, or phrases which denote the sentiment Liddy (2001). The absence of sufficient context along with linguistic, facial, or tonal cues denoted in text makes sarcasm detection an interesting challenge for machines which often require explicit programming to tackle the task as a result.

Pre-trained LLMs such as BERT and its numerous iterations have also been applied to sarcasm detection Yaghoobian, Arabnia and Rasheed (2021). The application of LLMs to a challenge such as sarcasm detection appears to be a natural progression for such models, primarily due to their application of the Transformer-based architecture Vaswani et al. (2017) which has been proven to attain superior results in the NLP field, and which has led to these models' impressive ability to derive contextual information from the data provided. The Transformer's novel self-attention mechanism eliminated the requirement for recurrences or convolutions, allowing the model to contextually process entire sequences at each time step during processing Vaswani et al. (2017). However, these pre-trained LLMs are compute-hungry architectures, which often comprise millions or billions of parameters while processing billions of tokens derived from datasets of billions of words. The development of LLMs has therefore mostly been restricted to teams working in industry, such as OpenAI's Radford et al. (2019) and Google's Devlin et al. (2019) due to the large costs associated with powering the models and cloud compute required for their storage.

There has been a recent trend towards developing less compute-hungry LLMs which are capable of being trained on the same amount of data using less or similar training times for smaller compute costs. Such models have included ELECTRA, proposed by Clark et al. (2020) in 2020 to streamline BERT's approach to Masked Language Modelling (MLM), thus using less computational power. While ELECTRA has been applied to general sentiment analysis B et al. (2023) and binary sarcasm detection in English Nigam and Shaheen (2022) and Arabic Abu Farha and Magdy (2021), it has not yet been applied to the task of multi-class sarcasm subcategory classification.

While binary sarcasm detection remains a difficult task, primarily due to a lack of open source labelled corpora for training supervised deep learning models for sarcasm detection specifically, this research aims to focus on the equally challenging task of multi-class classification of subcategories of sarcastic speech. Abu Farha et al. (2022) proposed Semeval 2022 Subtask B, which comprised a multi-label classification task for sarcastic speech subcategories. Du et al. (2022) identified that there exist hierarchical relationships between the subcategories in the dataset provided, and therefore the approach delineated in this project seeks to assess ELECTRA's efficacy when used for the task of multi-class sarcasm subcategory classification. In this approach, each sample can only belong to one distinct class; as opposed to a multi-class approach - in which samples can belong to more than one class. This research thus seeks to address gaps in the existing literature by evaluating the effectiveness of ELECTRA, a more compute-efficient pre-trained LLM, in a distinct, multi-class approach to the task of sarcasm subcategory classification. This project aims to leverage ELECTRA's proficiency in discerning subtle linguistic nuances and context, which has resulted in performance comparable if not superior to more compute-intensive models such as BERT Clark et al. (2020).

The structure of this dissertation is outlined as follows:

Chapter 1: Introduction. This chapter introduces the project, its background, and aims, while encompassing the main problem areas addressed during its course.

Chapter 2: Literature and Technology Survey. This chapter explores the literature and technology surrounding the project, focusing on three key areas: Natural Language Processing, Sarcasm Detection, and LLMs - with particular emphasis on the Transformer and its iterations.

Chapter 3: Methodology. This chapter will endeavour to highlight methods included in the project methodology when designing and implementing the project.

Chapter 4: Implementation and Testing. This chapter will encompass the various iterations and experiments conducted when implementing the project, providing an overview of the implementation at present.

Chapter 5: Results. This chapter will introduce the results obtained during testing of the Custom ELECTRA Classifier module, contextualising these results while considering the quantitative results obtained during the project.

Chapter 6: Reflections. This chapter will seek to evaluate the process implemented to arrive at the results obtained, while providing suggestions for future work in the discipline.

Chapter 2

Literature and Technology Survey

2.1 Natural Language Processing

The Natural Language Processing (NLP) discipline derives primarily from its foundational basis in the 1940s, during which time researchers in the field initially focused on harnessing the capabilities of machines for machine translation Liddy (2001). NLP broadly attempts to utilise machines in order to accomplish "human-like language processing" (Liddy, 2001, p. 1) through various approaches, and provides a nexus between disciplines including linguistics, computer science, and psychology Liddy (2001) Jones (1994). NLP is considered to be a sub-division of the Artificial Intelligence (AI) discipline due to its goal of emulating human-like language processing through the creation of a Natural Language Understanding (NLU) system, which is capable of understanding, translating, and inferring to and from written text Liddy (2001) Rafail and Freitas (2020).

A divergence in NLP research became prevalent in the 1950s and 1960s, during which time researchers took distinct approaches to achieve the above goal of "human-like language processing" (Liddy, 2001, p.1). This division was categorised by either a symbolic, or rule-based approach, in which principles of languages were manually constructed which were subsequently employed to help machines generate syntaxes based on provided input, and a stochastic approach, which attempted to quantify the statistic or probabilistic qualities of language in order to process it Jones (1994) Stanford University (n.d.b) Yaghoobian, Arabnia and Rasheed (2021). A common goal shared by researchers in the NLP development in the latter 20th century was how to provide the machine with sufficient context in order to facilitate learning and language processing Liddy (2001).

Today NLP is a broad discipline, and the ultimate parent of many subsets of learning, including machine translation, information retrieval, information extraction, and sentiment analysis Rafail and Freitas (2020). The increase in computational power, in tandem with the greater availability of labelled language corpora in dataset form, has greatly contributed to the field's progression in the past 20 years Rafail and Freitas (2020). As such, most subsets of the field now focus at least in part on employing neural architectures for NLP, with significant exemplars of state-of-the-art applications of such architectures, including OpenAI's GPT Radford et al. (2019), profoundly increasing the field's profile outside of research.

2.1.1 Sentiment analysis

The sentiment analysis NLP sub-discipline has existed in its current form since at least the early 2000s Liu (2020). The overarching goal of sentiment analysis, also known as opinion mining, is to analyse the opinions, attitudes, or sentiments of an individual expressed in text towards ideas, events, products, services, or other topics Liu (2020). Most generally, the sentiment analysis field was born out of a necessity to help NLP systems to understand subjectivity behind the meaning of textual words expressed by individuals Liu (2020). Although there is a slight semantic difference between the meaning of 'opinion' and that of 'sentiment', the inherent meaning of both is that these things rely on the subjectivity of the expressor Liu (2020).

The practical applications of sentiment analysis today include opinion mining on social media or for marketing purposes, analysis of political and social discourse, psychological and educational research, and customer support Chatterjee, Aggarwal and Maheshwari (2020). Approaches to sentiment analysis have varied, but have broadly included: lexicon-based approach, in which sentiment is determined based on the identification of pre-determined words present in text which correlate with the sentiment being classified; machine learning approach, in which a classifier is trained on labelled data in a supervised environment to identify instances of a particular sentiment based on previously seen examples; and the hybrid approach, which incorporates the two Thelwall (2020). The increase in popularity of deep learning has led to greater capabilities of machine learning systems in the task of sentiment analysis, and has become a dominant approach in the field Chatterjee, Aggarwal and Maheshwari (2020).

2.1.2 Sarcasm in text

The issue of subjectivity is compounded in text, where commonly-used triggers or identifiers humans usually employ to evaluate whether an expression indicates sentiment or opinion, such as tone, inflection, facial expression, or body language, are not present. Sarcastic sentiment is therefore particularly difficult to identify in text, as there can be subjective degrees of what is deemed to be 'clearly' expressing an oppositional subversive sentiment, i.e. there exists an intentional level of nuance between what the speaker says and that which is perceived (or not) by the receiver Joshi, Bhattacharyya and Carman (2016). The presence of sarcasm in text ultimately prevents NLP systems' performance, affecting their ability to complete the prerequisite tasks for which they were designed Yaghoobian, Arabnia and Rasheed (2021).

The sarcasm detection problem remains a challenge for NLP deep and machine learning applications as these systems are as of yet not fully adept at identifying this sentiment, or ironic speech in general Yaghoobian, Arabnia and Rasheed (2021). There is a great deal of not only subjectivity but also subtlety and nuance involved in sarcastic or ironic expressions, often incorporating idiomatic or figurative phraseology - "Isn't that just the cherry on top" - which obfuscate the true meaning intended by the speaker. Such phraseology often also relies heavily on shared cultural knowledge and context which are difficult for models to understand without explicit training Yaghoobian, Arabnia and Rasheed (2021) Joshi, Bhattacharyya and Carman (2016). Models are thus prevented from deriving the true meaning behind these sentences during learning, and often are not provided with sufficient contextual information surrounding an interaction to identify whether instances of sarcasm are present in the text. Context is crucial when training NLP systems for sarcasm detection Wallace et al. (2014), and thus directing a model's focus simply to linguistic cues or semantic rules in the text is insufficient, particularly given the existing constraints surrounding a lack of non-linguistic cues (inflection, facial expression) which are often not delineated in text.

2.2 Sarcasm detection benchmark

2.2.1 Semeval

The International Workshop on Lexical and Computational Semantics and Semantic Evaluation (Semeval) is an annual NLP workshop run in tandem with the Annual Meeting of the Association of Computational Linguistics (ACL). ACL was founded in 1962, and is one of the most significant associations for professionals and researchers working in the field of NLP. Semeval, established in 1998, operates annual research workshops which comprise of several shared research tasks run by Semeval organisers in which members of the NLP community participate. Semeval's stated aim is to "advance the current state-of-the-art" Semeval (n.d.) datasets and processes employed for semantic analysis in NLP.

2.2.2 Semeval 2022 Task 6

The iSarcasmEval Semeval Task 6 (2022) produced by Abu Farha et al. (2022), was created for the identification of "intended sarcasm" (Abu Farha et al., 2022, p. 802) in text. The task contains two languages, English and Arabic, and is split into three sub-tasks: (A) binary "sarcasm detection, (B) sarcasm category classification, and (C) pairwise sarcasm detection given a sarcastic text and its non-sarcastic rephrase" (Abu Farha et al., 2022, p. 805). The two datasets for the task were provided by Abu Farha et al. (2022), and comprised of training and test datasets in English and Arabic for Subtasks A and C, and English language training and test datasets for Subtask B. [1]

iSarcasmEval received participation from over 60 teams internationally, with the majority of the participants utilising various implementations of several neural architectures for the tasks Abu Farha et al. (2022). Data was collated directly from English and Arabic speakers, with further labelling required for Subtask B conducted by trained annotators who were compensated for their work Abu Farha et al. (2022). The majority of the participating teams across all tasks utilised various iterations of Google's Bidirectional Encoder Representations from Transformers (BERT) model, initially proposed by Devlin et al. (2019). BERT is renowned in the NLP community for its significant contributions to the field, due to its novel iteration of the transformer architecture in introducing bi-directionality, in which sequence inputs can be read from both directions (i.e. left to right and right to left) simultaneously, providing the model with greater contextual understanding of sequenced inputs Devlin et al. (2019).

22 teams submitted results for their participation in Subtask B of iSarcasmEval Abu Farha et al. (2022). Although not all participating teams published their results formally, Abu Farha et al. (2022) provide an overview of the approaches used and results obtained by select participants. The iSarcasmEval paper highlights that almost all of the top results across all tasks were obtained by employing iterations of BERT, including MARBERT Abdul-Mageed, Elmadany and Nagoudi (2021) and ALBERT Lan et al. (2019). We are not aware of any teams who employed the ELECTRA, "Efficiently Learning an Encoder that Classifies Token Replacements Accurately" (Clark et al., 2020, p.2) model for sarcasm subcategory classification, a novel iteration of the BERT model which utilises adversarial training through a novel approach to masked language modelling during its pre-training phase. ELECTRA has however attained demonstrably superior results compared to BERT when applied to general sentiment analysis

[1] All datasets for the Semeval task are available at: https://sites.google.com/view/semeval2022-isarcasmeval#h.t53li2ejhrh8

B et al. (2023), and highly performant results when applied to Arabic sentiment and sarcasm detection Abu Farha and Magdy (2021). Further information in respect of ELECTRA can be found in section 2.4.1.

2.3 Sentiment analysis for sarcasm detection in NLP

Research regarding sentiment analysis for sarcasm detection has seen several changes in the past decades. Early attempts to use machines for the task encompassed the rule-based approach commonly found in early NLP practices, in which the machine was used to identify syntactic or linguistic samples of sarcasm or ironic speech which closely correlated with predefined words or phrases defined as sarcastic. Tepperman, Traum and Narayanan (2006) utilised the identification of textual and verbal speech acts and features which denoted sarcasm, but identified that context was required to attain high accuracy.

Machine and deep learning approaches, implemented using supervised, semi-supervised, and unsupervised learning, have been commonplace in the search for successful sarcasm detection systems since the mid-2000s. Machine learning approaches have included the implementation of support vector machines (SVMs) Godara and Aron (2021), neural networks Zhang, Zhang and Fu (2016), and decision trees Bhakuni et al. (2022), often employing algorithms such as naive bayes Bhakuni et al. (2022) and logistic regression Godara, Batra and Aron (2021) in a hybrid approach to improve results.

Godara and Aron (2021) assessed four distinct hybrid machine learning approaches on data collated from Twitter, and identified that an approach comprising of an SVM along with logistic regression and decision trees attained an average F1 Score of 84% across 5 datasets. SVMs remain a powerful tool for text classification tasks, and often produce impressive results and generalisation abilities in supervised settings where there is a wide margin between the hyperplanes of data to be categorised, though they attain less impressive results when there is overlap or noise between the classes in the data Cervantes et al. (2020). Naive bayes classifiers assume that all data features are distinct Chavan et al. (2014), while neural networks such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and long short-term memory networks (LSTMs) - the fundamental blocks on which current LLMs are built - remain popular techniques due to their ability to generate word embeddings and extract sentiment features from the data provided Chatterjee, Aggarwal and Maheshwari (2020).

Many approaches have incorporated datasets collated from social media sources, primarily Twitter Joshi, Bhattacharyya and Carman (2016), and some have focused almost entirely on the effects of emoticons and punctuation for identifying sarcasm Yaghoobian, Arabnia and Rasheed (2021) Subramanian et al. (2019). Wallace et al. (2014) identified that, due to the necessity for humans to rely on contextual information for sarcasm identification, it was also highly important that approaches to the problem consider including some context for the machine to efficiently learn on the data. Pre-trained language models, including BERT Baruah et al. (2020) Nashold (2021), have been a popular choice in attempting to tackle the task, and Yaghoobian, Arabnia and Rasheed (2021) suggested that there would be an increase in employing these models for sarcasm detection in sentiment analysis in the coming years.

2.3.1 The Transformer

The state-of-the-art Transformer neural architecture, initially proposed by Vaswani et al. (2017), has revolutionised the deep learning field, and particularly the NLP discipline Tay et al. (2020). Vaswani et al. (2017) proposed the Transformer's novel self-attention mechanism, which built on the recurrent neural network (RNN's) encoder-decoder architecture. In this architecture, an input sequence (x_1, ..., x_n) is processed in its entirety, encoded to an internal representation (y_1, ..., y_n), and decoded as an output sequence (z_1, ..., z_n) thereby providing the model with the context of the entire sequence Vaswani et al. (2017).

Unlike earlier sequence transduction models such as RNNs or LSTMs, whch process the input sequence x_1, ..., x_n sequentially, the self-attention mechanism allows the model to employ parallel computation over the input sequence in its entirety at every time step and at each model layer Vaswani et al. (2017). The self-attention mechanism however does not inherently account for token order. The Transformer thus applies positional encodings - a fixed function of input sequence position - to the input embeddings. This allows the model to utilise valuable positional information to better capture relationships between tokens based on their "relative or absolute" (Vaswani et al., 2017, p. 5) position in the sequence.

Vaswani et al. (2017) propose Scaled Dot-Product Attention - a measure of similarity between pairs of input tokens. This attention is computed as a weighted sum of the values V based on the similarity (dot product) of the query Q with each key K. For a query Q, keys K, and values V, the Scaled Dot-Product Attention is defined as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \tag{2.1}$$

Where d_k is the dimensionality of queries and keys scaling the dot-product to prevent exponential increases in tandem with increase in dimensionality. The matrix of outputs is calculated as a series of n queries packed into a matrix Q, and n keys and values are packed into matrices K and V (Vaswani et al., 2017, p. 4). The Scaled Dot-Product Attention can formally be visualised as so:

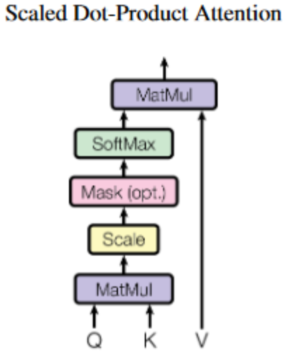


Figure 2.1: Scaled Dot-Product Attention

(Vaswani et al., 2017, p. 4)

Multiple parallel attention layers or heads are used by the Transformer to capture distinct relationships in the input. Each head learns different, linear projections of the input embeddings into Q, K, and V. The output vectors are later mapped to produce an output vector. These parallel attention heads allow the model to focus on separate characteristics within the input Vaswani et al. (2017). The attention heads are "stacked" in the Transformer, creating a model architecture described by (Vaswani et al., 2017, p.4) as "multi-head attention" in which multiple attention layers are parallelised:

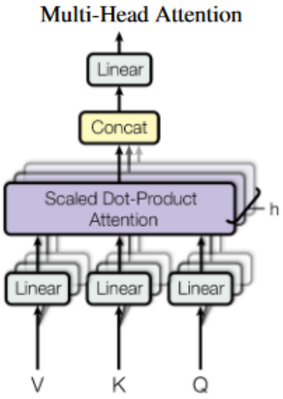


Figure 2.2: Multi-head Attention

(Vaswani et al., 2017, p. 4)

Vaswani et al. (2017) (p. 3) thus define the full Transformer model architecture as follows:

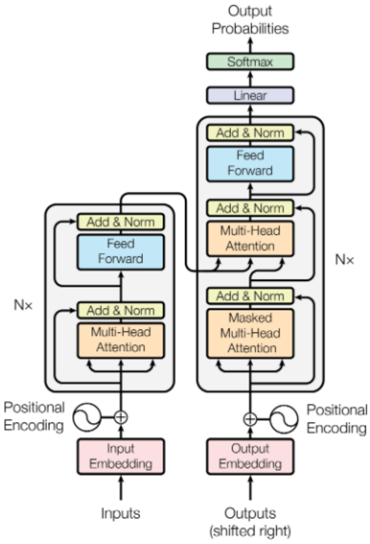


Figure 2.3: Architecture of the Transformer

In NLP, the effects of the Transformer have been formidable, with multiple prominent large language models (LLMs) pre-trained on significant amounts of training tokens derived utilising this architecture. In 2018, Google's Devlin et al. (2019) proposed the BERT architecture, which facilitated bi-directionality of the self-attention mechanism whilst using masked language modelling (MLM) to generate tokens which disrupt input, directing the model's attention to these tokens in order to increase model performance Devlin et al. (2019). BERT and its numerous subsequent iterations have attained state-of-the-art scores on evalution metric benchmarks including GLUE, BLEU, and F1 Score on a variety of NLU tasks, and is widely used in multiple NLP subfields such as machine translation and question answering Devlin et al. (2019).

Open AI's family of Generative Pretrained Transformer (GPT) models initially proposed in 2019 also employ the Transformer architecture by employing a significant increase in the number of parameters and training data used Radford et al. (2019). The autoregressive GPT models, which are pre-trained on massive textual corpora and then fine-tuned on separate downstream tasks to maximise performance, have gained notoriety through the release of ChatGPT, an AI chatbot which operates on the GPT 3.5 and 4.0 foundation models Radford et al. (2019) OpenAI (2022) Brown et al. (2020). ChatGPT has elicited a range of reactions from researchers, consumers, and governments alike due to its successful ability to mimic human-written text, despite some of its shortcomings including hallucination and incorrect assertions Lee (2023). OpenAI researchers have also claimed that their models are able to obtain state-of-the-art performance on multiple few-shot tasks, i.e. on tasks for which the model was not explicitly trained Brown et al. (2020), though Espejel et al. (2023) identified that GPT 4.0 does not perform well in a zero-shot setting in several areas including mathematical problem-solving and common sense reasoning.

2.4 LLMs and progression

As illustrated above, a significant trend towards employing the Transformer architecture in the construction of LLMs which attain superior results in the NLP field has been observed since Vaswani et al. (2017)'s initial proposition Tay et al. (2020). Issues arise surrounding the pre-training and deployment of LLMs, which often comprise millions or billions of parameters, require datasets which constitute billions of words, and are compute-hungry due to processing billions of tokens during the pre-training phase Sharir, Peleg and Shoham (2020). The total financial costs of pre-training LLMs derive primarily from: Large computational costs during "several weeks of pre-training with thousands of GPUs" (Zhang et al., 2021, p.1); large storage costs of these huge models, which necessitate the employment of cloud computing providers; and requirements for sophisticated equipment used during inference Zhang et al. (2021).

Strubell, Ganesh and McCallum (2019) calculated the fiscal cost and CO_2 emissions (lbs) of training several LLMs including BERT, NAS So, Liang and Le (2019), and GPT-2 during a 6 month period, identifying a maximum cost of USD 3.2 million for cloud compute costs alone, and a maximum CO_2 emission of 620,000 lbs due to the necessity to run these models utilising electricity - often generated using fossil fuels. In 2019, the average per capita CO_2 emission for UK inhabitants was 11464 lbs Climate Watch and GHG Emissions (2020). Training of these models alone for a sum GPU time of 9998 days was estimated to cost USD 9870 in electricity charges (Strubell, Ganesh and McCallum, 2019, p. 3648). Sharir, Peleg and Shoham (2020) acknowledge that, while changes to model architecture and training schemes can effectuate smaller changes in the cost of LLM floating-point operations, the general trend surrounding the cost of these models' operations appears to be increasing overall. Strubell, Ganesh and McCallum (2019) also point out that academic researchers in the field are significantly constrained by the costs associated with training LLMs, contrasting with industry researchers who are often not constrained in the same way.

Furthermore, LLMs require significant amounts of data, and are often being trained in an unsupervised manner of large corpora of data collated from one of the largest sources of human-written text in history - the internet. During initial training of its GPT 3.0 model, OpenAI identified that the model produced such “toxic" Perrigo (2023) output when trained on unlabelled data from the internet, that manual data annotators had to be employed before the model could be made available to the public. Developers and operators of multiple Transformer-based LLMs and other neural architectures, which utilised proprietary data collated from the internet, have also subsequently been subject to copyright lawsuits as a result Vincent (2023) Shang (2023) Claburn (2023). Acquisition of labelled training corpora suitable for models which require significant resources is thus incredibly difficult to obtain due to data quality, copyright, and privacy considerations.

Thus, the training of Transformer-based and other neural architectures particularly for the research community necessitates a move towards models which are less computationally and fiscally expensive, perpetuate fewer existing biases from training data, and do not infringe on the intellectual property of individuals through the data used. Stanford University's Taori et al. (2023) recently proposed Alpaca 7B, a fine-tuned version of Meta's LLaMa LLM Touvron et al. (2023) which reportedly only cost USD 600 to train due to fewer parameters whilst attaining relatively impressive results. Alpaca 7B's development symbolises a necessary move towards creating sustainable models for the academic community which are trainable using fewer resources in order to better study the behaviour and limitations of such models. There is a necessity for academicians to be able to conduct research in the field, which has seen great progress due to “industry access to large-scale compute" (Strubell, Ganesh and McCallum, 2019, p.3649), without being bound by financial resource limitations.

2.4.1 ELECTRA

ELECTRA, introduced by Clark et al. (2020) in the paper ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators, proposed a novel pre-training and training approach to MLM. Clark et al. considered that BERT's MLM approach, in which $\sim$15% of the input tokens to the model were masked, only allowed for the model to correctly learn 15% of each input sequence's tokens whilst incurring "substantial compute cost" (Clark et al., 2020, p. 1). The ELECTRA model emulates a General Adversarial Network (GAN) neural architecture by pre-training a small generator which corrupts input tokens through the creation of synthetic examples, then utilises a discriminator network during pre-training in order to train the model to distinguish between false and positive tokens, a process described as "replaced token detection" (Clark et al., 2020, p. 1).

The base ELECTRA model utilises the same model architecture and most hyperparameters as BERT, and, when trained on a similar amount of steps, "substantially outperforms"(Clark et al., 2020, p.6) BERT-Base and its more complex iteration BERT-Large on benchmarks including the GLUE NLU benchmark and the SQuAD question-answering benchmark. The ELECTRA-Large model also performs comparably on the same benchmarks to BERT state-of-the-art iteration RoBERTa "despite having fewer parameters and using 1/4 of the compute for training" (Clark et al., 2020, p.2-7). Clark et al. consider that these results demonstrate that training LLMs in this way is more computationally and parameter efficient than previous existing approaches Clark et al. (2020).

2.4.2 Why ELECTRA?

The task of sarcasm subcategory classification is inherently difficult, particularly considering that there tends to be overlap or hierarchical relationships between labels within the Semeval dataset Du et al. (2022). The three top-scoring teams for the Semeval 2022 Task 6 utilised iterations of the BERT architecture, and attained impressive results. ELECTRA however has been described as a "supercharged" Briggs (2021) version of BERT, and attained comparable and superior results to BERT and its variants whilst using less compute at the time of its release Clark et al. (2020).

Choosing ELECTRA for the task of multi-class sarcasm subcategory classification therefore stems from a number of strengths and advantages inherent to the model, which are anticipated to be beneficial for this task:

1. **Performance Efficiency:** ELECTRA has been proven to delivery comparable if not superior performance to BERT and its iterations whilst using significantly less computational resources Clark et al. (2020). ELECTRA's adversarial training approach utilising a smaller generator "increases the relative per-step costs"(Sharir, Peleg and Shoham, 2020, p.3) of floating-point operations whilst ultimately requiring fewer steps, thus decreasing the overall financial and compute cost when training. Though financial cost for simply fine-tuning an existing model is not entirely relevant to its application in this project, performance efficiency is a key benefit considering the resource constraints often associated with pre-trained LLM model training.

2. **Replaced Token Detection and Learning:** ELECTRA's replaced token detection approach enables the model to learn from the entire input sequence, unlike BERT which learns from only 15% of the masked tokens in a sentence Clark et al. (2020). It is considered likely therefore that ELECTRA's capacity for learning in this instance could potentially lead to greater encapsulation of subtleties and nuances in sarcasm subcategories which other models may miss.

3. **Discriminatory Approach:** ELECTRA's discriminatory approach to MLM focuses on generating "high quality" (Clark et al., 2020, p. 10) false samples through the generator which can therefore aid the model's discriminator to more efficiently distinguish between true and false inputs. This approach is designed to help the model understand context more effectively than BERT's MLM approach. As previously mentioned, context is significant when training models to identify ironic speech and sarcasm, and thus ELECTRA's contextual capabilities are expected to provide an advantage for the task.

4. **Previous successful applications:** ELECTRA's impressive performance in tasks similar to multi-class sarcasm subcategory classification - including binary sarcasm detection in English Nigam and Shaheen (2022) and Arabic Abu Farha and Magdy (2021) and general sentiment analysis B et al. (2023) - suggests that the model will also be successful when applied to this task.

5. **Building on existing research:** To date, there does not appear to be research conducted assessing ELECTRA's efficacy when applied to the task of either binary multi-label sarcasm subcategory classification as proposed by Abu Farha et al. (2022), or multi-class sarcasm subcategory classification. Thus, it is hoped that this study could contribute to the task at hand whilst also understanding the potential applications of ELECTRA in novel areas.

Chapter 3

Methodology

3.1 System diagram

The methodology outlined within this section primarily centers around the decisions for implementing an initial model - the ELECTRA Classifier - for general sarcasm detection, and a secondary model - the Custom ELECTRA Classifier - for the downstream task of multi-class classification. The models implemented during this project include a base ELECTRA model which is partially fine-tuned to generalise on sarcastic data with a binary classification head attached. This model is then saved and loaded in to the second class, after which the classification head is discarded and replaced with a multi-class classification head to be trained. Figure 3.1 (below) provides an illustration of the training workflow, where the ELECTRA base model (in blue) represents the binary sarcasm detection model, and the resultant fine-tuned ELECTRA model (in pink) represents the multi-class classification model:

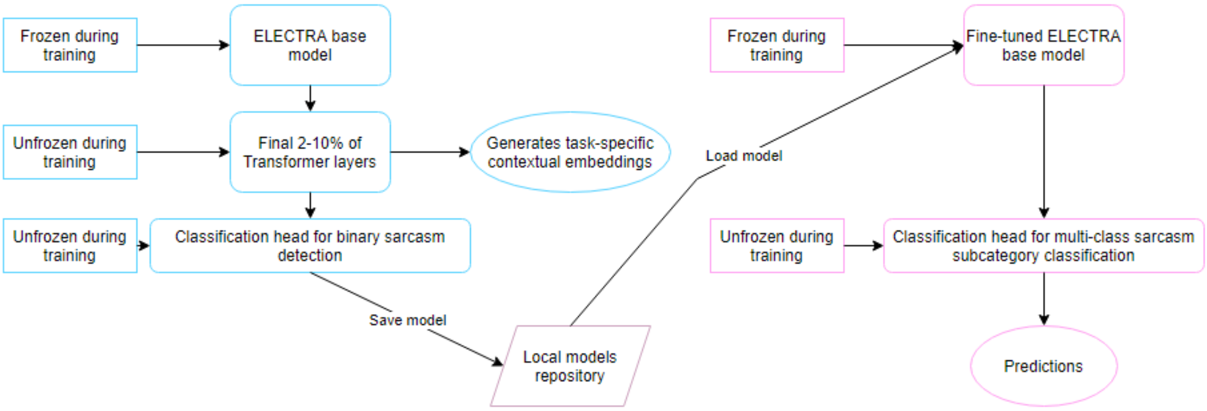

Frozen during training
ELECTRA base model
Unfrozen during training
Final 2-10% of Transformer layers
Generates task-specific contextual embeddings
Unfrozen during training
Classification head for binary sarcasm detection
Save model
Local models repository
Load model
Frozen during training
Fine-tuned ELECTRA base model
Unfrozen during training
Classification head for multi-class sarcasm subcategory classification
Predictions

Figure 3.1: Fine-tuning ELECTRA for Sarcasm Subcategory Classification

3.1.1 Multi-label vs multi-class

The implementation of the ELECTRA Classifier raised questions surrounding its applicability for a multi-label classification task. Multi-label classification, the primary aim for participants in Semeval Subtask B, aims to classify samples which may belong to more than one class in a range of classes. The Semeval datasets were created for employment in a multi-label classification task, and as such each positive sample can be classified as one or more of the sarcastic subcategories. For example, the sentence:

“42.90% of adulthood is just refilling your Brita pitcher"

In Abu Farha et al. (2022)'s training dataset is identified as an instance of both irony and overstatement, with a label list of 0,1,0,0,1,0. Multi-class classification however aims to predict a single label for each sample from a range of classes, and was the primary aim for the Custom ELECTRA Classifier. This design decision was made due to the overlapping relationships identified between data samples, where a high proportion of the under-represented subcategories such as understatement, overstatement, and satire were also labelled as ‘sarcasm’ or ‘irony’, which could prevent a model fine-tuned on sarcastic data from attaining superior results.

3.2 Data

Two datasets were used for the task: a general sarcasm detection dataset comprised of sarcastic and non-sarcastic news headlines in order to partially fine-tune the ELECTRA base model to generalise on sarcastic data, and the prerequisite dataset compiled by the Semeval 2022 Task 6 organisers Abu Farha et al. (2022). This section will provide an overview of both datasets and evaluate their suitability for each task.

3.2.1 Primary project dataset: Semeval 2022 Subtask B

As previously mentioned, the Semeval 2022 Task 6 was split into three tasks: binary sarcasm detection, binary multi-label classification for subcategory classification, and sarcasm determination from a sarcastic text and its non-sarcastic rephrase. Subtask B - the task on which this project is based - provided labelled English language training and test data. The training dataset comprises a total of 3468 samples, 867 of which are sarcastic and 2601 of which are non-sarcastic, while the test dataset comprises a total of 1400 samples, 201 of which are sarcastic and 1199 of which are non-sarcastic. Data was collated by Abu Farha et al. (2022) from the authors of tweets who were asked to provide examples of three non-sarcastic tweets and one sarcastic tweet published from their account. Trained data annotaters were subsequently employed to provide the subcategory labels for each sarcastic data sample. According to Abu Farha et al. (2022) the following definitions were assigned to the English language data samples for each subcategory:

> “1. Sarcasm: contradicts the state of affairs, is directed towards an addressee, and expresses a critical attitude; 2. Irony: contradicts the state of affairs, may or may not be directed towards an addressee, but if it is, is not obviously critical towards that addressee; 3. Satire: is directed towards and addressee whom it appears to support, but underneath it express disagreement, mocking, contempt, or derogation; 4. Understatement: does not contradict the state of affairs, but

> undermines its weight; 5. Overstatement: does not contradict the state of affairs, but assigns unrealistically high weight to it; 6. Rhetorical question: a question with an implicated answer that contradicts the state of affairs." (Abu Farha et al., 2022, p. 804)

There are unequal class distributions within the Semeval Task 6 datasets, primarily deriving from the larger proportion of non-sarcastic to sarcastic samples, thus the non-sarcastic samples were removed from both the training and testing datasets during preprocessing. The initial phase of the project attempted to implement an ELECTRA classifier utilising only the base model with this dataset, and identified that the model itself routinely produced null predictions on several classes in the test dataset, illustrating bias towards the sarcasm and non-sarcastic samples.

Previous applications of Semeval dataset

The class imbalance due to unequal class distributions in the Semeval dataset elicited several different approaches by teams participating in the task to mitigate it and attain superior results or improve generalisation. These approaches included ensemble learning approaches, adversarial training through data mutation, and fine-tuning models for each subcategory's downstream task Du et al. (2022) El Mahdaouy et al. (2022) Abu Farha et al. (2022). One of these teams identified that there appears to be an "apparent hierarachical relationship" (Du et al., 2022, p. 817) between the subcategories, with sarcasm and irony representing primary labels, and the remaining classes representing secondary labels.

3.2.2 Data preprocessing for the Semeval dataset

The quality of any data produced by a model is dependent upon the quality of the data inputted to that model. As such, the data preprocessing stage is significant to ensure that the dataset integrity is preserved whilst also ensuring that the model can draw out contextual meanings from the data. To this end, the data preprocessing techniques used were implemented to ensure the least possible fragmentation of the existing data. The Semeval dataset comprised of individuals' tweets, and so functions were implemented in order to remove any user twitter handles to preserve anonymity, replacing these with "account_name"; remove urls which were replaced with "http"; and remove contractions such as "isn't", replacing these with the full words. The column containing a sarcastic sentence's non-sarcastic rephrase was removed, along with a column entitled 'sarcastic' used primarily for the Semeval Subtask A (binary sarcasm detection), which was not present in the test dataset outlined for the task. The Semeval training dataset comprises a total of 3468 samples, 867 of which are examples of sarcastic text (i.e. any of sarcasm, irony, satire, understatement, overstatement, and rhetorical question) and 2601 of which are non-sarcastic. The non-sarcastic samples are removed from the dataset in order to help the model identify particular instances of sarcastic subcategories.

Additionally, during implementation for the multi-class task, several issues with the Semeval datasets were identified which were remedied by the following data preprocessing techniques:

Removing overlap

As (Du et al., 2022, p. 817) identified, there is indeed an "hierarchical relationship" between the positive values for 'Sarcasm' and 'Irony' and the remainder of the subcategories (satire, overstatement, understatement, rhetorical question). Initial implementation experimented with preserving all positive value instances of sarcasm, irony, and the minority subcategories as they are found in the Semeval dataset. However, closer analysis of the training dataset before the training/validation split identified that the subcategories overlap with the 'Sarcasm' and 'Irony' subcategories in the following ways:

Table 3.1: Overlap between 'Sarcasm' class and subcategories:

Label	Num Samples	Overlap with 'Sarcasm'	Percentage
Satire	25	21	84%
Irony	155	1	0.65%
Overstatement	40	31	76%
Understatement	10	6	60%
Rhetorical Question	101	86	85%
Total	331	145	43.8%

Table 3.2: Overlap between 'Irony' class and subcategories:

Label	Num Samples	Overlap with 'Irony'	Percentage
Satire	25	4	16%
Overstatement	40	9	23%
Understatement	10	4	40%
Rhetorical Question	101	15	14%
Total	176	32	18%

When approaching this task as a multi-class problem as opposed to a multi-label problem, it is important that samples do not overlap between classes as a model training on a small dataset is unable to distinguish the difference between instances of the minority classes and the majority classes of sarcasm and irony, meaning that during testing it will default to predicting the majority-represented class. Therefore, it was decided that removing all positive values within the sarcasm and irony columns which intersected with positive values in any of the minority columns in the training dataset could help the model to identify these samples better.

This preprocessing technique removed 144 samples from the 'Sarcasm' class, and 31 samples from the 'Irony' class which overlapped with the minority classes in the Semeval dataset. It should be noted that the values of 144 and 31 are slightly different from the values of 176 and 32 identified as total values in Tables 3.1 and 3.2, as each sample may belong to more than two of the subcategories. No other overlap between samples of the minority classes was identified, with the exception of 2 samples labelled both 'Understatement' and 'Rhetorical question' which were re-labelled as only 'Understatement' due to its initial small class sample size. The Semeval training dataset was then split into training and validation datasets. The following table illustrates the change in sample sizes following the removal of class overlap in the training dataset:

Table 3.3: Semeval training dataset overlap removal

Overlap	Value before	Value after	Difference
Sarcasm	713	569	144
Irony	155	124	31
Rhetorical question	101	99	2
Total	969	792	177

The final class distribution of all classes in the training dataset following removal of overlap is identified as follows:

Table 3.4: Semeval training dataset values following overlap removal

Label	Value
Sarcasm	569
Irony	124
Satire	25
Understatement	10
Overstatement	40
Rhetorical question	99
Total	867

This overlap removal technique was also applied to the test dataset for consistency when predicting. The test dataset comprises of a total of 1400 samples, 200 of which are sarcastic (encompassing all subcategories) and 1200 of which are non-sarcastic. The values of the test dataset prior to and post overlap removal can therefore be defined as follows:

Table 3.5: Semeval test dataset values prior to overlap removal

Label	Value
Sarcasm	180
Irony	20
Satire	48
Understatement	1
Overstatement	10
Rhetorical question	11
Total	270

Table 3.6: Semeval test dataset values following overlap removal

Label	Value
Sarcasm	115
Irony	15
Satire	48
Understatement	1
Overstatement	10
Rhetorical question	11
Total	200

Data augmentation

Following implementation of several data augmentation techniques utilising the TextAttack library Morris et al. (2020)[1], the ELECTRA Classifier demonstrated superior results when identifying some instances of the minority classes represented within the Semeval dataset in comparison with initial models which were not trained on augmented datasets. These data augmentation techniques included using TextAttack's *WordSwapExtend, WordSwapRandomCharacterInsertion, and WordSwapRandomCharacterDeletion* classes, which use an augmenter to generate a number of synthetic samples proportionate to each class's representation within the training dataset. These samples do not alter the meaning of the sentence or underlying nuances, as they do not use more widespread text augmentation techniques which replace words with synonyms, swap words, or delete words entirely. The TextAttack augmentation techniques only alter certain characters within the sample text. Therefore, the semantic and nuanced meanings behind the words used in each particular context remain unchanged.

3.2.3 Supplementary project data: the News Headlines Dataset

Due to the class imbalance in the Semeval dataset and the ELECTRA base model's inability to effectively train on this small, imbalanced dataset, it was hypothesised that using a larger, open source dataset to aid in the ELECTRA base model's generalisation on sarcastic versus non-sarcastic data could be beneficial for results. As such, the supplementary dataset used in fine-tuning the ELECTRA base model is the News Headlines Dataset for Sarcasm Detection, produced by Misra and Grover (2021) and Misra (2022) [2]. The data in this repository was collated from the headlines of two news sources, the Onion and the Huffington Post, in order to help overcome noisy data derived primarily from Twitter-based datasets which had previously been prevalent in NLP for sarcasm detection Misra (2022). The dataset comprises $\sim$28000 samples, with approximately half of these samples comprising sarcastic and non-sarcastic text. The dataset was preserved almost in its entirety, only removing an extra column which included values for the link to the article from which the headline had been extracted. No data augmentation was performed on this dataset due to its balanced sample distribution.

[1] A full list of the extensions and packages used during project implementation can be found in Appendix A of the report

[2] Dataset available at: https://rishabhmisra.github.io/publications/

Evaluation of supplementary data

The News Headlines Dataset utilised in the fine-tuning stage is mostly suitable for generalisation purposes, as the authors have specifically generated the dataset for the task of sarcasm detection. However, there are questions raised regarding how a model trained using supervised learning on a general sarcasm dataset could identify meaningful patterns of sarcasm subcategories which are otherwise unlabelled. Due to a general unavailability of large corpora of labelled sarcastic data for sarcasm subcategory detection using supervised learning, this hurdle at present is insurmountable.

3.3 Tokenization for LLMs

Tokenization is an integral part of training neural architectures in NLP - particularly for LLMs such as ELECTRA. In the context of NLP, tokenization takes input and splits it into smaller units of granularity - in the form of characters, words, or sentences - to produce tokens which can be fed into the neural architecture and mapped into vector space Hugging Face (n.d.c). Tokenization for machine learning is significant, as it allows the network to understand simple representations of textual data whilst preserving context and quantifying meanings or word frequency represented in the data Stanford University (n.d.c). Most machine learning networks also expect data to be fed into the network as embedding vectors, and thus text tokenization encodes the textual data as values in this format.

3.3.1 ELECTRA tokenization

Tokenization techniques vary for each model, and it is therefore often necessary to utilise the tokenizer which has been developed for each specific model to mitigate errors when fine-tuning these existing models. ELECTRA utilises a tokenizer based on WordPiece, the tokenization algorithm developed to pre-train BERT which has not been published in open source Hugging Face (n.d.a). WordPiece tokenization first splits input into words divided by punctuation and whitespaces, then into subword tokens which "often retain linguistic meaning" Song and Zhou (2021) Hugging Face (n.d.c). By splitting textual data in this way to preserve subwords from the data, the tokenization process mitigates the ‘out-of-vocabulary’ problem often previously encountered when training NLP models, in which a model was unable to process tokens which it had previously not come across, by splitting the words into identifiable words. The ELECTRA tokenizer also introduces special tokens to the textual data, including [CLS], [SEP], [UNK], and [PAD], which denote sentence start, sentence end, unknown tokens, and padding tokens respectively. Padding tokens are appended to the end of sentence samples in order to ensure that all sentences are the same length. An example of a WordPiece-tokenized sentence could therefore be represented as so:

Original sentence:

Cats are unbelievably cool!! 🐱

After tokenization:

[‘[CLS]’,‘cat’,‘are’,‘un’,‘##bel’,‘##iev’,‘##ably’,‘cool’, ‘[UNK]’,‘!’,‘!’,[PAD],[PAD],[SEP]]

Where ‘##’ denotes that the tokens are subword tokens which have been broken down from a larger word, and [UNK] representing a symbol which the tokenizer has not encountered - the cat emoji. The tokens are then represented as numerical values in the form of unique IDs based on the model's vocabulary, following which positional encoding is applied to preserve word order, ready to be encoded as embedding vectors corresponding to these unique IDs Devlin et al. (2019) Vaswani et al. (2017). The tokenizer afforded by the ELECTRA pre-trained model was used for this task, as it is available to use as part of the HuggingFace library [3] from which the ELECTRA base model was imported. Emojis included in the tweets for the Semeval datasets remain in the tokenized text, as they often provide information from which important linguistic and contextual nuances can be derived.

[3] A full list of the libraries and extensions used in project implementation can be found in Appendix A of the report.

Chapter 4

Implementation and Testing

4.1 System implementation overview

The models are implemented in a python environment configured in a Docker container on Windows Subsystem for Linux 2 (WSL 2), running on an NVIDIA RTX 2080 Ti PC. This implementation decision was made following initial experimentation with cloud computing via Google Colab, which led to extremely slow training times (∼15 minutes per epoch on the supplementary dataset). It was therefore decided that accessing the PC's GPU with NVIDIA's CUDA application through WSL 2 and Docker could help ameliorate the slow training times. This resulted in faster iteration through models and testing. Additionally, a full list of the extensions and packages utilised during implementation can be found in Appendix A of the report.

4.2 Data

4.2.1 Analysis of primary dataset

Two datasets are used for the project's implementation: the Semeval 2022 Subtask B dataset which includes examples of sarcasm subategories, and the News Headlines Dataset which only includes examples of sarcastic and non-sarcastic text. The Semeval training dataset comprises 3468 samples, 867 of which are positive instances of sarcastic text (i.e. any of sarcasm, irony, satire, understatement, overstatement, and rhetorical question) and 2601 of which are non-sarcastic. The Semeval testing dataset comprises 1400 samples, 200 of which are sarcastic (encompassing all subcategories) and 1200 of which are non-sarcastic. As previously mentioned, the non-sarcastic samples are removed from the training and test datasets in order to aid the model in identification of sarcasm subcategories. As a reminder, Table 4.1 is the distribution of the test dataset following overlap removal between the classes:

Table 4.1: Semeval Test Dataset Values Following Overlap Removal

Label	Value
Sarcasm	115
Irony	15
Satire	48
Understatement	1
Overstatement	10
Rhetorical question	11
Total	200

Initial implementation randomly split the training dataset 80/20 into training and validation sets. Initial analysis of the Semeval training dataset following this split and prior to the application of the overlap removal and data augmentation techniques identified the following class sample distribution throughout:

Table 4.2: Initial Semeval Training, Validation, and Test Dataset Sample Distribution

Values	Training Dataset	Validation Dataset	Test Dataset
Sarcasm	545	168	180
Irony	124	31	20
Satire	20	5	48
Understatement	7	3	1
Overstatement	29	11	10
Rhetorical question	77	24	11
Total	802	242	270

There are proportionally large under-representations of almost all features in an already small dataset, which made it incredibly difficult for the base model to extract meaningful information from. Additionally, there was an overlap of 177 samples between the columns labelled 'Sarcasm', 'Irony', and 'Rhetorical Question' with the rest of the minority classes - 'Satire', 'Overstatement', and 'Understatement' - thus data preprocessing techniques were applied in order to remove this overlap between the columns. Table 4.3 (below) highlights the training dataset class sample distribution following the removal of overlap between the classes, and the 80/20 splitting of the Semeval training dataset into training and validation datasets for a total of 867 samples, 693 of which are training and 174 of which are validation:

Table 4.3: Semeval Training Dataset Positive Values Following Overlap Removal and Dataset Split

Label	Training Value	Validation Value	Total
Sarcasm	447	122	569
Irony	106	18	124
Satire	20	5	25
Understatement	7	3	10
Overstatement	31	9	40
Rhetorical question	82	17	99
Total	693	174	867

The negative values of the dataset following the removal of overlap and 80/20 split into training and validation datasets is also delineated in Table 4.4 (below):

Table 4.4: Semeval Training Dataset Negative Values Following Overlap Removal and Dataset Split

Label	Training value	Validation value	Total
Sarcasm	246	52	298
Irony	587	157	743
Satire	673	169	842
Understatement	686	171	857
Overstatement	662	165	827
Rhetorical question	611	157	768

Table 4.5 (below) outlines the total class sample distribution for each class in the Semeval training dataset after the overlap removal techniques have been applied:

Table 4.5: Total Sample Distribution

Label	Positive samples	Negative samples	Total
Sarcasm	569	298	867
Irony	124	743	867
Satire	25	842	867
Understatement	10	857	867
Overstatement	40	827	867
Rhetorical question	99	768	867

The class imbalance within the training and validation datasets led to the hypothesis that an approach to aid the base model with generalising on sarcastic data samples could be advantageous to the results produced. Thus, the more imbalanced Semeval dataset is used in the downstream transfer learning task of sarcasm subcategory classification with a custom classifier following fine-tuning of the general ELECTRA Classifier with the News Headines Dataset.

4.2.2 Analysis of secondary dataset

The supplementary News Headlines Dataset was therefore used for the task of fine-tuning the ELECTRA base model for generalisation on sarcastic data. This dataset is split into training, validation, and testing datasets through an 80/10/10% split, and analysis of the datasets following this split identified the following class sample distribution:

Table 4.6: News Headlines Dataset Training and Validation samples

Values	Training Dataset	Validation Dataset	Test Dataset
Is sarcastic	10905	1350	1379
Is not sarcastic	11990	1512	1483

The News Headlines Dataset is utilised in the first stage of model training - fine-tuning part of the ELECTRA base model's layers. The balance between sarcastic and non-sarcastic classes means that there is no requirement to augment or mutate data, with the base ELECTRA Classifier converging well on the dataset. The dataset can therefore be used to help the final 2% of the model's layers to generate task-specific embeddings for sarcasm identification in text. Utilising a larger, more balanced dataset in this way reduces the requirement to augment or mutate the Semeval Subtask B dataset, which, when altered through techniques such as synonym replacement, could lose nuances in the sarcastic text leading to inaccurate data samples and inferior performance.

Given the class distribution imbalance highlighted in the Semeval dataset, and the risk of losing important contextual and linguistic nuance in the dataset when using popular NLP data augmentation or mutation techniques, the decision was made to first partially fine-tune some of the task-specific embeddings of the base model to generalise on sarcastic data. Results of training did not show a significant change in performance when unfreezing between 2 and 8% of the model's top layers, and as such it was decided to only unfreeze the top 2% of the base model's layers during training. Following this, the fine-tuned model could then be saved and re-loaded with a new multi-class classification head in order to use the fine-tuned model for transfer learning, training the classification head only on the Semeval dataset. Further information in relation to the decisions behind implementing both models can be found below.

4.3 ELECTRA Classifier

This section will provide an overview of the ELECTRA Classifier, the model trained on the News Headlines Dataset in order to fine-tune the top 2% of ELECTRA's base layers, following the hypothesis that this could aid the model in generalising on sarcastic data.

The ELECTRA Classifier is initialised with none of the base model's layers unfrozen for an initial warmup period of 10,000 steps, following which the final 2% of the layers are incrementally unfrozen. The model is initialised with a base learning rate of 2e-5 using the Rectified Adam (RAdam) optimizer Liu et al. (2019), and uses the Cosine Annealing with Warmup Restarts learning rate scheduler Pytorch (n.d.a) to schedule learning rate changes following the initial 10,000 warmup steps. Further information relating to the decision making process surrounding the warmup period, learning rate scheduler, and RAdam optimizer can be found in section 4.3.2. The model is initialised with a maximum epochs value of 10,000 in order to provide a higher total value for the number of training steps required for instantiating the learning rate scheduler, though the model rarely required more than 100 epochs to achieve convergence on the training data.

Training and validation loss, accuracy, precision, and recall are logged at each training and validation step end, with early stopping with a patience of 5 epochs based on the validation loss implemented as a callback in the *trainer.fit()* method to prevent overfitting. These metrics were chosen in order to aid with debugging during the training process, and are not used as an accurate representation of model performance. Logs are created automatically by the Pytorch Lightning framework, although there is also a custom Metrics Callback class implemented in order to track and store all metrics recorded during the training and validation epochs. Tensorboard logs are also created during each training round, in order to visualise the training and validation results. Loss is computed using PyTorch's binary cross entropy loss, which measures the difference between the model's predicted value and the expected (true) value.

4.3.1 Fine-tuning

In initial experiments with the ELECTRA base model and the Semeval dataset, the model yielded F1 Scores inferior to those achieved by previous participants in Semeval Subtask B. These results led to the hypothesis that fine-tuning the base ELECTRA model to generalise on sarcastic data before applying this fine-tuned model to the subcategory classification task could yield better results. This resulted in the approach delineated above in Figure 3.1, where the model is first partially fine-tuned on a larger dataset before being saved, re-loaded, and trained with a different classification head.

The initial approach taken was to unfreeze all of the layers of the base ELECTRA model after 10,000 warmup steps. However, this approach yielded extremely poor results including overfitting on the validation dataset and resulted in a model which failed to make predictions when saved, loaded, and trained on the Semeval dataset. Overfitting in this instance resulted from updating the entirety of the base model's weights during training, losing valuable contextual information obtained during the base ELECTRA model's pre-training phase. Striking a balance between fine-tuning the model for the general task of sarcasm detection on a moderately-sized dataset whilst also ensuring its adaptability for the downstream task of subcategory classification after saving and loading the model was crucial.

Therefore, the revised approach taken was to only unfreeze the top $\sim$ 2-8% of the base model's layers. These top Transformer layers, which perform self-attention and feed-forward operations, typically produce task-specific contextual embeddings from information learned during pre-training and can therefore be partially adapted for a new specific task. The current implementation gradually unfreezes only the top 2% of ELECTRA's base layers after 10,000 warmup steps. The modification of unfreezing 2% instead of a maximum of 8% of the base model's layers yielded a modest increase in performance of 0.01%. The gradual unfreezing of the top 2% of ELECTRA's layers is defined as so:

Listing 4.1: Gradual Unfreezing of ELECTRA's layers

```
def unfreeze_next_layer(self):
    num_params = len(list(self.model.electra.parameters()))
    freeze_idx = int(num_params * 0.98)
    unfreeze_idx = self.current_unfreeze_idx
```

```
if self.current_unfreeze_idx >= 0 and self.current_unfreeze_idx
   >= freeze_idx:
     param = list(self.model.electra.parameters())[unfreeze_idx]
     param.requires_grad = True
     print(f"unfroze␣layer␣idx␣{self.current_unfreeze_idx␣-␣1}")
     self.current_unfreeze_idx -= self.unfreeze_step
```

4.3.2 Scheduling learning rates

The scheduling of learning rates is functionality afforded by multiple deep learning frameworks, and allows the learning rate to be adapted at certain points during training based on certain conditions. In this instance it was decided that utilising a learning rate scheduler which could affect changes in the learning rate during training after a period of 10,000 warmup steps could be beneficial. Implementation experimented with several learning rate schedulers, including Hugging Face's *get_cosine_schedule_with_warmup* and *get_linear_schedule_with_warmup*, both of which produce variations in learning rates during and after the warmup step period. Cosine Annealing with Warm Restarts (CAWR) is a PyTorch learning rate scheduler Pytorch (n.d.a). CAWR periodically decreases a relatively large learning rate to a lower rate value following a cosine curve Loshchilov and Hutter (2016). CAWR simulates a restart of the optimization process by restarting the learning rate, increasing it back to its initial value and providing a chance to escape local minima Loshchilov and Hutter (2016).

The RAdam optimizer Liu et al. (2019) initially proposed in 2019, was chosen in tandem with CAWR due to its ability to handle large variances in adaptive learning rates and optimize convergence. Initial training runs indicated that convergence on the training and validation data was normally being reached at ~10,000 steps with the News Headlines Dataset. Therefore, it was hypothesised that keeping the learning rate at a consistently small number during this period would allow the model to extract more information from the data during these steps, following which the learning rate could be adapted per step. Configuring the learning rate, optimizer, and learning rate scheduler in this way provides substantive conditions for the model to extract more information from the News Headlines Dataset during and after the warmup steps implemented, in order to help the model better generalise to sarcasm detection and facilitate downstream transfer learning.

4.3.3 Evaluation metrics

As previously mentioned, several metrics are logged during training, validation, and testing of the ELECTRA Classifier, in order to debug issues when they arose while evaluating the model's performance from a range of perspectives. As such, loss, accuracy, precision, and recall are logged during each of the model's training and validation steps, while only F1 Score and accuracy are logged during the test and predict steps. Evaluation metrics were used during the partial fine-tuning phase primarily to ensure that the model was deriving information from the dataset, but were not used as a measure of success in the project implementation.

Accuracy, whilst not always entirely useful for gauging a model's performance on certain tasks, provides a valuable insight into whether the model is learning any meaningful representations from the data. Precision and Recall are key components for calculating a model's F1 Score, and so being able to see these results during training aided when identifying issues with the learning rate or how the data had been pre-processed. Validation loss, logged at each batch, step, and epoch end, is monitored during training, and is the metric monitored for signalling early stopping to prevent overfitting. Accuracy is perhaps a superfluous addition to logging in the test and predict steps, as this evaluation metric holds no bearing on the F1 Score, but was useful to include for reference nonetheless.

4.4 Custom ELECTRA Classifier

This section provides an overview of the second model, the Custom ELECTRA Classifier, and its iterations, which were implemented to be trained and evaluated on the smaller Semeval dataset.

Implementation experimented with two iterations of the Custom ELECTRA Classifier. The first, Custom-ELECTRA, is the module for training a 6-class classification head for each subcategory defined in the dataset: Sarcasm, irony, satire, understatement, overstatement, and rhetorical question. The second, Aggregate-ELECTRA, is a similar module, but which only has a 3-class classification head for separate subcategories defined during experimentation and created from the dataset in its totality: Sarcasm, not sarcastic, and other - which encapsulates all of the minority classes into one class. Both models are a saved and re-loaded version of the base ELECTRA Classifier, which is automatically saved to a checkpoint following model training and testing. Following re-loading, the existing model's binary classification head is discarded and a new multi-class classification head is appended. These models are initialised with none of the fine-tuned models layers unfrozen.

Consideration was given to the creation of a small neural network built on top of the ELECTRA Classifier and before the Custom ELECTRA Classifier's classification head. However, when training and testing on the smaller dataset this led to inferior results. As a result of this failed implementation, it was decided that increasing the size of the model to use with the smaller Semeval dataset would not aid in increasing model performance.

Similarly to the ELECTRA Classifier, training and validation loss, precision, and recall are logged at each training and validation step end for both models. Early stopping with a patience of 5 epochs based on validation loss is implemented as a callback in the *trainer.fit()* method to prevent overfitting, particularly on a smaller dataset. There is also a custom Metrics Callback class implemented in order to track and store all metrics during training and validation epochs, while Tensorboard logs are also created for data visualisation and debugging. The models are initialised with a maximum epoch value of 1000, though they did not require more than 30 epochs to train before stopping when early stopping conditions were met.

4.4.1 Learning rates

The Custom-ELECTRA model for 6-class classification is initialised with a base learning rate of 5e-6. The learning rate is particularly small due to the class imbalance previously highlighted in the Semeval dataset, and uses the RAdam optimizer similarly to the ELECTRA Classifier with a weight decay of 0.01 to help prevent overfitting. Iterations utilising a higher learning rate (between 1e-3 and 5e-4) yielded poor performance with no decrease in output loss after around 5 epochs.

The Aggregate-ELECTRA model for 3-class classification is initialised with a base learning rate of 5e-4, for similar reasons to those outlined above, though the learning rate is higher as this yielded better performance. A learning rate scheduler based on PyTorch's *OneCycleLR* Pytorch (n.d.b) is implemented for Aggregate-ELECTRA which adjusts the learning rate each step between a maximum value of 2 times the instantiated learning rate (5e-4) and a minimum value lower than the instantiated learning rate, in a strategy which has been proven to aid fast convergence on training data Smith and Topin (2018). This learning rate scheduler was initially applied to the Custom-ELECTRA (6-Class) model, but yielded a decrease in performance. Therefore, the learning rate scheduler with a slightly higher learning rate is initialised in Aggregate-ELECTRA, which led to a modest performance increase in this model for the 3-class task.

4.4.2 Evaluation Metrics

In order to ensure that this research was able to be contextualised in line with other results obtained by teams participating in the Semeval Subtask B, the macro F1 Score of the model's predictions over all classes was used as the primary evaluation metric for this task. The macro F1 Score is calculated as follows:

$$F_1 = \frac{1}{n}\sum_{c=1}^{n}(F_1^c), \tag{4.1}$$

"where F_1^c represents the F_1 score for the cth category and n is the number of categories." (Abu Farha et al., 2022, p. 806)

The macro F1 Score is the harmonic mean of precision and recall, where precision is the ratio of true positive predictions to the total number of positive predictions and recall is the ratio of true positive predictions to the total number of actual positive instances. F1 Score is therefore defined as follows:

$$F_1 Score = 2 * \frac{Precision * Recall}{Precision + Recall}, \tag{4.2}$$

Korstanje (2021)

where:

$$Precision = \frac{TP}{TP + FP}, Recall = \frac{TP}{TP + FN} \tag{4.3}$$

Korstanje (2021)

Macro F1 Score is calculated in the Custom ELECTRA Classifier instance by computing the F1 Score for each class separately then taking the average of each of these values, meaning that equal weight is given to each class. Thus, precision, recall, class F1 Score, and macro F1 Score are all logged at each step and epoch during training, testing, and prediction, along with loss to measure model performance. The Custom-ELECTRA Classifiers are instantiated with a weighted categorical cross entropy loss function based on the proportions of the classes represented in order to direct the models' attention to under-represented classes in the data. Categorical cross entropy is defined as follows:

$$L_i = -log(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}), \tag{4.4}$$

Stanford University (n.d.a)

Or equivalently:

$$L_i = -f_{y_i} + log\sum_j e^{f_j} \tag{4.5}$$

Stanford University (n.d.a)

Where f_j is the jth element of the class score vector f and L_i denotes the full loss across the dataset Stanford University (n.d.a). The softmax function, applied as part of PyTorch's categorical cross entropy loss function in order to obtain predictions across all classes between the values of 0 and 1, is defined as follows:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \tag{4.6}$$

Stanford University (n.d.a)

The weights in the loss function ensure that the model is penalized more heavily for confidently asserting incorrect predictions when predicting on the majority class(es). The weights, which are the number of samples per class, are therefore passed to PyTorch's CrossEntropy function, and implementation of a weighted loss function displayed a small improvement in results when predicting across all classes.

4.4.3 6-class classification versus 3-class classification

Initial implementation of Custom-ELECTRA experimented solely with a 6-class classification head appended to the fine-tuned ELECTRA Classifier. However, following training and testing on the Semeval dataset, it was identified that during prediction Custom-ELECTRA was yielding low F1 and macro F1 scores. As such, consideration was given to the possibility that augmenting and adjusting the Semeval dataset could help the model to obtain better results and extract further information from the training dataset. Thus the data preprocessing and augmentation techniques as outlined in the Methodology section were applied to the Semeval training, validation, and test datasets, in order to help train the model more effectively. Data augmentation of all classes originally included in the Semeval dataset was applied to the training dataset only, and table 4.7 (below) outlines the class distribution of the Semeval training dataset following data augmentation techniques for the 6-class Custom-ELECTRA Classifier:

Table 4.7: Semeval training dataset samples (6-class) Custom-ELECTRA

Label	Original Value	Positive Values	Negative Values
Sarcasm	447	2235	8900
Irony	106	2650	8485
Satire	20	1000	10135
Understatement	7	420	10715
Overstatement	31	1550	9585
Rhetorical question	82	3280	7855

3-class classification

Following the application of data preprocessing techniques for the Custom-ELECTRA (6-Class) Classifier, including the removal of overlap between classes and the minority-represented subclasses in the training, validation, and test datasets, and data augmentation techniques applied solely to the training dataset, the results obtained were slightly disappointing.

As illustrated in table 4.7, issues arise when using data augmentation techniques are applied to a heavily imbalanced dataset such as the one provided for Semeval Subtask B. The heavily under-represented classes such as satire, understatement, overstatement, and rhetorical question were augmented by the addition of between 40 and 60 synthetic samples per existing dataset sample. The augmentation however led to a greater imbalance of data not included in the class, simply compounding the existing issue of a comparative lack of samples per minority class despite a greater amount of samples overall. Consideration was therefore given to the possibility of approaching the task in a different way, by utilising some feature engineering methods to manipulate the data and create new classes.

First, a majority class of the non-sarcastic samples originally included in the dataset which did not have their own label was created, and were subsequently assigned the label 'not_sarcastic'. Second, the remaining minority classes which were under-represented in the original dataset (irony, satire, understatement, overstatement, and rhetorical question) were aggregated into another column which was subsequently assigned the label 'Other'. The following information summarises the methodology behind creating Aggregate-ELECTRA, a 3-class Custom-ELECTRA Classifier, with only the previous data preprocessing step of removal of sarcasm, irony, and rhetorical question overlap having already been applied:

Creating a majority class

Initially, a class for the majority values which were included in the dataset was created. That is, the non-sarcastic values which did not have a class of their own, but comprised the majority of samples at around 75% of the original training dataset prior to their removal. A new column labelled 'not_sarcastic' was created which extracted all rows where no positive values were identified for any of the other classes, inverting these negative values as positive to create a class of non-sarcastic samples. It is important to note that simply inverting the 'Sarcasm' values in this instance is insufficient, as then the subcategory samples would be included both in their individual columns, as well as in the 'not_sarcastic' column. The model would therefore be trained on incorrect data for a multi-class task. This method was applied to the Semeval training dataset prior to splitting, as well as to the test dataset for consistency. It should be noted that implementation of the same method (creating a majority class from the not sarcastic samples) was applied to the 6-class classifier for experimentation purposes, but elicited poor task-specific results. Further information can be found in Chapter 5 of the report.

Creating an 'Other' class

The second step which was taken was to create a class label 'Other' which would allow all minority values to be aggregated into one class. This was conducted by identifying all instances where columns of the subcategories excluding 'Sarcasm' - irony, satire, understatement, overstatement, and rhetorical question - held positive values. These values were then aggregated as one label in the 'Other' column, while preserving the text in its original place.

Data augmentation

Following the creation of the 'not_sarcastic' and 'Other' columns while preserving the 'Sarcasm' column, analysis of the training dataset identified the following class sample distribution:

Table 4.8: Aggregate-ELECTRA Semeval Training Dataset Samples Before Augmentation

Label	Positive Values	Negative Values
Sarcasm	440	2334
Not sarcastic	2106	668
Other	228	2546

In order to ensure that there was a relatively equal sample distribution within the training dataset, the data augmentation techniques utilising the predefined classes afforded by the TextAttack library were subsequently applied to the dataset. 20 and 50 synthetic examples are generated for each sample in the 'Sarcasm' and 'Other' classes respectively. Following application of these data augmentation techniques, the class distribution of the Semeval training dataset is as follows:

Table 4.9: Aggregate-ELECTRA Semeval Training Dataset Samples After Augmentation

Label	Positive Values	Negative Values
Sarcasm	8800	13506
Not sarcastic	2106	20200
Other	11400	10906

Chapter 5

Results

5.1 Custom ELECTRA Classifier quantitative results

The Custom ELECTRA Classifier, outlined in its totality in section 4.4 of the report, obtained interesting results during testing. The testing phase of model training is significant as it allows the researcher to evaluate the model's performance on previously unseen data. PyTorch Lightning provides functionality for both testing and predict steps which are implemented as part of the project, however only the test dataset is able to be used for these steps due to a lack of further labelled data required for such a specialised task. The test dataset is therefore implemented in the testing and predict step methods, though only one method is called at a time in order to prevent the model from memorising the test dataset.

The Custom-ELECTRA (6-Class) Classifier runs for between 15 and 30 epochs, while the ELECTRA-Aggregate (3-Class) Classifier runs for between 10 and 30 epochs during an average training cycle. Both models normally terminate training when the early stopping threshold of no decrease in validation loss after 5 epochs is met. A base ELECTRA model (ELECTRA-no-FT), implemented using the same data preprocessing, augmentation and loss function techniques as 6-class Custom-ELECTRA with no layer unfreezing, attains poor results and some null values when predicting the sarcasm subcategories. During implementation, it was hypothesised that the complexity of the ELECTRA base model in tandem with the small, fairly imbalanced Semeval dataset was a partial cause for these null predictions, which prevented the model from deriving meaningful representations from the dataset and which led to the implementation of the fine-tuned Custom-ELECTRA classifier. The 6-class Custom-ELECTRA classifier attained the following results:

Table 5.1: Custom-ELECTRA (6-Class) Classifier Results

Class	Test Metric	Test Value
Class 0 (sarcasm)	F1	0.0523
Class 1 (irony)	F1	0.11
Class 2 (satire)	F1	0.0498
Class 3 (understatement)	F1	0.08
Class 4 (overstatement)	F1	0.1162
Class 5 (rhetorical question)	F1	0.0640
All classes	Macro F1	0.0787
All classes	Test loss	0.2033

Table 5.2: Custom-ELECTRA (6-Class) Classifier Results Breakdown

Class	True Positives	False Positives	False Negatives	True Negatives
Class 0 (sarcasm)	3	4	112	81
Class 1 (irony)	8	115	7	70
Class 2 (satire)	2	3	47	148
Class 3 (understatement)	1	10	0	189
Class 4 (overstatement)	3	44	7	146
Class 5 (rhetorical question)	2	5	8	185

Implementation experimented with generating synthetic samples per class proportional to the representation of these samples within the dataset, varying from generation of $\sim$ 20 to 50 samples per dataset sample, with little difference in performance identified. For reference, ELECTRA-no-FT attained the following test results on the same dataset:

Table 5.3: ELECTRA-no-FT results without fine-tuning

Class	Test Metric	Test Value
Class 0 (sarcasm)	F1	0.4149
Class 1 (irony)	F1	0.0364
Class 2 (satire)	F1	0.0279
Class 3 (understatement)	F1	0.0
Class 4 (overstatement)	F1	0.0
Class 5 (rhetorical question)	F1	0.0274
All classes	Macro F1	0.0844
All classes	Test loss	0.0341

Table 5.4: ELECTRA-no-FT Results Breakdown

Class	True Positives	False Positives	False Negatives	True Negatives
Class 0 (sarcasm)	22	19	93	66
Class 1 (irony)	5	79	10	106
Class 2 (satire)	6	17	43	134
Class 3 (understatement)	0	1	1	198
Class 4 (overstatement)	0	25	8	168
Class 5 (rhetorical question)	6	20	4	170

The 3-Class Aggregate-ELECTRA Classifier, which aggregates the minority classes into one column and includes a majority class not sarcastic column, attained the following results:

Table 5.5: Aggregate-ELECTRA (3-Class) Classifier Results

Class	Test Metric	Test Value
Class 0 (sarcasm)	F1	0.1088
Class 1 (not sarcastic)	F1	0.5282
Class 2 (other)	F1	0.0719
All classes	Macro F1	0.2363
All classes	Test loss	1.4573

Table 5.6: Aggregate-ELECTRA (3-Class) Classifier Results Breakdown

Class	True Positives	False Positives	False Negatives	True Negatives
Class 0 (sarcasm)	21	147	108	1124
Class 1 (not_sarcastic)	872	167	328	33
Class 2 (other)	14	180	73	1135

Screenshots of results obtained by the Custom ELECTRA Classifier modules can be found in Appendix B of the report. The Macro F1 Score for Aggregate-ELECTRA is 0.2363 - the most superior result attained by any of the models which produced predictions for all classes. The results also show that the removal of overlap between samples labelled as both sarcastic and one of the sarcasm subcategories generates less impressive results than were originally attained during the Semeval Subtask B competition. An explanation for the strong results previously obtained could be that there exists a clear hierarchical relationship between the primary labels of sarcasm and irony and those of the subcategories which would elicit biases towards the dominant classes during inference.

Additionally, during the project's testing phase, implementation experimented with a version of the Custom-ELECTRA (6-class) Classifier which had a 'not_sarcastic' class created in a similar way to Aggregate-ELECTRA. This version of Custom-ELECTRA therefore had 7 classes: sarcasm, irony, satire, overstatement, understatement, rhetorical question, and not_sarcastic. This model was implemented in order to assess the classifier's ability to distinguish between sarcastic and non-sarcastic samples, while classifying instances of the subcategories, by utilising all data available within the Semeval datasets - i.e. preserving the dataset without removing the non-sarcastic samples. The results of this experiment are highlighted in Table 5.7 (below):

Table 5.7: Custom ELECTRA (7-Class) Classifier Results

Class	Test Metric	Test Value
Class 0 (not_sarcastic)	F1	0.9730
Class 1 (sarcasm)	F1	0.8449
Class 2 (irony)	F1	0.0
Class 3 (satire)	F1	0.0
Class 4 (understatement)	F1	0.0
Class 5 (overstatement)	F1	0.0
Class 6 (rhetorical question)	F1	0.0
All classes	Macro F1	0.2597
All classes	Test loss	0.1661

5.2 Custom-ELECTRA Classifiers trend description

As illustrated in the above tables, it is clear that, although the base ELECTRA model ELECTRA-no-FT is able to efficiently predict values for most of the sarcastic subcategories with data augmentation techniques applied and therefore attains a higher macro F1 score overall, it is unable to correctly identify instances of two of the most under-represented minority classes of overstatement and understatement, which comprise only 40 and 10 samples in the original dataset. The fine-tuned version of ELECTRA implemented in the Custom-ELECTRA (6-class) Classifier is able to correctly predict a small amount of instances of understatement and overstatement examples, whilst also attaining superior results for the satire, irony, and rhetorical question classes which are relatively under-represented in the dataset. It is interesting that the fine-tuned version of ELECTRA does not perform better than the base model when predicting the 'Sarcasm' class despite being fine-tuned on sarcastic data, though one consideration for this could be that ELECTRA's approach to both replaced token detection and discriminatory MLM aids the model generally in understanding sarcastic context more efficiently, and is thus able to identify more nuanced samples within the dataset when fine-tuned on general sarcastic data.

Overall, approaching the task as multi-class as opposed to multi-label as defined in Semeval Subtask B increased performance across less well-represented subcategories, for which F1 Scores in understatement, overstatement, and satire were generally not consistently identified by the original participants' models [1]. Although it is important not to quantitatively compare the results obtained in this study in comparison with those obtained by the original Semeval Subtask B participants, it is possible that the multi-class approach outlined in this project elicited more consistent results across all subcategories: Approaching this classification task as multi-class as opposed to multi-label necessitated a greater inspection of the Semeval data, which clearly illustrated the overlap between the highest-represented 'Sarcasm' and 'Irony' classes. Eliminating hierarchical overlap between the labels therefore allowed more under-represented classes to be given more attention by the model during training.

[1] A full copy of the results attained by Semeval 2022 participants can be found in Appendix C of the report.

The experimental 7-Class Custom ELECTRA Classifier results (Table 5.4) highlight that the creation of a model which is both capable of distinguishing between sarcastic and non-sarcastic data samples as well as classifying instances of the sarcastic subcategories remains a difficult task. The non-sarcastic samples in the Semeval dataset comprise a total of 2601 out of 3468 samples, or 75% of the dataset. This experimental implementation highlights that there is a large bias towards non-sarcastic samples in the original dataset, which elicits seemingly highly performant results in the 7-Class Classifier in predicting instances of the majority-represented classes: 'Sarcasm' and 'not_sarcastic'. It could therefore be concluded that, when trained on the full dataset, these biases have an impact on the representations which models are able to derive from this dataset, leading to the model defaulting to predictions made in favour of the majority classes in order to achieve higher scores.

The Aggregate-ELECTRA results attained when aggregating the minority classes into one column and creating a majority non-sarcastic class represent the highest macro F1 Score attained by any of the Custom-ELECTRA (6-Class) Classifier, the Aggregate-ELECTRA (3-Class) Classifier, and ELECTRA-no-FT. This model attains the most superior results when predicting 'Sarcasm', while predicting values for 'not_sarcastic' and 'Other' relatively well. The results attained by this model suggest that during classification, the model is able to distinguish between sarcasm, not sarcastic, and a category which relates to the two relatively well, strengthening the potential argument for the creation of multiple models trained primarily for sarcasm detection and subsequently for subcategory classification as a related downstream task. The comparatively low F1 Score attained for the 'Other' category also suggests that more samples for the under-represented categories are required despite their aggregation into one class.

Chapter 6

Reflections

6.1 Evaluation

This project sought to assess the ELECTRA model's efficacy when dealing with the task of multi-class classification of sarcasm and its subcategories, utilising the dataset afforded by the Semeval 2022 Subtask B. Initial implementation identified that the ELECTRA base model was unable to effectively train and predict on the dataset. It was therefore hypothesised that the creation of a fine-tuned version of the model, trained to generalise on sarcastic data, could aid in producing a secondary custom model which was able to more successfully predict the class labels on the dataset.

6.1.1 Model performance

Implementation of a fine-tuned version of ELECTRA trained for the downstream task of sarcasm subcategory classification resulted in a model which was able to predict values for all classes, while providing values for the samples least represented in the dataset. Custom-ELECTRA (6-Class), the fine-tuned version of ELECTRA, compared with the base ELECTRA model ELECTRA-no-FT, is able to identify more correct samples of the most under-represented categories: Satire, understatement, and overstatement, though the macro F1 Score overall is inferior to that obtained by ELECTRA-no-FT. It is possible therefore that ELECTRA's discriminatory approach to MLM helps the model to understand more contextual nuances in under-represented dataset samples when fine-tuned for a particular task.

Although the results obtained do not surpass those obtained by participants in the multi-label Semeval task, it is interesting to note that the fine-tuning of the ELECTRA base model appeared to produce some improvement in the ELECTRA model's consistent categorisation of minority under-represented classes in the dataset. Without further contextual information regarding any other ELECTRA models implemented for this particular task, it is not possible to draw definitive conclusions regarding this model's results in comparison to other versions. Due to the increase in model performance following the application of data augmentation techniques, it is possible that an increase in data samples accurately reflecting a variety of class instances could lead to an overall increase in ELECTRA's performance on the task.

6.1.2 Evaluation of overall implementation techniques

The best-scoring participating team in Semeval Subtask B utilised an ensemble learning approach in which the predictions of multiple models trained on separate, relevant tasks are averaged to attain superior results Du et al. (2022). The experimental results of the 7-Class Custom ELECTRA Classifier, which displayed the model defaulting to prediction of the majority classes, suggest that separating the tasks of *(A)* binary sarcasm detection and *(B)* multi-class sarcasm subcategory classification could be beneficial for obtaining superior results. It is possible therefore that using the fine-tuned version of ELECTRA to identify instances of sarcasm, combining this using ensemble methods with a separate model to identify instances of the subcategories of sarcasm, could also lead to superior results. Separating the more prevalent classes such as sarcasm and not sarcasm into one model, and linking this with another model for the subcategories could ensure that these models encapsulate more granular linguistic nuances derived from the data. Additionally, it is possible that creating a smaller trainable neural network between the fine-tuned ELECTRA base model and the Custom ELECTRA Classifier classification head could lead to better results due to more layers being trainable, though initial experimentation with this method did not lead to performance improvements in this study. The current research endeavoured to focus primarily on the performance of an unaltered ELECTRA model on the task of multi-class sarcasm subcategory classification.

6.2 Reflections: the Data Problem

As referenced several times throughout the report, the class imbalance and bias within the Semeval dataset elicited numerous difficulties when trying to train and test the various ELECTRA iterations outlined. With the most under-represented classes in the dataset of satire, understatement, and overstatement representing only 25, 40, and 10 samples respectively, it was necessary for data augmentation techniques to be applied to the dataset in order to aid the model in extracting valuable semantic information from the data.

6.2.1 Consideration of data augmentation techniques

The research conducted identified that data augmentation techniques applied to the Semeval dataset had a marked effect on improving results, leading to predictions made on several classes which had previously been unidentifiable for the model. The choice of utilising the TextAttack library for data augmentation increased results. TextAttack's character-based synthetic data augmentation was identified as the method which would least fragment the existing data, by only altering characters and thus preserving nuance and semantics in the existing samples.

Consideration was given to the theory that simply augmenting existing class samples in the dataset - some of which comprise a limited amount of samples - could lead to overfitting on these particular classes, while preventing the model from being correctly equipped to handle variations of under-represented subcategories when utilised in a real-life setting. Data augmentation through character alteration increases the size of the dataset, but does not necessarily provide the model with a wide variety of samples from which to draw information and may even introduce biases towards particular sample instances. This approach is therefore not necessarily the most optimal for training intelligent NLU systems to comprehensively understand and identify instances of sarcasm and its subcategories. However, due to a lack of labelled corpora for the task, this hurdle remains insurmountable.

In addition to character-based data augmentation, consideration was also given to a wide range of techniques which could also have been implemented in order to improve results, including using more complex data augmentation techniques such as back translation - in which target sentences are translated to a foreign language and then translated back into the original language to diversify dataset samples; or word embedding synonym replacement - in which synonyms close in meaning to the original word are generated by the word embedding generation models and used to replace words. However, it was decided that significantly altering the nuances of already under-represented sentences - many of which are also short in length - could lead to inferior results.

6.2.2 Data availability conclusions

As such, the highly specialised task of data availability remains a significant issue for NLP and in particular sarcasm detection. The dataset collated by Abu Farha et al. (2022) is impressive, and comprises the first of its kind for an aggregate corpora of sarcastic subcategories, however it has its limitations. Data augmentation, while an important strategy in training and evaluating neural architectures for the task of sarcasm detection generally, is not effective enough in equipping these models to identify a wide range of examples of sarcasm and its subcategories. Therefore, it is recommendable that larger corpora of labelled data should be compiled for these subcategories in order to effectively train models for the task of sarcasm subcategory classification in supervised settings.

6.3 Future Work

Implementation of Custom-ELECTRA and its iterations involved a fine-tuned version of the base ELECTRA model, as instantiated in the ELECTRA Classifier module. The results attained illustrate that fine-tuning a model for a more general task can aid in increasing performance for under-represented class samples which include heavy semantic and linguistic nuances. However, these results are still far from state-of-the-art, and illustrate that significant work can still be conducted when fine-tuning pre-trained LLMs such as ELECTRA for multi-class classification of sarcasm subcategories.

To this end, there is room for research to be conducted relating to the improvement of results attained by LLMs in general, and to establish a superior ELECTRA benchmark specifically in the task. ELECTRA represents a movement by researchers to develop more lightweight and computationally efficient LLMs, which can attain comparable or superior results to larger LLMs such as BERT or GPT. However, it remains to be seen whether ELECTRA can still outperform these models on more nuanced tasks, possibly due to fewer contextual embeddings generated by the model during pre-training. It would be interesting to see how an ensemble learning method utilising several versions of ELECTRA fine-tuned on a variety of sarcastic speech categories or indeed trained to first identify sarcasm and then identify the subcategories would perform in comparison to a single model, and whether this approach could attain results superior to those delineated in this study.

The limitations of the Semeval dataset, primarily relating to biases and under-representation of nuanced minority classes highlight the urgency for the community to create larger corpora of labelled data for particular tasks. Although it would be possible to fine-tune several versions of the ELECTRA model on each of the subcategories outlined within the dataset, questions remain surrounding how equipped such a model architecture would be to handle samples of such subcategories which are not represented in this dataset. There is also room for the development of more lightweight, targeted models which are trained to attain high results for particular tasks - as opposed to attempting to develop more LLMs which use huge amounts of compute - in a more generative approach to creating and studying NLP models for sarcasm detection.

Bibliography

Abdul-Mageed, M., Elmadany, A.A. and Nagoudi, E.M.B., 2021. ARBERT & MARBERT: deep bidirectional transformers for arabic. *Corr* [Online], abs/2101.01785. 2101.01785, Available from: https://arxiv.org/abs/2101.01785.

Abu Farha, I. and Magdy, W., 2021. Benchmarking transformer-based language models for Arabic sentiment and sarcasm detection [Online]. *Proceedings of the sixth arabic natural language processing workshop*. Kyiv, Ukraine (Virtual): Association for Computational Linguistics, pp.21–31. Available from: https://aclanthology.org/2021.wanlp-1.3.

Abu Farha, I., Oprea, S.V., Wilson, S. and Magdy, W., 2022. SemEval-2022 task 6: iSarcasmEval, intended sarcasm detection in English and Arabic [Online]. *Proceedings of the 16th international workshop on semantic evaluation (semeval-2022)*. Seattle, United States: Association for Computational Linguistics, pp.802–814. Available from: https://doi.org/10.18653/v1/2022.semeval-1.111.

B, M.J., J, A.A.S., M, A.R.S. and Rajan, R., 2023. Efficacy of electra-based language model in sentiment analysis [Online]. *2023 international conference on intelligent systems for communication, iot and security (iciscois)*. pp.682–687. Available from: https://doi.org/10.1109/ICISCoIS56541.2023.10100342.

Baruah, A., Das, K., Barbhuiya, F. and Dey, K., 2020. Context-aware sarcasm detection using BERT [Online]. *Proceedings of the second workshop on figurative language processing*. Online: Association for Computational Linguistics, pp.83–87. Available from: https://doi.org/10.18653/v1/2020.figlang-1.12.

Bhakuni, M., Kumar, K., Garg, S., Iwendi, C. and Singh, A., 2022. Evolution and evaluation: Sarcasm analysis for twitter data using sentiment analysis. *Journal of sensors* [Online], 2022. Available from: https://doi.org/10.1155/2022/6287559.

Briggs, J., 2021. *Electra is bert - supercharged* [Online]. Towards Data Science. Available from: https://towardsdatascience.com/electra-is-bert-supercharged-b450246c4edb [Accessed 2023-02-01].

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I. and Amodei, D., 2020. Language models are few-shot learners. *Corr* [Online], abs/2005.14165. Available from: https://arxiv.org/abs/2005.14165.

Cambridge English Dictionary, 2023. Sarcasm [Online]. Available from: https://dictionary.cambridge.org/dictionary/english/sarcasm [Accessed 2023-06-23].

Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L. and Lopez, A., 2020. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing* [Online], 408, pp.189–215. Available from: `https://doi.org/https://doi.org/10.1016/j.neucom.2019.10.118`.

Chatterjee, N., Aggarwal, T. and Maheshwari, R., 2020. *Sarcasm detection using deep learning-based techniques* [Online], Singapore: Springer Singapore, pp.237–258. Available from: `https://doi.org/10.1007/978-981-15-1216-2_9`.

Chavan, G., Manjare, S., Hegde, P. and Sankhe, A., 2014. A survey of various machine learning techniques for text classification. *International journal of engineering trends and technology* [Online], 15, pp.288–292. Available from: `https://doi.org/10.14445/22315381/IJETT-V15P255`.

Claburn, T., 2023. *Github, microsoft, openai fail to wriggle out of copilot copyright lawsuit* [Online]. The Register. Available from: `https://www.theregister.com/2023/05/12/github_microsoft_openai_copilot/` [Accessed 2023-05-27].

Clark, K., Luong, M., Le, Q.V. and Manning, C.D., 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *Corr* [Online], abs/2003.10555. Available from: `https://arxiv.org/abs/2003.10555`.

Climate Watch and GHG Emissions, 2020. Co2 emissions (metric tons per capita) [Online]. Available from: `https://data.worldbank.org/indicator/EN.ATM.CO2E.PC?end=2019&start=1990&view=chart` [Accessed 2023-06-19].

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding [Online]. *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp.4171–4186. Available from: `https://doi.org/10.18653/v1/N19-1423`.

Du, X., Hu, D., Zhi, J., Jiang, L. and Shi, X., 2022. PALI-NLP at SemEval-2022 task 6: iSarcasmEval- fine-tuning the pre-trained model for detecting intended sarcasm [Online]. *Proceedings of the 16th international workshop on semantic evaluation (semeval-2022)*. Seattle, United States: Association for Computational Linguistics, pp.815–819. Available from: `https://doi.org/10.18653/v1/2022.semeval-1.112`.

El Mahdaouy, A., El Mekki, A., Essefar, K., Skiredj, A. and Berrada, I., 2022. CS-UM6P at SemEval-2022 task 6: Transformer-based models for intended sarcasm detection in English and Arabic [Online]. *Proceedings of the 16th international workshop on semantic evaluation (semeval-2022)*. Seattle, United States: Association for Computational Linguistics, pp.844–850. Available from: `https://doi.org/10.18653/v1/2022.semeval-1.117`.

Espejel, J.L., Ettifouri, E.H., Sanoussi, M., Alassan, Y., Chouham, E.M. and Dahhane, W., 2023. Gpt-3.5 vs gpt-4: Evaluating chatgpt's reasoning performance in zero-shot learning [Online]. [Online]. Available from: `https://arxiv.org/pdf/2305.12477.pdf`.

Godara, J. and Aron, R., 2021. Support vector machine classifier with principal component analysis and k mean for sarcasm detection [Online]. *2021 7th international conference on advanced computing and communication systems (icaccs)*. vol. 1, pp.571–576. Available from: `https://doi.org/10.1109/ICACCS51430.2021.9442033`.

Godara, J., Batra, I. and Aron, R., 2021. Ensemble classification approach for sarcasm detection. vol. 2021. Available from: https://doi.org/10.1155/2021/9731519.

Hugging Face, n.d.a. *Electra* [Online]. Hugging Face. Available from: https://huggingface.co/docs/transformers/model_doc/electra [Accessed 2023-05-27].

Hugging Face, n.d.b. Electra [Online]. Available from: https://huggingface.co/docs/transformers/model_doc/electra [Accessed 2023-05-27].

Hugging Face, n.d.c. *Wordpiece tokenization* [Online]. Hugging Face. Available from: https://huggingface.co/learn/nlp-course/chapter6/6?fw=pt [Accessed 2023-05-27].

Jones, K.S., 1994. Natural language processing: A historical review [Online]. In: A. Zampolli, N. Calzolari and M. Palmer, eds. *Current issues in computational linguistics: In honour of don walker*. Springer Dodrecht, Linguistica Computazionale, Vol. 9-10, pp.3–16. 1st ed. Available from: https://doi.org/10.1007/978-0-585-35958-8.

Joshi, A., Bhattacharyya, P. and Carman, M.J., 2016. Automatic sarcasm detection: A survey. *Corr* [Online], abs/1602.03426. 1602.03426, Available from: http://arxiv.org/abs/1602.03426.

Joshi, A., Bhattacharyya, P. and Carman, M.J., 2018. *Investigations in computational sarcasm* [Online], Cognitive Systems Monographs. 1st ed. Springer Singapore. Available from: https://doi.org/10.1007/978-981-10-8396-9.

Korstanje, J., 2021. *The f1 score* [Online]. Towards Data Science. Available from: https://towardsdatascience.com/the-f1-score-bec2bbc38aa6 [Accessed 2023-06-15].

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P. and Soricut, R., 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *Corr* [Online], abs/1909.11942. 1909.11942, Available from: http://arxiv.org/abs/1909.11942.

Lee, M., 2023. A mathematical investigation of hallucination and creativity in gpt models. *Mathematics* [Online], 11(10). Available from: https://doi.org/10.3390/math11102320.

Liddy, E.D., 2001. Natural language processing [Online]. *Encyclopedia of library and information science*. NY: Marcel Decker, Inc. 2nd ed. Available from: https://surface.syr.edu/cgi/viewcontent.cgi?article=1043&context=istpub.

Liu, B., 2020. *Sentiment analysis: mining opinions, sentiments, and emotions* [Online]. 2nd ed. Cambridge: Cambridge University Press. Available from: https://doi.org/10.1017/9781108639286.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J. and Han, J., 2019. On the variance of the adaptive learning rate and beyond. *Corr* [Online], abs/1908.03265. Available from: http://arxiv.org/abs/1908.03265.

Loshchilov, I. and Hutter, F., 2016. SGDR: stochastic gradient descent with restarts. *Corr* [Online], abs/1608.03983. Available from: http://arxiv.org/abs/1608.03983.

Misra, R., 2022. News category dataset [Online]. Available from: https://doi.org/10.48550/arXiv.2209.11429.

Misra, R. and Grover, J., 2021. *Sculpting data for ml: The first act of machine learning* [Online]. Available from: https://rishabhmisra.github.io/Sculpting_Data_for_ML.pdf.

Morris, J.X., Lifland, E., Yoo, J.Y. and Qi, Y., 2020. Textattack: A framework for adversarial attacks in natural language processing. *Corr* [Online], abs/2005.05909. 2005.05909, Available from: https://arxiv.org/abs/2005.05909.

Nashold, L., 2021. Cascade + bert : Using context embeddings and transformers to predict [Online]. Stanford University. Available from: https://web.stanford.edu/class/cs224n/reports/final_reports/report023.pdf.

Nigam, S.K. and Shaheen, M., 2022. Plumeria at semeval-2022 task 6: Robust approaches for sarcasm detection for english and arabic using transformers and data augmentation [Online]. Available from: https://doi.org/10.48550/arXiv.2203.04111.

OpenAI, 2022. *Introducing chatgpt* [Online]. Open AI. Available from: https://openai.com/blog/chatgpt [Accessed 2023-01-21].

Perrigo, B., 2023. *Exclusive: Openai used kenyan workers on less than usd 2 per hour to make chatgpt less toxic* [Online]. Time Magazine. Available from: https://time.com/6247678/openai-chatgpt-kenya-workers/ [Accessed 2023-03-14].

Pytorch, n.d.a. *Cosineannealingwarmrestarts* [Online]. Pytorch. Available from: https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.CosineAnnealingWarmRestarts.html [Accessed 2023-06-10].

Pytorch, n.d.b. *Onecyclelr* [Online]. Pytorch. Available from: https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.OneCycleLR.html [Accessed 2023-06-10].

PyTorch Lightning, n.d. Welcome to pytorch lightning [Online]. Available from: https://lightning.ai/docs/pytorch/stable/ [Accessed 2023-06-01].

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I., 2019. Language models are unsupervised multitask learners [Online]. Available from: https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Rafail, P. and Freitas, I., 2020. *Natural language processing* [Online]. London: SAGE Publications Ltd. Available from: https://doi.org/10.4135/9781526421036879118.

Semeval, n.d. [Online]. Available from: https://semeval.github.io/ [Accessed 2023-06-01].

Shang, E., 2023. *In generative ai legal wild west, the courtroom battles are just getting started* [Online]. CNBC. Available from: https://www.cnbc.com/2023/04/03/in-generative-ai-legal-wild-west-lawsuits-are-just-getting-started.html [Accessed 2023-05-27].

Sharir, O., Peleg, B. and Shoham, Y., 2020. The cost of training NLP models: A concise overview. *Corr* [Online], abs/2004.08900. 2004.08900, Available from: https://arxiv.org/abs/2004.08900.

Smith, L.N. and Topin, N., 2018. Super-convergence: Very fast training of neural networks using large learning rates [Online]. Available from: https://doi.org/10.48550/arXiv.1708.07120.

So, D.R., Liang, C. and Le, Q.V., 2019. The evolved transformer. *Corr* [Online], abs/1901.11117. 1901.11117, Available from: http://arxiv.org/abs/1901.11117.

Song, X. and Zhou, D., 2021. *A fast wordpiece tokeization system* [Online]. Google. Available from: https://ai.googleblog.com/2021/12/a-fast-wordpiece-tokenization-system.html [Accessed 2023-05-27].

Stanford University, n.d.a. *Linear classification: Softmax classifier* [Online]. CS231n Convolutional Neural Networks for Visual Recognition. Stanford University. Available from: https://cs231n.github.io/linear-classify/#softmax [Accessed 2023-06-15].

Stanford University, n.d.b. *Natural language processing - history* [Online]. Stanford University. Available from: https://cs.stanford.edu/people/eroberts/courses/soco/projects/2004-05/nlp/overview_history.html [Accessed 2023-05-23].

Stanford University, n.d.c. *Tokenization* [Online]. Stanford University. Available from: https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html [Accessed 2023-05-27].

Strubell, E., Ganesh, A. and McCallum, A., 2019. Energy and policy considerations for deep learning in NLP [Online]. *Proceedings of the 57th annual meeting of the association for computational linguistics*. Florence, Italy: Association for Computational Linguistics, pp.3645–3650. Available from: https://doi.org/10.18653/v1/P19-1355.

Subramanian, J., Sridharan, V., Shu, K. and Liu, H., 2019. Exploiting emojis for sarcasm detection [Online]. *Social, cultural, and behavioral modeling: 12th international conference, sbp-brims 2019, washington, dc, usa, july 9–12, 2019, proceedings 12*. Springer, pp.70–80. Available from: https://doi.org/10.1007/978-3-030-21741-9_8.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P. and Hashimoto, T.B., 2023. *Alpaca: A strong, replicable instruction-following model* [Online]. Stanford University. Available from: https://crfm.stanford.edu/2023/03/13/alpaca.html [Accessed 2023-04-20].

Tay, Y., Dehghani, M., Bahri, D. and Metzler, D., 2020. Efficient transformers: A survey. *Corr* [Online], abs/2009.06732. 2009.06732, Available from: https://arxiv.org/abs/2009.06732.

Tepperman, J., Traum, D. and Narayanan, S., 2006. Yeah right: Sarcasm recognition for spoken dialogue systems [Online]. Available from: https://doi.org/10.21437/Interspeech.2006-507.

Thelwall, M., 2020. *Sentiment analysis* [Online]. Sage Research Methods Foundations. Available from: https://doi.org/10.4135/9781526421036754533.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E. and Lample, G., 2023. Llama: Open and efficient foundation language models [Online]. Available from: https://doi.org/10.48550/arXiv.2302.13971.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and

Polosukhin, I., 2017. Attention is all you need. *Corr* [Online], abs/1706.03762. 1706.03762, Available from: http://arxiv.org/abs/1706.03762.

Vincent, J., 2023. *Ai art tools stable diffusion and midjourney targeted with copyright lawsuit* [Online]. The Verge. Available from: https://www.theverge.com/2023/1/16/23557098/generative-ai-art-copyright-legal-lawsuit-stable-diffusion-midjourney-deviantart [Accessed 2023-05-27].

Wallace, B.C., Choe, D.K., Kertz, L. and Charniak, E., 2014. Humans require context to infer ironic intent (so computers probably do, too) [Online]. *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)*. Baltimore, Maryland: Association for Computational Linguistics, pp.512–516. Available from: https://doi.org/10.3115/v1/P14-2084.

Yaghoobian, H., Arabnia, H.R. and Rasheed, K., 2021. Sarcasm detection: A comparative study. *Corr* [Online], abs/2107.02276. Available from: https://arxiv.org/abs/2107.02276.

Zhang, M., Zhang, Y. and Fu, G., 2016. Tweet sarcasm detection using deep neural network [Online]. *International conference on computational linguistics*. Available from: https://aclanthology.org/C16-1231.pdf.

Zhang, Z., Gu, Y., Han, X., Chen, S., Xiao, C., Sun, Z., Yao, Y., Qi, F., Guan, J., Ke, P., Cai, Y., Zeng, G., Tan, Z., Liu, Z., Huang, M., Han, W., Liu, Y., Zhu, X. and Sun, M., 2021. Cpm-2: Large-scale cost-effective pre-trained language models. *Ai open* [Online], 2, pp.216–224. Available from: https://doi.org/https://doi.org/10.1016/j.aiopen.2021.12.003.

Appendix A

Packages and Extensions

Several common machine learning packages and extensions were utilised during research in order to ensure reproducibility of results and expedite implementation. This section will provide an overview of the packages and extensions utilised and evaluate their suitability for this task.

A.0.1 Hugging Face

The pre-trained version of ELECTRA used in fine-tuning and transfer learning for the task was downloaded from the Hugging Face Transformers library. Hugging Face provides an open source framework to employ pre-trained transformer models, and thus is an optimal choice for research.

The ELECTRA base model chosen from the Hugging Face library is the ElectraForSequenceClassification class, a version of ELECTRA which has been pre-trained for sequence classification tasks Hugging Face (n.d.b). Sequence classification encompasses a task in which a category is predicted based on a sequence of inputs or data over space or time, and is widely used in NLP for classification of sentences or inputs, including in sentiment analysis. The ElectraForSequenceClassification was deemed to be most applicable to this task due to the relevance of sequence classification in sentiment analysis.

Evaluation of HuggingFace

The ease of use and integration with common machine learning frameworks such as Pytorch and Tensorflow afforded by the Hugging Face Transformers library makes it an invaluable addition for this research. Although all ELECTRA models published by Google Research are available on the company's repository, they only provide integration with the Tensorflow framework which is difficult to use when attempting to save and load fine-tuned pre-trained models.

A.0.2 Pytorch Lightning

The difficulties with saving and loading fine-tuned pre-trained models using Tensorflow gave rise to the necessity to use Pytorch and, more specifically, the Pytorch Lightning library. This library was created for "professional AI researchers" PyTorch Lightning (n.d.) in order to expedite experimentation and research. Pytorch Lightning ensures that results obtained during research is reproducible, and was therefore an optimal solution in this project's implementation.

Evaluation of Pytorch Lightning

Pytorch Lightning's inbuilt functionality and streamlined processes facilitated experimentation and greater iterations through ideas and various implementations during research. The modular functionality afforded great ease when creating classes for the Electra Classifier (base model fine tuning) and the Custom Electra Classifier (downstream task model), whilst ensuring that training iterations could be done quickly. However, the high level of abstraction with which the library operates made it sometimes difficult to understand what was going on and where, delaying progress in some areas.

A.0.3 TextAttack

Poor performance in model implementation initially necessitated the creation of synthetic but similar samples to those included in the Semeval dataset. Thus, the TextAttack library was employed. TextAttack was initially proposed by Morris et al. (2020) in 2020 as a machine learning framework to be employed for a variety of use cases such as data augmentation and adversarial attacks or training. TextAttack's data augmentation classes allow custom synthetic data samples to be created per sample which are similar to existing samples in the dataset, by altering words, characters, or even replacing words with words closely associated in vector embedding space.

Evaluation of TextAttack

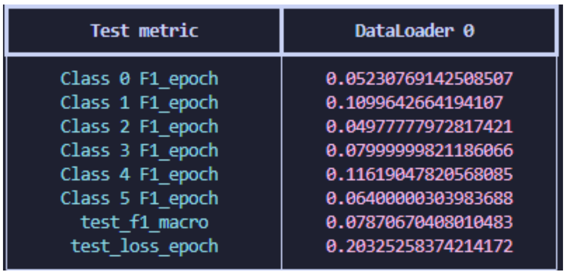
The implementation of TextAttack for this project was invaluable, as it allowed the creation of synthetic data samples without *(A)* utilising other LLMs such as GPT and *(B)* altering the existing nuances or semantics of the existing - and relatively limited - samples. Conventional data augmentation techniques such as word replacement, swapping, or removal would have potentially altered the meaning behind the sentences of under-represented categories, leading to the model being unable to derive meaningful representations from the data. However, it is also possible that utilising such data augmentation techniques does not allow for the model to learn meaningful representations which can apply across a wide variety of real-life samples, though due to the general lack of specialised data for the task, this remains a challenge in general for research in this area.

Appendix B

Raw Results Output

B.0.1 Results Screenshots

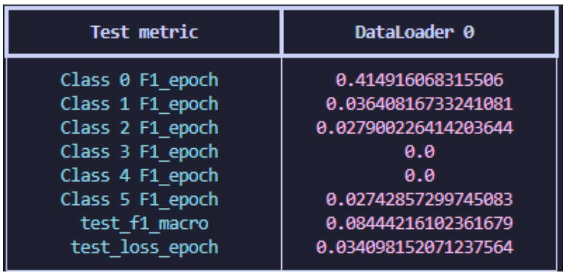
The following figures provide screenshots of output results for the Custom ELECTRA Classifier and its iterations which have already been detailed in Chapter 5.



```
      Test metric              DataLoader 0

   Class 0 F1_epoch        0.05230769142508507
   Class 1 F1_epoch        0.1099642664194107
   Class 2 F1_epoch        0.04977777972817421
   Class 3 F1_epoch        0.07999999821186066
   Class 4 F1_epoch        0.11619047820568085
   Class 5 F1_epoch        0.06400000303983688
     test_f1_macro         0.07870670408010483
    test_loss_epoch        0.20325258374214172
```

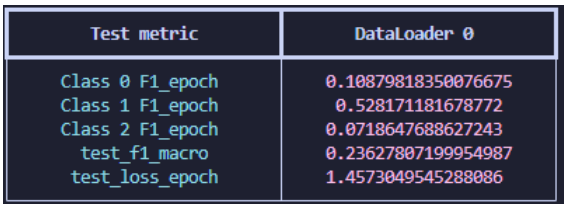
Figure B.1: Custom-ELECTRA (6-Class) Classifier Results



```
      Test metric              DataLoader 0

   Class 0 F1_epoch         0.414916068315506
   Class 1 F1_epoch        0.03640816733241081
   Class 2 F1_epoch       0.027900226414203644
   Class 3 F1_epoch               0.0
   Class 4 F1_epoch               0.0
   Class 5 F1_epoch        0.02742857299745083
     test_f1_macro         0.08444216102361679
    test_loss_epoch       0.034098152071237564
```

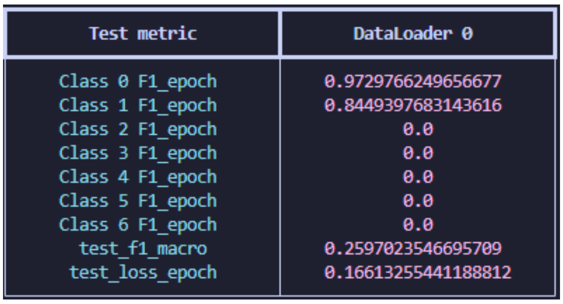
Figure B.2: ELECTRA-no-FT Results



```
      Test metric              DataLoader 0

   Class 0 F1_epoch        0.10879818350076675
   Class 1 F1_epoch         0.528171181678772
   Class 2 F1_epoch        0.0718647688627243
     test_f1_macro         0.23627807199954987
    test_loss_epoch        1.4573049545288086
```

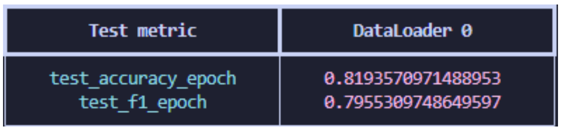
Figure B.3: Aggregate-ELECTRA (3-Class) Results



Test metric	DataLoader 0
Class 0 F1_epoch	0.9729766249656677
Class 1 F1_epoch	0.8449397683143616
Class 2 F1_epoch	0.0
Class 3 F1_epoch	0.0
Class 4 F1_epoch	0.0
Class 5 F1_epoch	0.0
Class 6 F1_epoch	0.0
test_f1_macro	0.2597023546695709
test_loss_epoch	0.16613255441188812

Figure B.4: Custom-ELECTRA Classifier (7-class) Results

For reference, the results of the ELECTRA Classifier, the model fine-tuned to generalise on sarcastic data using the News Headlines Dataset, can also be found below:



Test metric	DataLoader 0
test_accuracy_epoch	0.8193570971488953
test_f1_epoch	0.7955309748649597

Figure B.5: Fine-tuned ELECTRA Classifier Results

B.0.2 Loss Graphs

The following screenshots denote the training and validation loss on the training and validation dataset for the models:

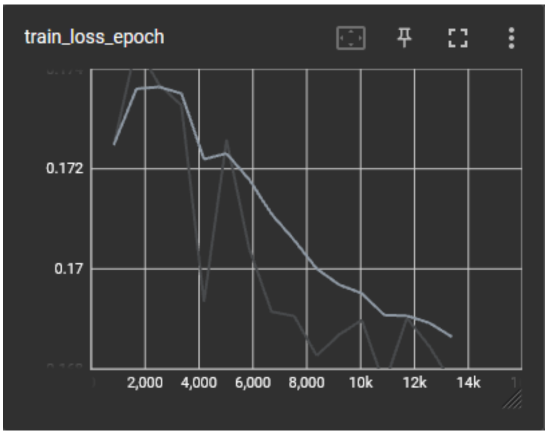


Figure B.6: Custom-ELECTRA (6-Class) Classifier Training Loss

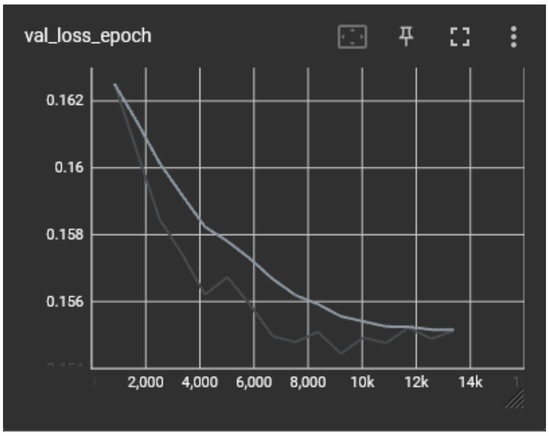


Figure B.7: Custom-ELECTRA (6-Class) Classifier Validation Loss

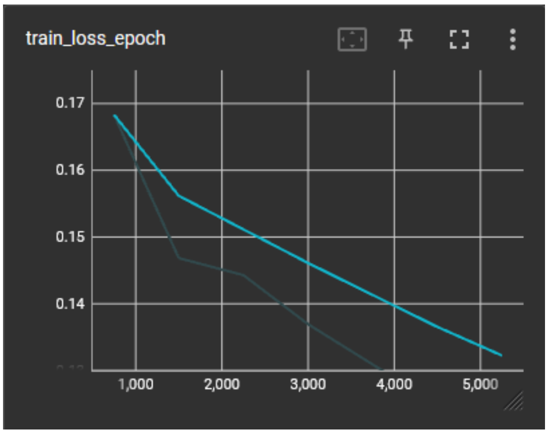


Figure B.8: ELECTRA-no-FT Training Loss

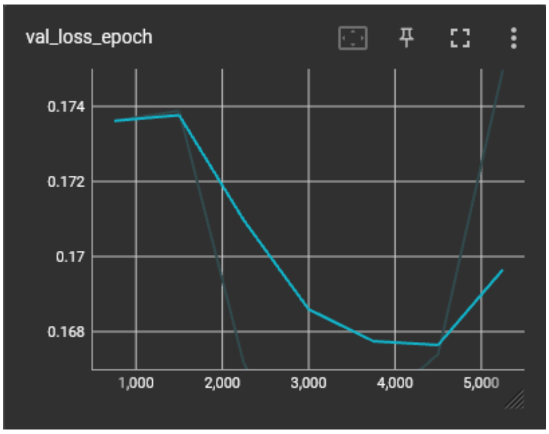


Figure B.9: ELECTRA-no-FT Validation Loss

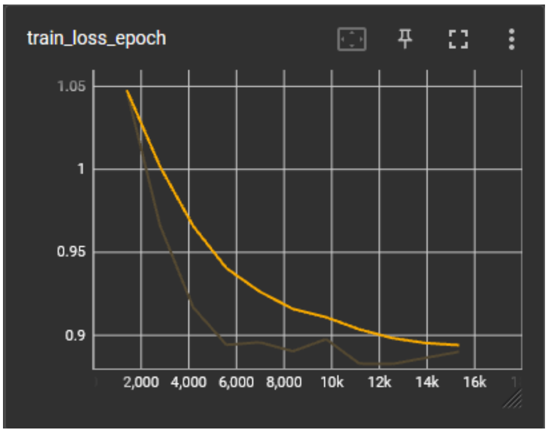


Figure B.10: Aggregate-ELECTRA (3-Class) Training Loss

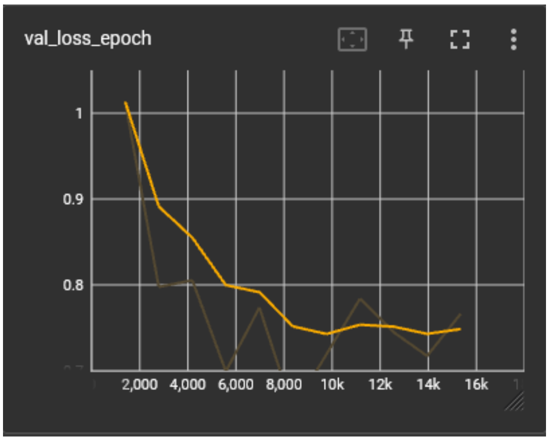
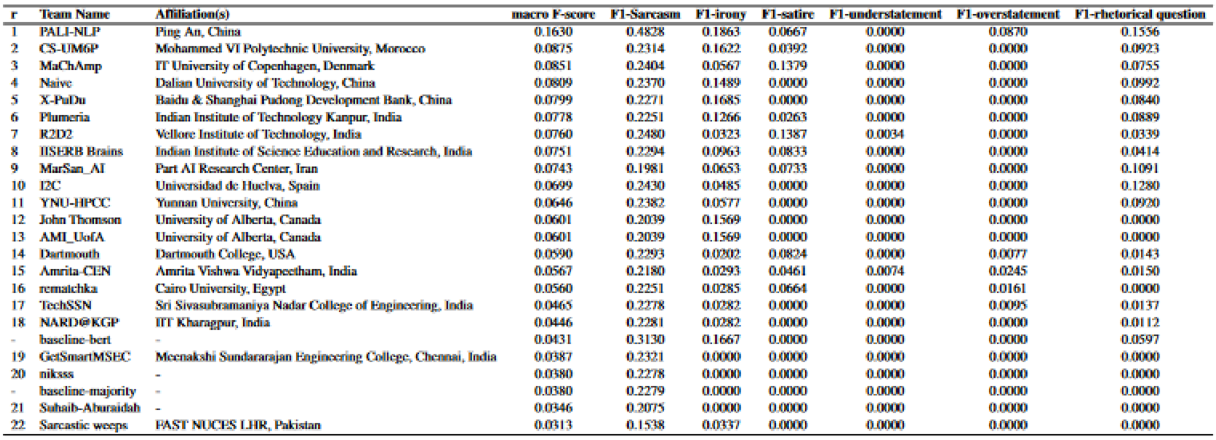


Figure B.11: Aggregate-ELECTRA (3-Class) Validation Loss

Appendix C

Semeval 2022 Subtask B Full Results



r	Team Name	Affiliation(s)	macro F-score	F1-Sarcasm	F1-irony	F1-satire	F1-understatement	F1-overstatement	F1-rhetorical question
1	PALI-NLP	Ping An, China	0.1630	0.4828	0.1863	0.0667	0.0000	0.0870	0.1556
2	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.0875	0.2314	0.1622	0.0392	0.0000	0.0000	0.0923
3	MaChAmp	IT University of Copenhagen, Denmark	0.0851	0.2404	0.0567	0.1379	0.0000	0.0000	0.0755
4	Naive	Dalian University of Technology, China	0.0809	0.2370	0.1489	0.0000	0.0000	0.0000	0.0992
5	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.0799	0.2271	0.1685	0.0000	0.0000	0.0000	0.0840
6	Plumeria	Indian Institute of Technology Kanpur, India	0.0778	0.2251	0.1266	0.0263	0.0000	0.0000	0.0889
7	R2D2	Vellore Institute of Technology, India	0.0760	0.2480	0.0323	0.1387	0.0034	0.0000	0.0339
8	IISERB Brains	Indian Institute of Science Education and Research, India	0.0751	0.2294	0.0963	0.0833	0.0000	0.0000	0.0414
9	MarSan_AI	Part AI Research Center, Iran	0.0743	0.1981	0.0653	0.0733	0.0000	0.0000	0.1091
10	I2C	Universidad de Huelva, Spain	0.0699	0.2430	0.0485	0.0000	0.0000	0.0000	0.1280
11	YNU-HPCC	Yunnan University, China	0.0646	0.2382	0.0577	0.0000	0.0000	0.0000	0.0920
12	John Thomson	University of Alberta, Canada	0.0601	0.2039	0.1569	0.0000	0.0000	0.0000	0.0000
13	AMI_UofA	University of Alberta, Canada	0.0601	0.2039	0.1569	0.0000	0.0000	0.0000	0.0000
14	Dartmouth	Dartmouth College, USA	0.0590	0.2293	0.0202	0.0824	0.0000	0.0077	0.0143
15	Amrita-CEN	Amrita Vishwa Vidyapeetham, India	0.0567	0.2180	0.0293	0.0461	0.0074	0.0245	0.0150
16	rematchka	Cairo University, Egypt	0.0560	0.2251	0.0285	0.0664	0.0000	0.0161	0.0000
17	TechSSN	Sri Sivasubramaniya Nadar College of Engineering, India	0.0465	0.2278	0.0282	0.0000	0.0000	0.0095	0.0137
18	NARD@KGP	IIT Kharagpur, India	0.0446	0.2281	0.0282	0.0000	0.0000	0.0000	0.0112
-	baseline-bert	-	0.0431	0.3130	0.1667	0.0000	0.0000	0.0000	0.0597
19	GetSmartMSEC	Meenakshi Sundararajan Engineering College, Chennai, India	0.0387	0.2321	0.0000	0.0000	0.0000	0.0000	0.0000
20	niksss	-	0.0380	0.2278	0.0000	0.0000	0.0000	0.0000	0.0000
-	baseline-majority	-	0.0380	0.2279	0.0000	0.0000	0.0000	0.0000	0.0000
21	Suhaib-Aburaidah	-	0.0346	0.2075	0.0000	0.0000	0.0000	0.0000	0.0000
22	Sarcastic weeps	FAST NUCES LHR, Pakistan	0.0313	0.1538	0.0337	0.0000	0.0000	0.0000	0.0000

Figure C.1: Semeval 2022 Subtask B Results (All Participants)

(Abu Farha et al., 2022, p. 814)