Detecting emotion from text using deep learning

Oliver Aarnikoivu

A report submitted as part of the requirements for the degree
of BSc in Computer Science
at the School of Computing
Robert Gordon University
Aberdeen, Scotland

May 2020

Supervisor Dr. Eyad Elyan

Abstract

This report demonstrates the ability of deep learning models to detect multiple emotions from text. In particular, an experiment is carried out using an Attention-based bidirectional LSTM (Long Short Term Memory) model and Text CNN (Convolutional Neural Network on the SemEval Task 1: E-c multi-label classification dataset (Mohammad et al. 2018). Due to the limited amount of training data, a transfer learning approach is used such that both models are initialized using pre-trained GloVe (Global Vectors for Word Representation) word embeddings, as well as the learned representations (contextualized embeddings) produced by a frozen pre-trained BERT transformer model. The study shows that the Attention LSTM using the contextualized embeddings generated by BERT outperforms all other models and achieves a micro-averaged F1 score of 0.67. These results are comparable and competitive to the official SemEval Task 1: E-c multi-label classification task competition post-evaluation results. An application was developed to demonstrate how the Attention LSTM model using the embeddings produced by BERT captures the most important semantic information from a given input text.

Keywords: Attention LSTM; Text CNN; Transfer Learning; Transformer; GloVe; BERT

Acknowledgements

I would like to give a special thank you to my supervisor Dr. Eyad Elyan, who's support and guidance throughout this project has been invaluable. I would also like to thank the team at HuggingFace who have created the *transformers* library making it easy for machine learning enthusiasts, like myself, to easily gain access to state-of-the-art Natural Language Processing general-purpose architectures for PyTorch and TensorFlow.

Declaration

I confirm that the work contained in this BSc project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Signed Oliver Aarnikoivu..... Date 17/05/20..

Oliver Aarnikoivu

Contents

Abstract ii

Acknowledgements iii

Declaration iv

1 Introduction 1
- 1.1 Overview . . . 1
- 1.2 Aims and objectives . . . 2
- 1.3 Summary of key findings . . . 2
- 1.4 Chapter List . . . 2

2 Literature Review 4
- 2.1 Background . . . 4
  - 2.1.1 Understanding Emotion . . . 4
  - 2.1.2 Affective Computing . . . 5
  - 2.1.3 Computational Techniques for Emotion Detection . . . 6
- 2.2 Deep Learning . . . 7
  - 2.2.1 Sequence Models . . . 7
  - 2.2.2 Convolutional Neural Network . . . 8
- 2.3 Transfer Learning . . . 9
  - 2.3.1 Word Embeddings . . . 10
  - 2.3.2 Language Modeling . . . 11
  - 2.3.3 Attention Mechanisms & The Transformer . . . 12
- 2.4 Datasets . . . 13
  - 2.4.1 Labeled Text . . . 13
  - 2.4.2 Pre-trained Word Embeddings . . . 14
  - 2.4.3 Pre-trained Language Models . . . 15
- 2.5 Related Work . . . 16
- 2.6 Conclusions . . . 17

3 Methods & Experimental Setup **19**
3.1 Dataset . . . 19
3.2 Pre-trained Embeddings . . . 20
3.3 Pre-processing & Tokenization . . . 20
3.3.1 GloVe . . . 20
3.3.2 BERT . . . 21
3.4 Baselines . . . 21
3.4.1 Logistic Regression . . . 22
3.4.2 SVM . . . 22
3.5 Models . . . 22
3.5.1 CNN . . . 23
3.5.2 Attention LSTM . . . 24
3.6 Hyperparameters and Training . . . 26
3.7 Performance Evaluation . . . 28
3.8 Technologies . . . 29
3.9 Applications . . . 29

4 Experimental Results **31**
4.1 Results . . . 31
4.2 Baselines . . . 31
4.3 Attention LSTM vs Text CNN . . . 32
4.4 Leaderboard Comparison . . . 32
4.5 Results on Plutchik categories . . . 33
4.6 Summary . . . 34

5 Analysis & Discussion **35**
5.1 Training and Validation Loss / Accuracy . . . 35
5.2 Attention Visualization . . . 38
5.3 Model Performance on Longer Text . . . 39

6 Conclusion **43**
6.1 Conclusions . . . 43
6.2 Future Work . . . 44

7 Appendices **51**
7.1 Appendix A - Project Log . . . 51
7.2 Appendix B - Source Code . . . 52
7.3 Appendix C - Application . . . 52
7.4 Appendix D - Poster . . . 53
7.5 Appendix F - Project Proposal . . . 54

7.6 Appendix E - Ethics Form 61

List of Tables

2.1 ELMo results vs. previous state-of-the-art. 12
2.2 BERT vs ELMo on NLP tasks. 13
2.3 Word embedding model properties. 14
2.4 Pre-trained ELMo models. 15
2.5 Pre-trained BERT models. 16

3.1 Tweet pre-processing example. 20
3.2 Hyperparameters for Text CNN model using GloVe and BERT embeddings. 27
3.3 Hyperparameters for Attention LSTM model using GloVe and BERT embeddings. 27

4.1 Performance comparison of the baseline algorithms, and the Attention LSTM and Text CNN using GloVe and BERT embeddings. 31
4.2 SemEval 2018: Task 1 E-c (multi-label emotion class.) English leaderboard example results. 33
4.3 Model performance on Plutchik Tweet Categories by F1 Score. 34

List of Figures

2.1 Plutchik's Wheel Of Emotions 5
2.2 Illustration of (a) LSTM and (b) gated recurrent units 8
2.3 Example of a CNN for text classification. 9
2.4 Learning process of transfer learning. 10
2.5 ULMFiT stages. 11

3.1 SemEval Task 1: Affect in Tweets label class balance 19
3.2 Illustration of BERT's tokenization 21
3.3 Text CNN Model Architecture. Note that this diagram utilizes the softmax function at the output layer due having only 2 classes to classify. This study makes use of the sigmoid function due to being a multi-label classification problem with 11 output classes. 24
3.4 Attention LSTM Model Architecture. 25
3.5 Illustration of the BERT encoder for single sentence classification. . . . 26
3.6 Application UI. 30

5.1 Training & Validation Loss for each of the defined models. 36
5.2 Training & Validation Accuracy (Jaccard) for each of the defined models. 37
5.3 Example of attention visualization for emotion classification. 38
5.4 Example of attention visualization on categories with insufficient training data. 39
5.5 Attention LSTM + BERT model on text with 102 words. 40
5.6 Text CNN + BERT model on text with 102 words. 40
5.7 Attention LSTM + BERT model on text with 138 words. 41
5.8 Text CNN + BERT model on text with 138 words. 41

Chapter 1

Introduction

1.1 Overview

Sentiment analysis is arguably one of the main practical and useful text classification problems, where the task is to classify a specific text as either "positive" or "negative". Evidently, if we can move from this binary classification task into classifying distinct emotions, this could lead to advancements in various applications and fields such as political science, human-computer interaction, artificial intelligence, psychology and more. For example, with regards to recommender systems, we can imagine how emotion detection can be used for providing recommendations based on the emotional state of a user. In marketing, the ability to identify emotions can produce a greater understanding towards the behavior and satisfaction of a business's customers. Psychologists could potentially study mental health problems and detect depressive symptoms in people based on the type of emotion that an algorithm could detect. This could even extend into analyzing the overall happiness of people on the planet as well as to other societal wellbeing metrics.

Although the usefulness of emotion detection from text is evident, the expression of emotion in language is very complex and context dependant. Additionally, unlike sentiment analysis, textual datasets which have been annotated with markers of emotional content are rare and in short supply. This is problematic since useful data is the key ingredient in allowing artificial intelligence models to unravel and understand the hidden information of a specific problem or task. While there have been various attempts to detect emotions in text using both supervised and unsupervised approaches (Seyeditabari et al. 2018), the use of deep learning remains a novel approach with room for improvement. Therefore, this paper investigates the feasibility of how deep learning can be used to effectively detect emotions from text.

1.2 Aims and objectives

The main aim of this project is to apply a deep learning model which can effectively detect emotions from text. A list of further objectives is shown below:

1. Explore current deep learning architectures specific to text classification.
2. Identify and make use of a reliable dataset.
3. Apply and modify existing architectures to the task of classifying emotion from text.
4. Explore and apply transfer learning learning techniques specific to NLP.
5. To report, evaluate and analyze attained results, which are to be subjected to a comprehensive discussion.
6. Implement a basic frontend interface where users can provide a model with text and view how the model detects specific emotions.

1.3 Summary of key findings

The report has showed that it's possible to effectively classify emotions from text using deep learning models. Two different model architectures were applied, with one being a CNN (Convolutional Neural Network) and the other an Attention-based bidirectional LSTM (Long Short Term Memory Network). The study made use of a transfer learning approach where the embedding layer of both models was initialized using word embeddings generated by the unsupervised learning algorithm, GloVe (Global Vectors for Word Representation), and contextualized word embeddings produced by a pre-trained BERT (Bidirectional Encoder Representations from Transformers) transformer model. The study showed that contextualized embeddings proved a significant advantage and resulted in an increase in model performance. The Attention LSTM model using extracted contextualized embeddings from BERT achieved a micro-averaged F1 score of 0.67 which was comparable to the top 10 post-evaluation results for the SemEval Task 1: E-c multi-label classification competition.

1.4 Chapter List

Chapter 2 Literature review. A review on the current literature surrounding deep learning and emotion classification.
Chapter 3 Methods & Experimental Setup. A detailed description on the methods and experiments used to carry out the experiment.

Chapter 4 Experimental Results. A comprehensive discussion on the results achieved using the chosen methods.

Chapter 5 Analysis & Discussion. An evaluation of the the attained results and an analysis of the chosen methods.

Chapter 6 Conclusion. Conclusions and discussion of future work.

Chapter 2

Literature Review

This chapter will review some of the general models of emotion to gain an understanding of the types of emotion that can be conveyed through text. The review also evaluates different deep learning models with relation to text classification, as well as discusses some of the recent trends in natural language processing with an emphasis on transfer learning along with word embedding and language modeling techniques. Furthermore, this chapter covers the availability of annotated datasets along with pre-trained word embeddings and language models. Lastly, this review assesses related work and any limitations in their approaches.

2.1 Background

2.1.1 Understanding Emotion

Before beginning to evaluate and discuss the detection of emotion, it is essential to gain an understanding of the general models of emotion as well as some of the principal theories in psychology. Thus, the first question we can ask ourselves is "What is emotion?". This is a difficult question to answer accurately due to the complexity of human behavior in the sense that emotion can be expressed in so many different ways. For example, emotion can be conveyed from facial expressions and gestures, speech, text and even from less obvious indicators such as heart rate, skin clamminess, temperature, and respiration velocity (Pantic et al. 2008). When it comes to classifying emotion from facial expressions, (Ekman 1992) showed that there is evidence for six emotions (happiness, surprise, fear, sadness, anger, and disgust). (Plutchik 1980) expanded on the emotions provided by Ekman as demonstrated by his illustration, "the wheel of emotions" (Figure 2.1). The illustration shows how different emotions can blend into one another creating new ones. If we can agree that emotion can be categorized into these distinct labels, it begs the question of whether it is possible to convey these

emotions through text?

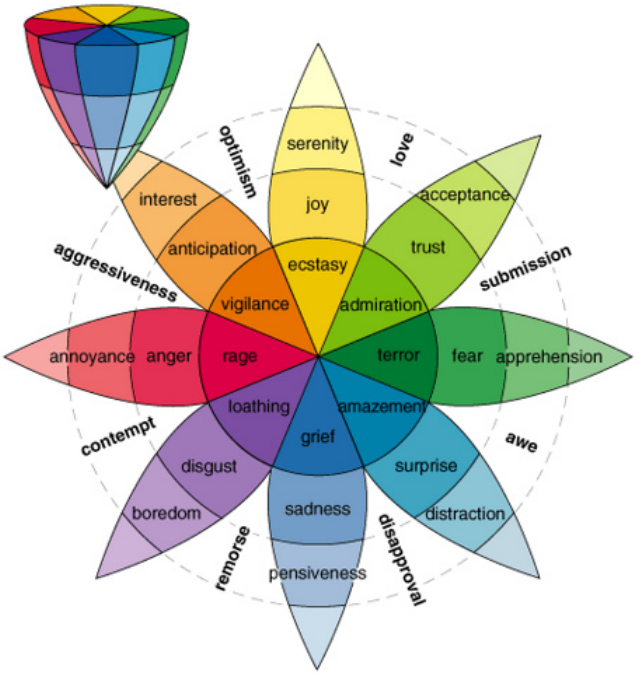


Figure 2.1: Plutchik's Wheel Of Emotions
(Plutchik 1980)

2.1.2 Affective Computing

The task of identifying emotion from text can be sectored into the field of "Affective Computing". Affective computing is the task of trying to assign computers human-like capabilities of observation, interpretation, and generation of emotional features (Tao & Tan 2005). The majority of research and applications with relation to affective computing has been focused on emotional speech processing, facial expressions, and body gestures and movement (Tao & Tan 2005). This is logical due to the innate ability of humans to be able to recognize and differentiate emotions from these features. However, when it comes to applying affective computing to text, researchers and practitioners have largely been fixated on machine learning-based approaches on sentiment analysis in which text is labeled based on a binary classification of either "positive" or "negative" (Seyeditabari et al. 2018). If we can effectively move from negative and positive sentiments into analyzing distinct emotions, this can lead to vast improvements in various fields such as political science, human-computer interaction, artificial intelligence, psychology and more (Seyeditabari et al. 2018).

2.1.3 Computational Techniques for Emotion Detection

2.1.3.1 Keyword-based

Early work concerned with emotion detection largely made use of keyword-based techniques. This was a relatively simple process as it involved searching and identifying specific words in text using pre-processing with an emotion dictionary and a parser. However, this approach is not the most effective as it relies heavily on task-specific keywords for sufficient results as well as a lot of pre-processing (Binali et al. 2010).

2.1.3.2 Machine Learning

Machine learning was deemed more suitable for the task of emotion detection due to the ability of quickly being able to learn new features from large training sets as well as being able to detect emotions that only have an indirect reference to the overall task (Binali et al. 2010) (Jain & Sandhu 2015). Specifically, the previous machine learning algorithms which were implemented in relation to affective computing and NLP included naive Bayes, random forests, decision trees, k-nearest neighbours, and support vector machines (Otter et al. 2018) (Jain & Sandhu 2015).

2.1.3.3 Machine Learning vs. Deep Learning

Machine learning has and continues to power various facets of modern society, however, has been limited in its ability to process data in a raw form, thus, requires a lot of domain expertise in order to transform data into a suitable internal representation (LeCun et al. 2015). **Representation learning** was introduced as a solution to this problem with the aim being to discover both the mapping from representation to output as well as the representation itself (Goodfellow et al. 2016). These learned representations allow AI systems to adjust to new domains without the need for human-intervention, nevertheless, they still face extreme difficulties in extracting high-level features from raw data (Goodfellow et al. 2016). Deep learning, on the other hand, has the ability to learn high-level intricate features from high dimensional data. Due to this, deep learning reduces the need for domain expertise and rigid feature extraction (LeCun et al. 2015). The next section evaluates some of the relevant deep learning models in relation to text classification.

2.2 Deep Learning

2.2.1 Sequence Models

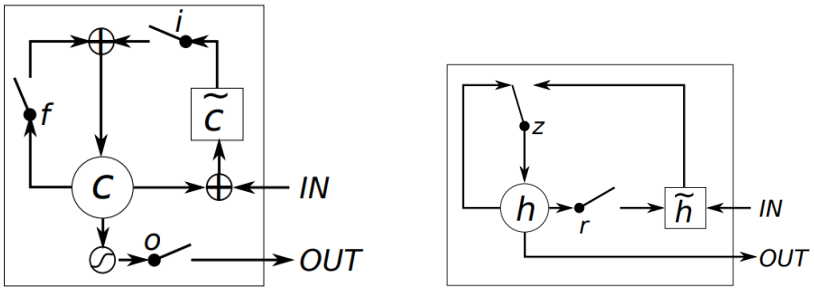
Sequence models have shown to be successful in supervised learning domains that consist of sequential data such as speech recognition, music generation, sentiment classification, DNA sequence analysis, machine translation and named entity recognition (Lipton 2015). It's important to consider sequence-based models for text specific tasks as opposed to standard neural networks mainly because standard networks rely heavily on the independence among training and test samples, meaning that the entire state of the network is erased after each sample is processed (Lipton 2015).

2.2.1.1 Recurrent Neural Network

Sequence-based models such as the recurrent neural network (RNN) has been used heavily in NLP and text classification related tasks. This is largely due to the structure and architecture of the network. Unlike standard networks, RNNs can recursively process arbitrary sequences of input by using their internal state allowing them to recognize patterns across time (Ghelani 2019). A limitation of standard RNNs is that they are unable to store information about past features for a long time (Graves 2013). This is also known as the **vanishing gradient problem** in which for multiple layer networks, the gradient can become too small for training to effectively reach a global minimum (Pascanu et al. 2012). There are multiple variations of RNNs which counter this issue of "amnesia". Most notably, the long short-term memory (LSTM) network and gated recurrent unit (GRU) have shown to perform well in many NLP tasks (Otter et al. 2018).

2.2.1.2 GRU & LSTM

The GRU and LSTM aim is to solve the vanishing gradient problem. The GRU differs from a standard RNN as it allows each recurrent unit to adaptively seize dependencies from different time scales (Chung et al. 2014). LSTMs and GRUs are designed in such a manner that allows them to retain important information for much longer while unnecessary information can be forgotten (Otter et al. 2018). These RNN variants have proved significant results in the field of word-level classification, sentiment classification and language generation (Young et al. 2017). Lample et al. (2016) showed compelling model performance for named entity recognition when using a bidirectional LSTM along with conditional random fields. Wang et al. (2015) demonstrated excellent results in predicting polarities of tweets using LSTM.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 2.2: Illustration of (a) LSTM and (b) gated recurrent units (Chung et al. 2014)

2.2.2 Convolutional Neural Network

Although sequence-based models such as the RNN have proven significant results in text classification and other NLP related tasks, Convolutional Neural Networks (CNNs) can be advantageous due to their ability in extracting position invariant features (Yin et al. 2017). CNNs have largely been applied to computer vision-related tasks, thus, it's important to note that techniques with regards to NLP and text classification differ to some extent (Yin et al. 2017). Furthermore, one of the major drawbacks of using sequence-based models is that they are blatantly slow to train. CNNs however, process elements simultaneously which in return speeds up the training process (Ghelani 2019). Within CNNs, the convolution results will signal when a unique pattern is detected. These patterns could be expressions such as "I love" or "That sounds fantastic" etc in which CNNs can identify such phrases within a sentence regardless of its position. Due to this, CNNs are suitable for classification tasks such as sentiment analysis and spam detection (Ghelani 2019). Concerning text classification, (Kim 2014) showed that a simple CNN with little hyperparameter tuning achieved excellent results when applied to sentence classification. Additionally, (Ruder et al. 2016) demonstrated convincing results when employing a CNN for aspect extraction and sentiment analysis. It's important to note that the results obtained by (Kim 2014) and (Ruder et al. 2016) were not attained by CNNs alone, however, benefitted from the use of transfer learning along with word embedding techniques.

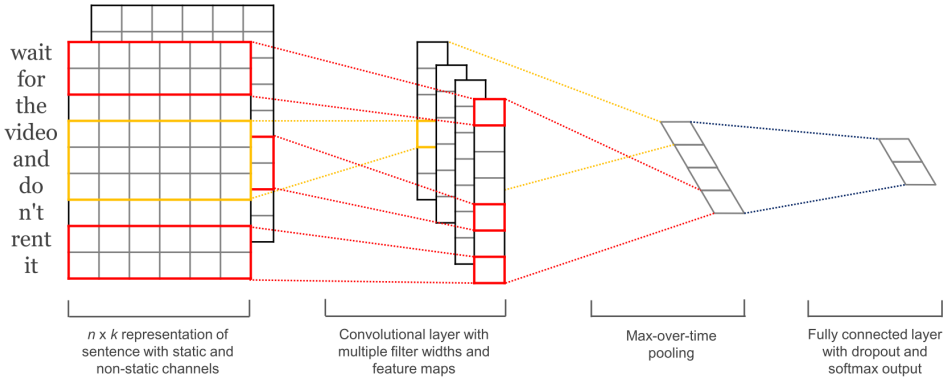


Figure 2.3: Example of a CNN for text classification.
(Kim 2014)

2.3 Transfer Learning

Collecting data is a complicated and expensive process which in return makes it difficult to build large-scale and high quality annotated datasets (Tan et al. 2018). This problem of insufficient training data applies to various domains such as emotion classification from text as there is a lack of available annotated data (Seyeditabari et al. 2018). When using a scarce dataset, it's evident that teaching a machine to learn the numerous grammatical nuances, cultural differences, sarcasm, and slang is an extremely difficult process. Transfer learning addresses this issue of insufficient training data by transferring knowledge from a different domain to the task at hand such that the pre-trained weights can be applied to the target domain (Tan et al. 2018). For example, we can imagine that learning to classify apples may help us in learning to recognize pears as we are leveraging the already existing knowledge learned from classifying apples to a related task (Pan & Yang 2010). Some of the key areas where transfer learning is being considered and applied are in self-driving cars and speech recognition (Ruder 2019). For example, companies such as Google and Udacity are applying transfer learning by allowing their models to learn from virtual environments in which the acquired knowledge is then transferred to the real world (Ruder 2019). With regards to text, however, transfer learning has mainly been applied in relation to language modeling and word embedding techniques (Otter et al. 2018).

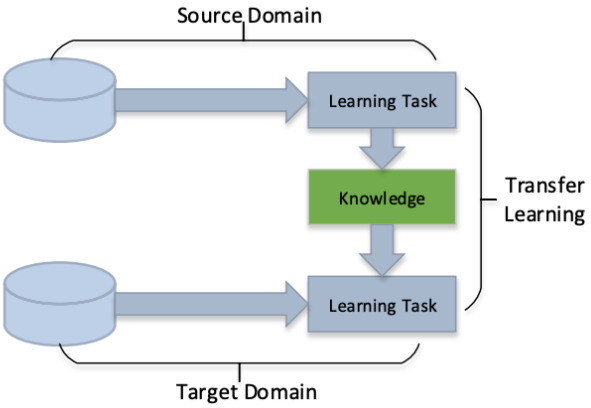


Figure 2.4: Learning process of transfer learning.
(Tan et al. 2018)

2.3.1 Word Embeddings

Simplistic models of transfer learning utilizing just a single layer of weights has been extremely popular for various years (Ruder & Eisenschlos 2019). Two of the most common techniques are Word2Vec and GloVe. Word2Vec, introduced by Google in 2013, is a word embedding model for computing continuous representations of words from extremely large datasets (Mikolov et al. 2013). While Word2Vec can learn embeddings by relating target words to their context, it fails at recognizing whether some context words appear together more often than others (Böhm 2018). Conversely, GloVe (Global Vectors for Word Representation), introduced by Stanford in 2014, creates a global co-occurrence matrix by estimating the probability that a given word will occur with other words (Pennington et al. 2014). Word embeddings are crucial to text classification as they turn text into a numerical format that deep neural networks are able to understand.

The above-mentioned word embedding techniques don't perform sufficiently on their own. Rather, these techniques have proven significant results when being able to inject knowledge from a larger corpus. For example, the already mentioned sentence level classifier using a CNN by (Kim 2014) trained their one layer convolution using Word2Vec in which vectors were trained on 100 billion words of Google News. Poria et al. (2016) also make use of the publicly available Word2Vec vectors trained on Google News as a means to classify sarcasm in tweets using deep CNNs.

2.3.2 Language Modeling

Though pre-trained word embeddings have been extremely influential, a major limitation is that they are trained on a neural network with one hidden layer, meaning that any previous knowledge can only be extracted from the first layer of the model (Ruder 2018). As quoted by Jeremy Howard and Sebastian Ruder from FastAI, *"only using transfer learning for a single layer is clearly just scratching the surface of what's possible"* (Ruder & Eisenschlos 2019). Language modeling is an NLP model that tries to predict the next word in a sentence, hence, is advantageous in comparison to static word embedding approaches such as Word2Vec and GloVe as it forces the model to use information from the entire sentence. Like with word embeddings, language models are typically trained with extremely large datasets, however, in an unsupervised manner which in return allows them to capture the syntactic features more deeply than word embeddings (Rao 2019). Most notably, (Howard & Ruder 2018) proposed Universal Language Model Fine-tuning (ULMFiT) for text classification. The ULMFiT model enables you to train the language model on an extremely large dataset in order to capture features of the language in different layers, as well as to fine-tune the language model on the target domain using discriminative fine-tuning and slanted learning rates to learn domain-specific features. The stages of the model are further portrayed in Figure 2.5. ULMFiT was shown to significantly outperform the state-of-the-art on six text classification tasks and showed that by using only 100 labeled examples, it matched the performance of training from scratch on 100x more data (Howard & Ruder 2018). Only shortly after the release of ULMFiT, similar models with different architectures were released, nevertheless, with pre-trained language models being at the core of the structure.

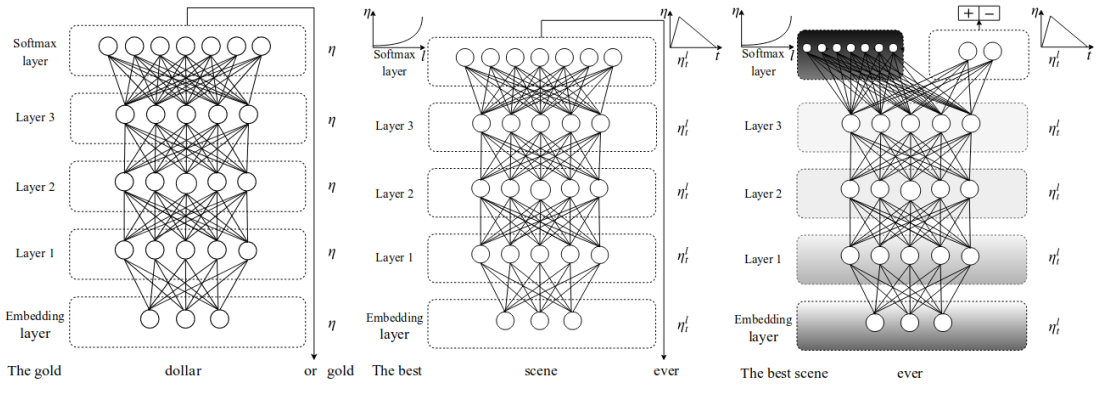


Figure 2.5: ULMFiT stages.
(Howard & Ruder 2018)

2.3.3 Attention Mechanisms & The Transformer

As discussed in section 1.2, RNNs have proven impressive results in various sequence modeling tasks, however, they face limitations due to their inability in storing information about past features for a long time as well as the inability to provide higher weight to more important words. A solution to this problem was proposed by (Vaswani et al. 2017) where they introduced the model of *self-attention*. The idea behind self-attention is to place "attention" towards words that are important to the meaning of the sentence. The representation of these instructive words is then formed into a sentence vector. Perhaps the most significant models where self-attention has been applied is in that of ELMo (Embeddings from Language Models) and transformer-based architectures such as BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al. 2018), Transformer-XL (Dai et al. 2019) and Open AI's GTP-2 (Radford et al. 2019).

With regards to text classification, (Reimers et al. 2019) applied both ELMo and BERT as a means to classify and cluster topic-dependent arguments and achieved compelling results. Their results indicate a captivating advantage in using contextualized word embeddings as opposed to previous word embedding approaches such as Word2Vec and GloVe.

2.3.3.1 ELMo

ELMo is designed such that is uses a deep bidirectional LSTM pre-trained on a large text corpus to learn context-dependent word representations (Peters et al. 2018). Furthermore, ELMo representations are purely character-based allowing it to handle out of vocabulary words. ELMo showed to significantly improve the state-of-the-art for multiple NLP related tasks.

Task	Previous SOTA	ELMo	Increase
SQuAD	SAN 84.4	85.8	1.4%
SNLI	(Chen et al. 2016) 88.6	88.7+/-0.17	0.1%
SRL	(He et al. 2017) 81.7	84.6	2.9%
Coref	(Lee et al. 2017) 67.2	70.4	3.2%
NER	(Peters et al. 2017) 91.93+/0.19	92.22+/-0.10	0.29%
SST-5	(McCann et al. 2017) 53.7	54.7+/-0.5	1.0%

Table 2.1: ELMo results vs. previous state-of-the-art.
(Peters et al. 2018)

2.3.3.2 BERT

BERT built on top of ELMo by pre-training deep bidirectional representations from unlabeled text using the transformer architecture to compute word embeddings (Devlin et al. 2018). Unlike ELMo, BERT uses masked language modeling and next sentence prediction for pre-training. BERT also proved to notably improve the state-of-the-art and showed better results in various tasks in comparison to ELMo.

Task	ELMo	BERT Large	BERT Base	Increase
NER	92.2	92.8	92.4	0.4%
SQuAD	85.8	93.2	91.8	6.7%
SST-2	90.4	94.9	93.5	3.8%
CoLA	36.0	60.5	52.1	20.3%
QNLI	79.8	92.7	90.5	11.8%

Table 2.2: BERT vs ELMo on NLP tasks.
(Devlin et al. 2018)

While it may seem that these attention based models are superior to previous approaches, (Ruder & Eisenschlos 2019) state that recurrent models such as ULMFiT still outperforms transformers on smaller datasets.

2.4 Datasets

2.4.1 Labeled Text

As was discussed in section 1.3, emotion detection from text faces the issue of having a lack of publicly available annotated data. In terms of labeled text, one of the most common resources is ISEAR, International Survey On Emotion Antecedents And Reactions. The ISEAR dataset consists of 3000 responses from people around the world in which they were asked to recall experiences which triggered at least one of the seven emotions depicted from (Ekman 1992) and (Plutchik 1980), i.e. anger, anticipation, joy, trust, fear, surprise, sadness, and disgust. This resulted in a dataset containing a total of 7600 instances of text conveying emotional states. Another more common dataset that has been used is SemEval-2018 Task 1 E-c. This resource consists of 6,857 tweets with binary labels for eight of the Plutchik classes as well as 3 more (optimism, pessimism, and love) (Mohammad et al. 2018). The SemEval-2007 Task 14 focused on affective text is another dataset which has been frequently used (Seyeditabari et al. 2018). It contains a total of 1250 annotated news headlines drawn from well-known newspapers such as the New York Times, BBC News, CNN as well as Google News.

Both the SemEval-2007 and 2018 tasks have mainly been used as benchmarks for comparing results achieved by different models. The lack of publicly available annotated data has largely pushed researchers and practitioners to mine text from social media sources such as Twitter where exposing emotion is possible through features such as hashtags and emoticons (Seyeditabari et al. 2018). Due to the lack of publicly available annotated data, it's crucial to consider incorporating the previously mentioned transfer learning approaches with regards to word embeddings and language modeling as this may allow deep neural networks to better "understand" the training text which in return could improve the overall performance in emotion detection.

2.4.2 Pre-trained Word Embeddings

Researchers behind the common word embedding models have open-sourced multiple pre-trained word embedding models for other researchers and practitioners to make use of. Table 2.3 displays some of these pre-trained word embeddings. The first row shows the properties of a Word2Vec model trained on a Google News corpus consisting of 100B words with 300-dimensional vectors for approximately 3M words and phrases (Mikolov et al. 2013). Additionally, we see that fastText distributes pre-trained vectors for both a Wikipedia and CommonCrawl corpus. The Wikipedia version consists of 16B tokens along with 300-dimensional vectors for 1M words, whilst the CommonCrawl version contains 600B tokens with 300-dimensional vectors for 2M words (Mikolov et al. 2018). Furthermore, we can see that GloVe issues pre-trained vectors for a Twitter, and Wikipedia combined with Gigaword corpus as well as two separate adaptations for a CommonCrawl corpus. The Twitter model is trained on 27B tokens with 300-dimensional vectors for 1.2M words. The Wikipedia and Gigaword combination is trained on 6B tokens with 300-dimensional vectors on 400,000 words. Lastly, the CommonCrawl versions are trained on both 42B tokens with 300-dimensional vectors for 1.9M words as well as 840 billion tokens with 300-dimensional vectors for 2.2M words (Pennington et al. 2014).

Corpus	Type	Size	Vocab size	Dim	Download
Google News	Word2Vec	100B	3M	300	link
Wikipedia	fastText	16B	1M	300	link
CommonCrawl	fastText	600B	2M	300	link
Twitter	GloVe	27B	1.2M	300	link
Wikpedia + Gigaword	GloVe	6B	400K	300	link
CommonCrawl	GloVe	42B	1.9M	300	link
CommonCrawl	GloVe	840B	2.2M	300	link

Table 2.3: Word embedding model properties.

2.4.3 Pre-trained Language Models

Like with pre-trained word embeddings, various organizations have open-sourced their pre-trained language models. Below we specifically view the properties of the pre-trained ELMo models released by (Peters et al. 2018) and pre-trained BERT models released by (Devlin et al. 2018). While there are other pre-trained language models to consider such as ones available with Transformer-XL (Dai et al. 2019) and Open-AI's GTP-2 (Radford et al. 2019), ELMo and BERT seem to be more applicable to text classification.

2.4.3.1 Pre-trained ELMo Models

Table 2.4 displays all pre-trained ELMo models. All models except for the Original 5.5B model were trained on the 1 Billion Word Language Model Benchmark which consists of approximately 800 million tokens of WMT 2011 News Crawl data. The ELMo 5.5B model was trained on a dataset containing 5.5B tokens of Wikipedia (1.9B) and all of the monolingual news crawl data from WMT 2008-2012 (3.6B) (Peters et al. 2018).

Model	Parameters	Weights
Small	13.6M	weights
Medium	28.0M	weights
Original	93.6	weights
Original (5.5B)	93.6	weights

Table 2.4: Pre-trained ELMo models.
(Peters et al. 2018)

Further information about ELMo along with downloads for the pre-trained weights can be found at `https://allennlp.org/elmo`.

2.4.3.2 Pre-trained BERT Models

Table 3.1 displays all pre-trained English BERT models. Both the BERT-Base and BERT-Large models were trained on Wikipedia data with 2.5B words from text passages of English Wikipedia as well as BookCorpus which is a dataset consisting of 11,038 unpublished books from 16 different genres (Devlin et al. 2018) (Zhu et al. 2015).

Model	Parameters	Weights
Large, Uncased (Whole Word Masking)	340M	weights
Large, Cased (Whole Word Masking)	340M	weights
Base, Uncased	110M	weights
Large, Uncased	340M	weights
Base, Cased	110M	weights
Large, Cased	340M	weights

Table 2.5: Pre-trained BERT models.
(Devlin et al. 2018)

Further information about BERT along with downloads for the pre-trained weights can be found at `https://github.com/google-research/bert`.

2.5 Related Work

A machine learning approach in the domain of emotion classification from text was done by (Calefato et al. 2017). They present a toolkit for emotion recognition from text where they specifically introduce an emotion lexicon designed to capture the presence of lexical cues that convey emotion in text. Using uni- and bi-grams with tf-idf weighting, they compute the tf-idf for each emotion category based on how many times they occur in WordNet affect, a lexical resource representing affective concepts in correlation with affective words. They train their classification models using Support Vector Machines (SVM) on a dataset containing 4,800 posts from Stack Overflow as well as 4000 commented posts by software developers on Jira. While having obtained sufficient results, they did not experiment with any deep learning approaches.

A deep learning approach was introduced by (Senarath & Thayasivam 2018). They describe a method to solve distinct emotion classification with the use of pre-trained word embedding models to train multiple neural networks along with an LSTM and CNN for feature extraction and feedforward neural network for classification. They specifically make use of Word2Vec, fastText and GloVe as their word-embedding algorithms. Their Word2Vec model was trained on both a Twitter corpus of 400 million tweets as well as the Google News corpus of 100 billion words. Interestingly, they found that their model performed best when using Word2Vec trained on Twitter data making the hypothesis that the model performed better due to the pre-trained Twitter corpus consisting of similar vocabulary to their domain. A limitation of their approach is that their research did not focus on fine-tuning the model architectures to different word-embeddings.

Furthermore, (Kratzwald et al. 2018) proposed "sent2affect", another transfer learning

approach in which they utilize a source domain that is semantically similar to the target domain. The related task they decided to use was sentiment analysis as it shares some similarities in the sense that classifying text as either "positive" or "negative" is achieved from lexical content. Their approach differs from (Senarath & Thayasivam 2018) as they decided to feed the pre-trained word-embeddings solely into a RNN as opposed to a combination of an LSTM and CNN. They compare their sent2affect approach to a pre-trained embeddings approach with a bidirectional LSTM layer and show that sent2affect produced some additional performance improvements.

The authors (Zhong & Miao 2019) presented a model to detect emotion from textual conversations. They compare the results obtained from using a Recurrent Convolutional Neural Network (RCNN) along with pre-trained word embeddings to those attained from using ELMo and BERT as their pre-trained language models. Interestingly, they found that the pre-trained language models performed worse in comparison to the RCNN. Their model was applied to classify emotions into just four different categories (happy, sad, angry and others), thus, results would likely be different if they were to attempt classifying into more labels.

Lastly, the research done by (Kant et al. 2018) demonstrated that multi-emotion sentiment can be classified using large pre-trained language models. Specifically, they train an attention-based transformer network on 50GB of Amazon reviews. In addition, they compare results obtained using an attention-based transformer to that achieved by a multiplicative LSTM (mLSTM) and show that the transformer generally out-performs the mLSTM.

2.6 Conclusions

Upon reviewing the multiple variations of deep learning models and recent trends in NLP, it would seem that there are various paths one can take to tackle the task of emotion detection from text. The review showed that both sequence-based models and convolutional neural networks have demonstrated sufficient results in the domain of text classification and noted that while RNN variants such as the GRU and LSTM are able to retain crucial information for much longer, CNNs can be superior due to being able to extract position invariant features. Due to this, it would be suitable to compare and contrast results obtained from both models. Furthermore, the findings in the research on transfer learning showed that the collection of data with regards to emotion classification from text is a complicated and expensive process, however, this problem is relaxed when leveraging existing knowledge learned from a different domain to the task at hand. Specifically, the review showed that the use of pre-trained

word embedding and pre-trained language models have proved to significantly improve the state-of-the-art in multiple NLP related tasks. In addition, multiple researchers and organizations have open sourced their variations for pre-trained word embedding and pre-trained language models which can be fine-tuned on small NLP tasks such as emotion detection. Therefore, these techniques would present a feasible approach for the area of research.

Chapter 3

Methods & Experimental Setup

This chapter details the methods and experimental setup used for carrying out the task of classifying emotion from text. The chapter outlines the choice of dataset, the use of pre-trained embeddings as well as pre-processing and tokenization. In addition, the chosen baseline machine learning algorithms, and deep learning models, the chosen hyperparameters and training setup, and performance evaluation is specified. Lastly, the chapter describes the technologies used for carrying out the experiment.

3.1 Dataset

The experiment is carried out using the *SemEval Task 1: Affect in Tweets* emotion classification dataset in which given a tweet, the task is to classify the text as having no emotion or as one, or more, emotions for eight of the Plutchik 2.1 categories plus Optimism, Pessimism and Love (Mohammad et al. 2018). The dataset consists of 6838 training examples, 886 validation examples and 3259 testing examples.

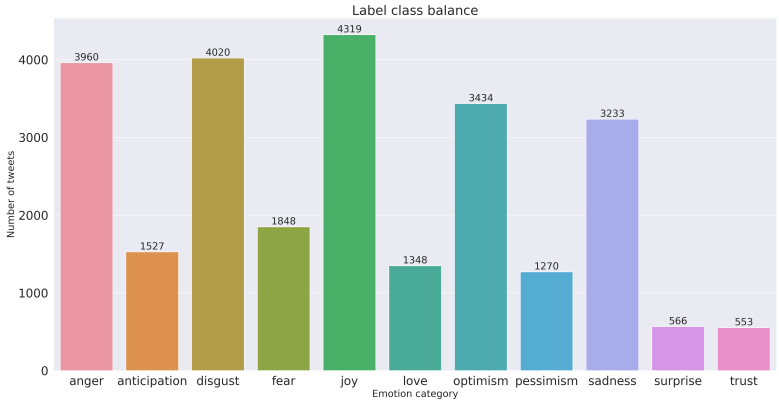


Figure 3.1: SemEval Task 1: Affect in Tweets label class balance

3.2 Pre-trained Embeddings

Due to the limited amount of training data, the selected models are initialized using the publicly available 300 dimensional GloVe word embeddings which have been trained on Wikipedia and Gigaword[1]. Additionally, the experiment makes use of the HuggingFace Transformers library as a means to obtain the pre-trained BERT base-uncased transformers model which contains 110M parameters[2]. The model is frozen[3] such that only the learned representations produced by the transformer are used as the embedding layer for the chosen neural network model architectures. These vectors have a dimensionality of 768. By using BERT as an encoder, the aim is that the powerful architecture behind the transformer model will assist in achieving a better understanding of emotions in text.

3.3 Pre-processing & Tokenization

The pre-processing steps taken differ depending on whether the chosen models are initialized with the pre-trained GloVe embeddings or BERT encodings. The differences in the pre-processing approaches are detailed below.

3.3.1 GloVe

Tweets may include URLs, mentions, hashtags, emojis, smileys and other specific types of text which likely do not benefit the performance of the chosen model architectures. Therefore, pre-processing is applied such that HTML tags and non-word characters are removed, emoticons are appended to the end of the tweet and all tweets are converted to lowercase. The table below shows an example of an original tweet and the resulting preprocessed tweet.

Original	I came to #work for no reason *emoji* I could've stayed in bed
Processed	i came to work for no reason I could ve stayed in bed *emoji*

Table 3.1: Tweet pre-processing example.

[1] https://nlp.stanford.edu/projects/glove/

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

[3] To not train the transformer itself, only train the section of the model which learns from the representations generated by the transformer.

3.3.2 BERT

BERT itself does not necessarily benefit from standard pre-processing as the BERT tokenizer is able to do the majority of this for you. When using a BERT pre-trained model, it is essential that the text conforms to the vocabulary that was initially used when the model was trained. The BERT tokenizer uses a WordPiece model such that the input sentence is broken down into multiple sub-words. All sub-words except the first begin with "##". This simply indicates to BERT that the first word of a token can be dismissed with the token for the entire word. Using the HuggingFace Transformers library, the BERT tokenizer is applied to the input tweets such that the input text is convered into tokens, and pre-processing is applied such that the tokens are converted to their indexes. It is essential to also provide the indexes marking the beginning of a sentence token *"[CLS]"*, end of sentence token *"[SEP]"*, padding token *"[PAD]"* and unknown token *"[UNK]"* as these are also required by the BERT model. Lastly, the number of tokens is set to a maximum length of *512 - 2* as the BERT model was trained on a maximum of 512 tokens. Note that two is subtracted from the maximum length because the BERT model requires two tokens to be appended to both the beginning and end of a sequence. This is further represented by figure 3.2.

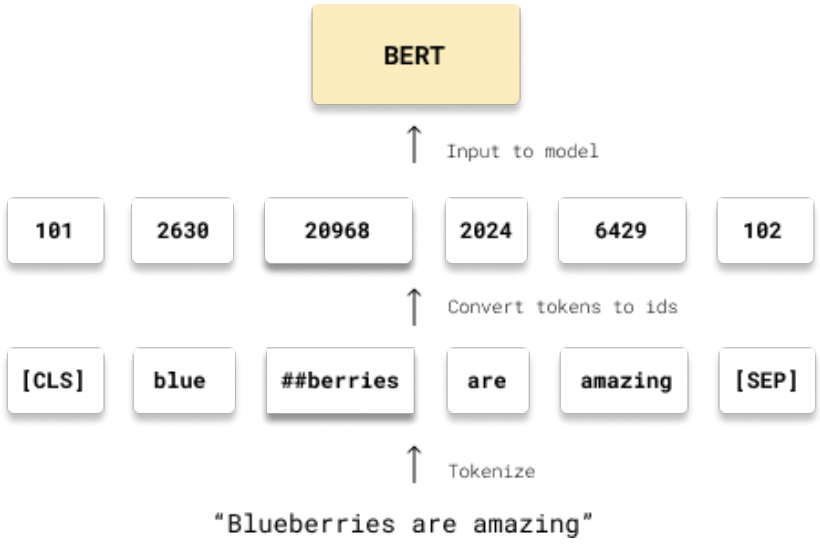


Figure 3.2: Illustration of BERT's tokenization

3.4 Baselines

As a means to gain an understanding on whether the emotion classification task is doable, and if there is a possibility for obtaining a better performance using deep learning methods, Logistic Regression and a Support Vector Machine (SVM) using

TF-IDF (Term Frequency Inverse Document Frequency) vectorization is first made use of. The vocabulary is set to consider the top 20,000 features ordered by term frequency across the corpus. Moreover, all characters are converted to lowercase before tokenizing, features are made of word n-grams, common english stop words are removed, and an n-gram range is implemented where the lower boundary is 1 and the upper boundary is 4. In addition, the same text pre-processing method discussed in section 3.3.1 is applied.

3.4.1 Logistic Regression

The first baseline approach makes use of Logistic Regression using L_2 regularization, and stochastic average gradient descent. By default Logistic Regression cannot handle target vectors with more than two classes, thus, training is done using One-Vs-Rest classification such that a separate classifier is trained for each emotion class. This splits the multi-label classification task into one binary classification problem per class.

3.4.2 SVM

The second baseline approach makes use of Linear Support Vector classification with squared hinge loss, and L_2-norm regularization which reduces the variance of estimated coefficients, and has shown to achieve better prediction accuracy (Martinez 2017). Similar to Logistic Regression, training is done using One-Vs-Rest classification such that for each each classifier, the class is fitted against all other classes.

Squared hinge loss is defined as follows:

$$L(y, \hat{y}) = \sum_{i=0}^{N} (\max(0, 1 - y_i \cdot \hat{y_i})^2) \tag{3.1}$$

3.5 Models

The experiment is carried out using both a CNN and Attention-based bidirectional LSTM model. Both models are assesed by comparing the performance obtained when using the standard word vectors produced by GloVe and contextualized embeddings produced by the pre-trained BERT model. In particular, the experiment makes use of the CNN architecture proposed by (Kim 2014) and the Attention LSTM architecture proposed by the authors (Zhou et al. 2016).

3.5.1 CNN

As described by the original paper, the model takes in as input an n x k representation of a sentence with static and non-static channels. The representation is then fed into a convolutional layer which consists of multiple filter widths and feature maps. The convolutional layer is then passed into a max-over-time pooling layer and finally a fully connected layer with dropout. The network architecture specific to our chosen dataset is specified below.

- Embedding layer
  - When using the GloVe embeddings, the embedding layer is initiated as 15828 x 300 where 15828 is the number of unique tokens and 300 is the embedding dimension.
  - When using BERT, the embedding layer is initiated as 30522 x 768 where 30522 is the vocabulary size and 768 is the hidden size of the BERT model. The embedding layer also contains position embeddings of 512 x 768 and token type embeddings of 2 x 768. Each embedding vector is transformed each time it iterates through one of the BERT Encoder layers (12 in total). After the final encoding layer, only the first encoder block is returned as this has shown to be sufficient for classification tasks. This is further represented in Figure 3.5.
- Hidden layers
  1. Convolutional layer containing 100 1 x 1 filters, with stride 1 followed by a rectified linear unit.
  2. Convolutional layer containing 100 1 x 1 filters, with stride 1 followed by a rectified linear unit.
  3. Convolutional layer containing 100 1 x 1 filters, with stride 1 followed by a rectified linear unit.
  4. Max-over-time pooling layer
- Output layer
  - Fully-connected layer with 300 input features *(length of the filter sizes* $\times$ *number of filters))* with dropout, and sigmoid output due to being a multi-label classification task (11 output features).

The sigmoid function is defined as follows:

$$h_\theta(x) = \frac{1}{1 + e^{-}\theta^T x} \tag{3.2}$$

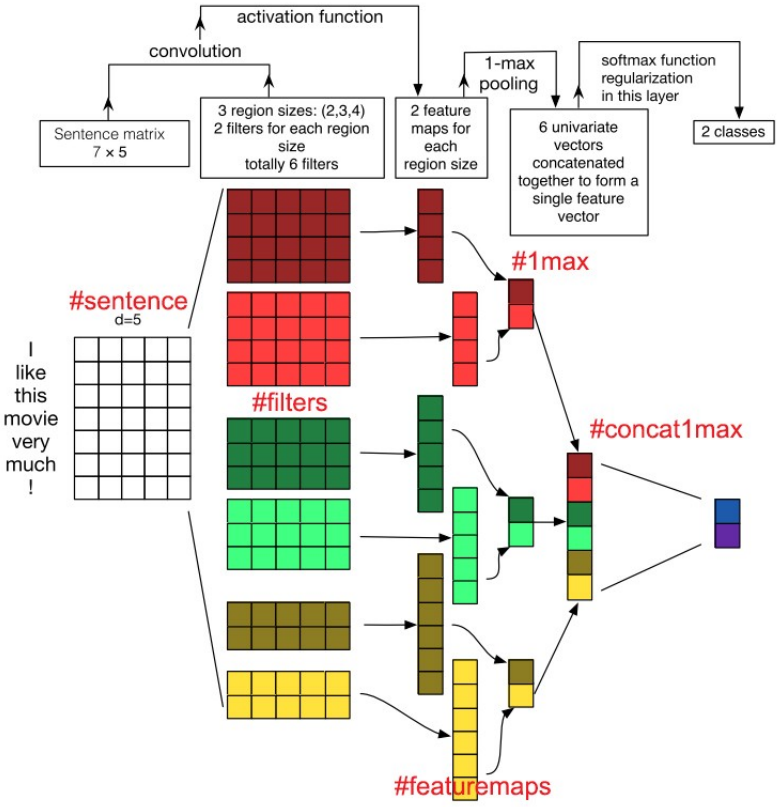


Figure 3.3: Text CNN Model Architecture. Note that this diagram utilizes the softmax function at the output layer due having only 2 classes to classify. This study makes use of the sigmoid function due to being a multi-label classification problem with 11 output classes.

(Kim 2014)

3.5.2 Attention LSTM

Likewise to the CNN, this model takes in as input the pre-processed tweet and passes it into an embedding layer which projects each word into a continuous vector space. The embedding layer is passed into the bidirectional LSTM layer in order to extract the high level features obtained from the embedding layer. Moreover, these extracted high level features are passed into an attention layer which produces a weight vector for each word capturing the most crucial semantic information in the given tweet. Note that the architecture proposed by the authors, shown in figure 3.4, utilizes a softmax layer to produce just a single prediction whereas for this experiment, the sigmoid function is made use of as each prediction can have a non-exclusive output. The network architecture specific to our chosen dataset is specified below.

- Embedding layer
  - Same as in section 3.4.1 with an additional dropout layer.
- Hidden layers
  - 2 layer bidirectional LSTM with dropout.
  - Contains 256 hidden units and 300 input features when using GloVe embeddings and 768 input features when using the BERT encodings.
  - Attention layer with 256 input features (hidden units) and 1 output feature.
- Output layer
  - Fully connected layer with 256 input features (hidden units), with dropout, and sigmoid output due to being a multi-label classification task (11 output features).

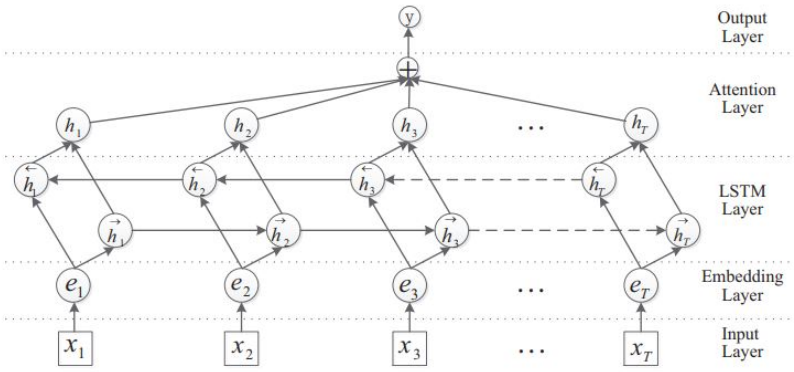


Figure 3.4: Attention LSTM Model Architecture.
(Zhou et al. 2016)

The attention mechanism is calculated as follows:

Let H be a matrix containing output vectors $[h_1, h_2, ..., h_T]$ which the LSTM layer generates, where T is the length of the tweet. The representation r is produced by a weighted sum of these output vectors:

$$M = \tanh(H) \quad \sigma = softmax(w^T M) \quad r = H\sigma^T \tag{3.3}$$

(Zhou et al. 2016)

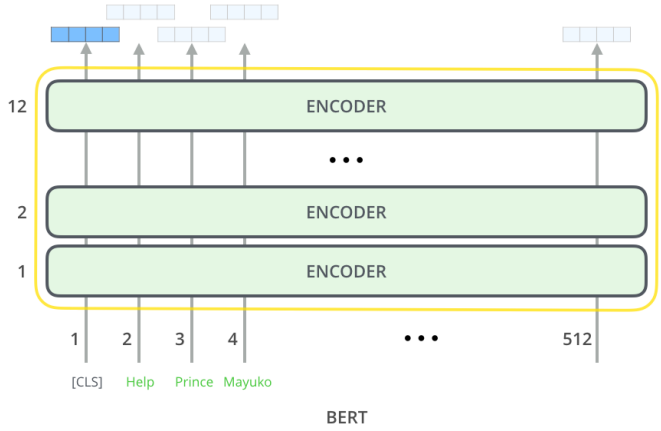


Figure 3.5: Illustration of the BERT encoder for single sentence classification. (Alammar 2020)

3.6 Hyperparameters and Training

With regards to the CNN, the architecture is set to make use of rectified linear units, $3 \times (1 \times$embedding dimension) filters with 100 feature maps for each filter window and a dropout rate of 0.5. Moreover, whilst the original paper uses a minibatch size of 50 along with stochastic gradient descent using the Adadelta update rule, a minibatch size of 10 is used for this experiment as it proved better model performance. Training is done using stochastic gradient descent along with the Adam algorithm. The hyperparameters are displayed in table 3.2.

With regards to the Attention LSTM, the hyperparameter setup described in the original paper (Zhou et al. 2016) is utilized. Thus, the architecture consists of an embedding layer, LSTM layer and fully-connected layer. The dropout is set to 0.3, 0.3, 0.5 respectively. Likewise to the Text CNN, a minibatch size of 10 is used as this also proved better model performance in comparison to higher batch sizes. Training is also done using stochastic gradient descent using the Adam algorithm with an L_2 penalty of 1×10^{-5}. These hyperparameters are displayed in table 3.3.

Hyperparameter	Value	Description
Minibatch size	10	Number of training examples used to calculate the stochastic gradient descent update
Embedding dim	300 when using GloVe and 768 when using BERT	Total number of features
Max vocabulary size	20000 (only applies when using GloVe)	The maximum size of the vocabulary
Number of filters	100	Number of feature maps per filter window
Filter windows	1,1,1	1×3 matrix of weights
Dropout	0.5	Randomly omits feature detectors at the penultimate layer
Optimizer	Adam	Method for stochastic optimization
Learning rate	1×10^{-3}	Learning rate for Adam algorithm
Loss	Sigmoid Binary Cross Entropy	Sigmoid activation plus cross entropy loss
Epochs	10	Number of training iterations

Table 3.2: Hyperparameters for Text CNN model using GloVe and BERT embeddings.

Hyperparameter	Value	Description
Minibatch size	10	Number of training examples used to calculate the stochastic gradient descent update
Embedding dim	300 when using GloVe and 768 when using BERT	Total number of features
Hidden size	256	Size of the hidden state of an LSTM unit
Number of layers	2	Number of LSTM layers
Embedding layer dropout	0.3	Randomly omits feature detectors at the embedding layer
LSTM layer dropout	0.3	Randomly omits feature detectors at the LSTM layer
Fully connected layer dropout	0.5	Randomly omits feature detectors at the penultimate layer
Optimizer	Adam	Method for stochastic optimization
Learning rate	1×10^{-3}	Learning rate for Adam algorithm
Weight decay	1×10^{-5}	L2 regularization for Adam algorithm
Loss	Sigmoid Binary Cross Entropy	Sigmoid activation plus cross entropy loss
Epochs	10	Number of training iterations

Table 3.3: Hyperparameters for Attention LSTM model using GloVe and BERT embeddings.

3.7 Performance Evaluation

The official competition metric used for this specific dataset was the Jaccard index. As this is a multi-label classification task, each tweet can have one or more gold emotion labels and one or more predicted emotion labels. Thus, the accuracy is calculated as the size of the intersection of the predicted and correct labels divided by the size of their union, as shown by the equation below (Mohammad et al. 2018).

$$Accuracy = \frac{1}{|T|}\sum_{t \in T}(\frac{G_t \cap P_t}{G_t \cup P_t}) \tag{3.4}$$

Note: G_t is the set of gold labels for tweet t and P_t is the set of predicted labels for tweet t.

Whilst the Jaccard index was deemed the official competition metric, recent papers have evaluated the performance of their models using the micro-averaged F1-score and macro-averaged F1-score. The macro-averaged method calculates the metrics for each label and calculates the unweighted mean. It does not take into account label imbalance. The micro-averaged method calculates the metric globally by calculating the total true positives, false negatives and false positives. The micro-averaged and macro-averaged F1-scores are suitable for imbalanced multi-label classification tasks. We can clearly see from figure 3.1 that our dataset is somewhat imbalanced as specific emotion categories such as surprise and trust contain significantly less tweets in comparison to the others. Therefore, these metrics provide a good comparison for the overall performance of the models implemented in this study.

A detailed instruction on how the micro-averaged and macro-averaged F1 scores are calculated is shown below:

- A true positive (TP) is an outcome where the model correctly predicts the positive class.
- A false positive (FP) is an outcome where the model incorrectly predicts the positive class.
- A true negative (TN) is an outcome where the model correctly predicts the negative class.
- A false negative (FN) is an outcome where the model incorrectly predicts the negative class.

The precision measure and macro and micro averaging for class j is defined as follows:

$$P_j = \frac{TP_j}{TP_j + FP_j} \quad \mathrm{P}_{\mathrm{micro}} = \frac{\sum_{j=1}^{M} TP_j}{\sum_{j=1}^{M}(TP_j + FP_j)} \quad \mathrm{P}_{\mathrm{macro}} = \frac{\sum_{j=1}^{M} P_j}{M} \tag{3.5}$$

The recall is respectively defined as follows:

$$R_j = \frac{TP_j}{TP_j + FN_j} \quad \mathrm{R}_{\mathrm{micro}} = \frac{\sum_{j=1}^{M} TP_j}{\sum_{j=1}^{M}(TP_j + FN_j)} \quad \mathrm{R}_{\mathrm{macro}} = \frac{\sum_{j=1}^{M} R_j}{M} \tag{3.6}$$

And the F1 score, micro-averaged F1 and macro-averaged F1 is defined as such:

$$F1_j = 2 \cdot \frac{P_j \cdot R_j}{P_j + R_j} \quad \mathrm{F1}_{\mathrm{micro}} = 2 \cdot \frac{\mathrm{P}_{\mathrm{micro}} \cdot \mathrm{R}_{\mathrm{micro}}}{\mathrm{P}_{\mathrm{micro}} + \mathrm{R}_{\mathrm{micro}}} \quad \mathrm{F1}_{\mathrm{macro}} = 2 \cdot \frac{\mathrm{P}_{\mathrm{macro}} \cdot \mathrm{R}_{\mathrm{macro}}}{\mathrm{P}_{\mathrm{macro}} + \mathrm{R}_{\mathrm{macro}}} \tag{3.7}$$

3.8 Technologies

The experiment makes use of PyTorch for constructing the architecture of the models. As was mentioned above, the HuggingFace Transformers library is made use of in order to apply and retrieve the embeddings from a pre-trained BERT model. In addition, the 300 dimensional GloVe vectors are retrieved from the *torchtext* library provided by PyTorch. The experiment also make use of the Scikit Learn[4] library in order to assess model performance using the Jaccard index, and micro-averaged and macro-averaged F1 scores, as well as for the implementation of the baseline machine learning algorithms. Model training is done with Google Colaboratory notebooks using an NVIDIA Tesla K80 GPU.

3.9 Applications

Inspired by the DeepMoji[5] website from MIT, a similar application has been implemented where you can view how the Attention LSTM model captures the most important semantic information from your given input text. As shown by figure 3.6, the words which the model deems the most significant are enlarged and highlighted in different shades of blue (dark being the most significant and light being the least). Moreover, the application provides you with the top 3 categories that the model predicted which

[4] https://scikit-learn.org/stable/
[5] https://deepmoji.mit.edu/

you can mark as either correct or incorrect. User corrections are appended to a separate dataset with the aim being that the additional data provided by users over a certain period of time would increase model performance and improve the label class imbalance. The application is available at https://ainoa.netlify.app.

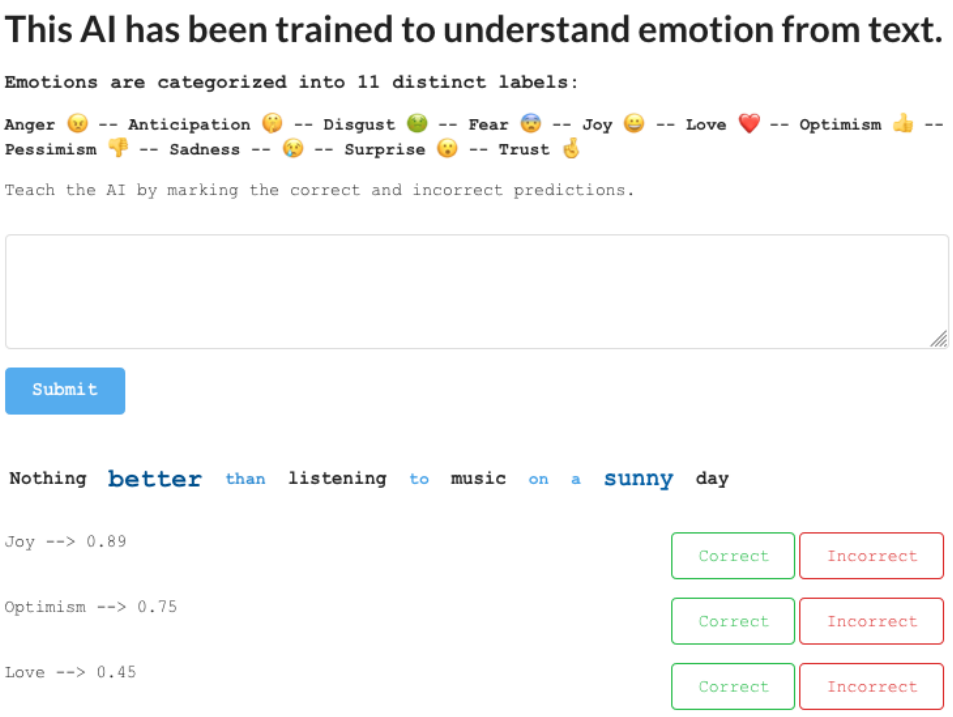


Figure 3.6: Application UI.

Chapter 4

Experimental Results

This chapter presents the results achieved using the methods and setup described in Chapter 3. A detailed review and comparison is provided on the results attained using the baseline methods and deep learning models. Model performance is compared with the current state of the art results on the experiment dataset, and results are assessed on the eight core emotions illustrated by the psychologist Robert Plutchik.

4.1 Results

The performance of the models are compared in Table 4.1.

	Accuracy (Jaccard)	**Micro F1**	**Macro F1**
Attention LSTM + GloVe	0.469	0.601	0.453
Attention LSTM + BERT	**0.544**	**0.668**	0.491
Text CNN + GloVe	0.407	0.547	0.387
Text CNN + BERT	0.527	0.659	0.463
SVM + TF-IDF	0.308	0.576	**0.533**
LR + TF-IDF	0.226	0.497	0.422

Table 4.1: Performance comparison of the baseline algorithms, and the Attention LSTM and Text CNN using GloVe and BERT embeddings.

4.2 Baselines

Our first observation is how the SVM outperforms the Logistic Regression model and proves better results in both the micro-averaged F1 score and macro-averaged F1 score in comparison to the Text CNN using GloVe embeddings. Moreover, the SVM produced

a better macro-averaged F1 score in comparison to all other models. Since the macro-averaged F1 weighs the performance on all classes equally, this suggests that the SVM does a better job taking into account challenging and rare emotion categories (Kant et al. 2018). On the other hand, the Jaccard index results for both Logistic Regression and the SVM are considerably less in comparison to the deep learning models. This tells us that the baseline models have more difficulties in distinguishing the overlap between predicted emotion categories and the true emotion categories.

4.3 Attention LSTM vs Text CNN

The results indicate that when both the Attention LSTM and Text CNN utilize the contextualized embeddings produced by the pre-trained BERT model, the results are significantly better. We can see that the Attention LSTM using BERT encodings outperforms all other models. This suggests that the ability for an LSTM to be able to "remember" previous information can prove advantageous with respect to emotion detection. It's important to note that the results attained by both the Attention LSTM and Text CNN using BERT embeddings are comparable, therefore, the results could likely differ if the results were recorded as the average of more training iterations. Moreover, with regards to the Text CNN, the approximately 10% difference for each performance metric between the GloVe and BERT embedding models, strongly indicates that the contextualized embeddings produced by a pre-trained BERT model are superior to the pre-trained context insensitive embeddings produced by GloVe.

4.4 Leaderboard Comparison

As shown by Table 4.2, the best model (Attention LSTM + BERT) performance is compared with the top 10 post-evaluation period results for the official SemEval: Task 1 E-c (multi-label emotion class.) competition. We can see that the Attention LSTM + BERT model achieves comparable results in terms of the micro-averaged F1 score and is only slightly less in terms of the other metrics. This suggests that the model does an adequate job when calculating the total true positives, false negatives and false positives, thus taking into account the label imbalance as shown in Figure 3.1.

[1]The competition results can be viewed here: https://competitions.codalab.org/competitions/17751#results.

#	Team Name or User	Accuracy (Jaccard)	Micro F1	Macro F1
1	zimkjh	0.593	0.704	0.565
2	psyML	0.574	0.697	0.574
3	prabod	0.574	0.689	0.500
4	yyyura	0.567	0.672	0.520
5	Amobee	0.566	0.673	0.490
6	Zupun7	0.565	0.680	0.545
7	YNU-HPCC	0.558	0.674	0.488
8	ventakesh-1729	0.558	0.677	0.476
9	Seekers7	0.555	0.667	0.505
10	ELiRF-UPV	0.552	0.658	0.512
	Attention LSTM + BERT	0.544	0.668	0.491

Table 4.2: SemEval 2018: Task 1 E-c (multi-label emotion class.) English leaderboard top 10 post-evaluation results[1].

4.5 Results on Plutchik categories

Similar to the work done by the authors (Kant et al. 2018), table 4.3 compares model performance by F1 Score for each Plutchik (Figure 2.1) category. Interestingly, we can see that all models have a difficult time generalizing to categories with just a few training examples, whereas categories with a sufficient amount of training data perform well. This strongly suggests that categories such as anticipation, surprise and trust, which have a worse class imbalance would benefit from having a larger training dataset. When comparing these results to those attained by the authors (Kant et al. 2018), we can see that both the Attention LSTM and Text CNN using the embeddings produced by BERT achieves comparable results on categories with ample training data, however, performs significantly worse on categories such as anticipation, surprise and trust which contain less training data. It's important to note that they train using a fine-tuned transformer language model on a large 40GB text dataset, and then transfer the model to the emotion classification dataset, whereas this experiment only uses the learned representations (contextualized embeddings) produced by a pre-trained BERT transformer model within the embedding layer of the chosen model architectures. Therefore, it is no surprise that they achieve better results on rare and difficult emotion categories since transformer networks have shown to be superior to other network architectures in various NLP related tasks.

Furthermore, a number of the top SemEval Task 1: E-c participants, such as the winners (Baziotis et al. 2018) use additional training data. They collected a big dataset

of 550 million English tweets and used it for calculating word statistics and training word2vec. Having only trained on 6838 training examples and achieving comparable results, indicates that contextualized embeddings produced by BERT can be effective when classifying emotion from text. Another interesting observation is how the Attention LSTM + GloVe model performed better in comparison to the Attention LSTM + BERT model on categories with less training data. Furthermore, we see that that on average the baseline SVM model significantly outperforms the Text CNN using GloVe embeddings and performs slightly better than the Attention LSTM using GloVe embeddings on a few categories such as disgust, joy and sadness. This suggests that unsupervised learning algorithms such as GloVe for producing vector representations for words isn't necessarily a superior approach.

Model	Anger	Anticipation	Disgust	Fear	Joy	Sadness	Surprise	Trust	**Avg**
Attention LSTM + GloVe	0.65	**0.25**	0.59	0.64	0.74	0.56	**0.14**	**0.16**	0.47
Attention LSTM + BERT	**0.76**	0.19	0.71	0.68	**0.82**	**0.65**	0.11	0.07	**0.50**
Text CNN + GloVe	0.63	0.00	0.58	0.55	0.72	0.46	0.11	0.00	0.38
Text CNN + BERT	**0.76**	0.03	**0.72**	**0.70**	**0.82**	0.62	0.00	0.05	0.46
SVM + TF-IDF	0.64	0.15	0.60	0.65	0.75	0.57	0.12	0.02	0.44
LR + TF-IDF	0.60	0.01	0.56	0.44	0.70	0.43	0.06	0.00	0.35

Table 4.3: Model performance on Plutchik Tweet Categories by F1 Score.

4.6 Summary

Overall, the models using attention performed the best as they outperformed both the Text CNN models and baseline models. This was expected due to the nature of the attention mechanism in being able to capture the most crucial semantic information, thus, identifying which words in the input text are the most significant. Therefore, the results indicate that attention is a promising technique for capturing information from text with regards to emotion detection. Additionally, it was expected that the model with embeddings produced by the pre-trained BERT model would outperform the same model initialized with pre-trained GloVe word embeddings. This is due to the relatively new transformer architecture that BERT uses for computing word embeddings, which enables it to take into account word ordering which GloVe does not.

Chapter 5

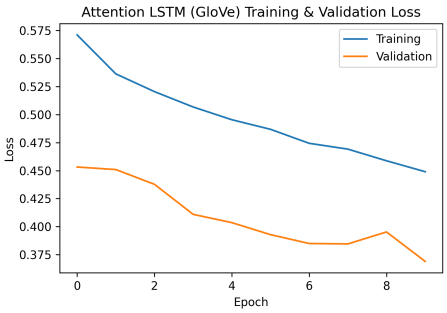
Analysis & Discussion

This chapter provides a detailed evaluation and discussion on the methods used to carry out the experiment. In particular, the training and validation loss & accuracy is analyzed, the attention mechanism of the Attention LSTM model using BERT embeddings is assessed on a variety of different text, and the Attention LSTM and Text CNN using BERT embeddings is evaluated on longer pieces of text.

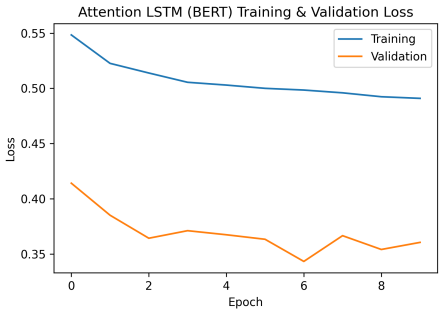
5.1 Training and Validation Loss / Accuracy

In Figure 5.1 the training and validation loss is compared for each of the selected neural network models, and 5.2 compares the training and validation jaccard accuracy. We can evidently see that with the chosen hyperparameters, the Attention LSTM is able to generalize much better. With regards to both Text CNN models, we can see that the model quickly begins to over-fit resulting in an increase in generalization error. While both the Text CNN using GloVe embeddings and BERT encodings results in over-fitting, an interesting observation is that the model using GloVe embeddings appears to over-fit at a much faster rate. Attempts were made to reduce the amount of over-fitting through hyperparameter tuning, however, no significant improvements were observed. Moreover, the Attention LSTM plots indicate that we could possibly train for longer than 10 epochs as we observe a validation loss that is less than the training loss and a validation accuracy that is greater than the training accuracy. Thus, it is possible that training for longer could improve the performance of the model. A likely reason for the validation loss to be less than the training loss and validation accuracy to be greater than training accuracy is the use of dropout at multiple layers. This is because dropout was activated during training, however, deactivated when evaluating on the validation set. Other likely causes include the training set potentially containing "harder" cases

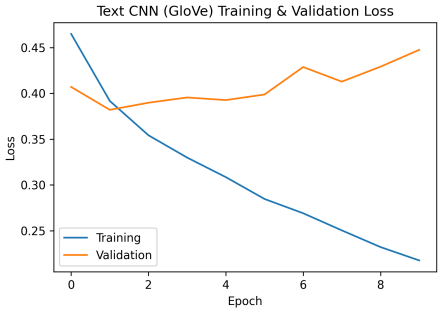
to learn, or the validation set consisting of "easier" cases to predict[1].



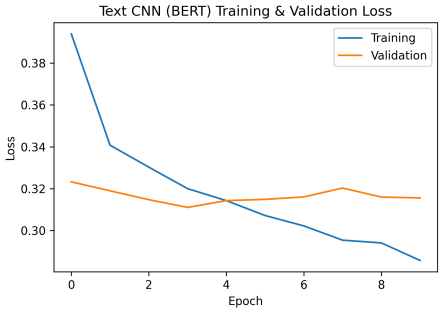
(a) Attention LSTM (GloVe) Training & Validation Loss



(b) Attention LSTM (BERT) Training & Validation Loss



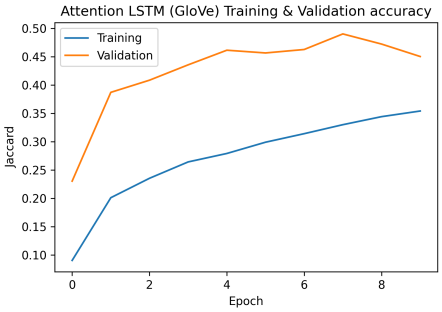
(c) Text CNN (GloVe) Training & Validation Loss



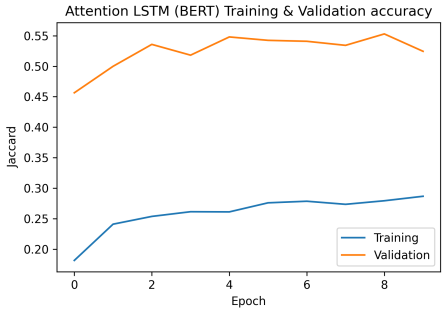
(d) Text CNN (BERT) Training & Validation Loss

Figure 5.1: Training & Validation Loss for each of the defined models.

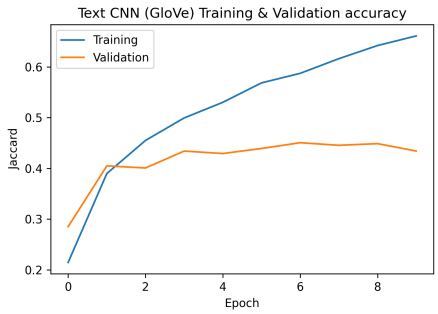
[1] https://stats.stackexchange.com/questions/187335/validation-error-less-than-training-error



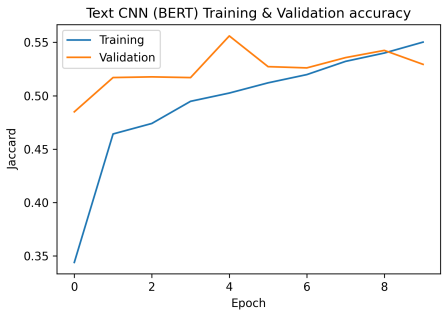
(a) Attention LSTM (GloVe) Training & Validation Accuracy



(b) Attention LSTM (BERT) Training & Validation Accuracy



(c) Text CNN (GloVe) Training & Validation Accuracy



(d) Text CNN (BERT) Training & Validation Accuracy

Figure 5.2: Training & Validation Accuracy (Jaccard) for each of the defined models.

5.2 Attention Visualization

Figure 5.3 displays the words that the attention model using BERT embeddings considers the most important with regards to its predictions. The color intensity and word size corresponds to the weight given to each word. Thus, the darker the shade of blue and the larger the font size, the more important it is to the final prediction. From the examples provided, it seems as if the attention model is successfully able to "place attention" towards words which have the most emotional meaning. For example, with the text: *"This lockdown got me feeling a bit scared"*, the model correctly correlates the two most significant words (feeling and scared) with sadness and pessimism. Similarly, with the text: *"This song is just sooo good"*, the model clearly depicts the words (song and good) with the positive emotions: joy, optimism and love.

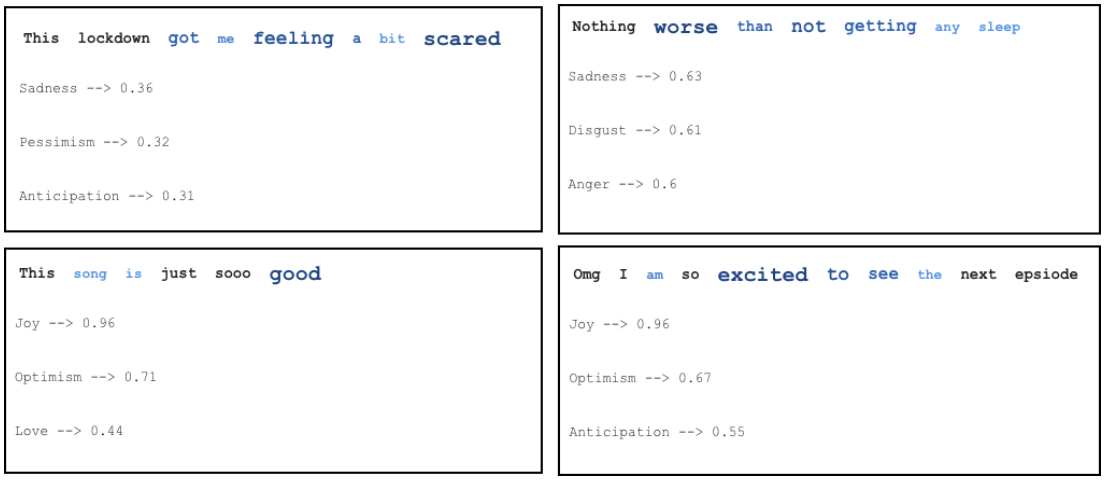


Figure 5.3: Example of attention visualization for emotion classification.

As was observed in table 4.3, the Attention LSTM model had difficulties in generalizing to categories with just a few training examples, such as surprise, trust and anticipation. As a means to observe how the model detects emotions for these categories, we provided the model with the following texts which we believe convey these emotions:

1. I wonder what life will be like next year (anticipation)
2. I was not expecting that to happen (surprise)
3. I feel comfortable around them (trust)

As can be seen in Figure 5.4, with regards to the first case, we see that the model is correctly able to detect the emotion as anticipation, and considers the words "like", "next", and "year" the most important. With regards to the two other cases, it's difficult to disagree with the models predictions. We see that for the second case,

the model detects the emotions as fear, anticipation and pessimism, and considers the word "expecting" the most significant. While not expecting something to happen could potentially be categorized as pessimism, "surprise" could be deemed a more accurate classification. Similarly, with the third case we see that the model classifies the text as optimism whereas "trust" could be considered a more accurate prediction. We can clearly see from each visualization that the model mainly classifies the provided texts as categories which have considerably more training data, such as joy, optimism and sadness. Thus, the assumption once again is that these predictions could be more accurate were the imbalanced classes to contain more training instances.

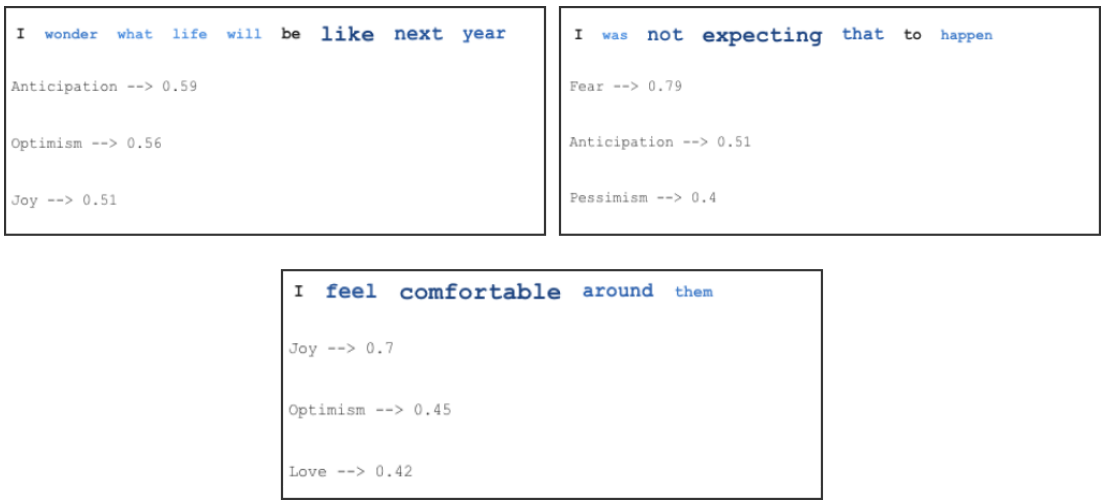


Figure 5.4: Example of attention visualization on categories with insufficient training data.

5.3 Model Performance on Longer Text

We briefly discussed in Chapter 2 how RNN models such as an LSTM are trained to recognize patterns across time, whilst CNN's detect patterns across space. With that in mind, the expectation is for the Attention LSTM model to better classify emotions from longer texts. In order to test this hypothesis, both the Attention LSTM and Text CNN were evaluated on a few Reddit posts from subreddits which clearly depict emotional content. Considering that the average length of a tweet in the SemEval dataset is approximately 95 words, this sub-experiment was tested on text with at least 100 words.

Attention LSTM

```
I'm only 24, and have worked part time jobs since I graduated from college.
With everything going on, I've been furloughed from both of my part time jobs.
Well one offered me full time, and I was able to negotiate the salary up a full dollar per hour!
And it has benefits and everything! I know it's not a big deal but I'm just so excited and
can't really tell anyone, especially as I haven't spoken to my other part time job yet.
Just hoping everything works out well, and that I have money for grad school in the fall!

Text length: 102 words

ANGER: 0.18
ANTICIPATION: 0.40
DISGUST: 0.24
FEAR: 0.42
JOY: 0.39
LOVE: 0.12
OPTIMISM: 0.55
PESSIMISM: 0.37
SADNESS: 0.41
SURPRISE: 0.20
TRUST: 0.24
```

Figure 5.5: Attention LSTM + BERT model on text with 102 words.

Text CNN

```
I'm only 24, and have worked part time jobs since I graduated from college.
With everything going on, I've been furloughed from both of my part time jobs.
Well one offered me full time, and I was able to negotiate the salary up a full dollar per hour!
And it has benefits and everything! I know it's not a big deal but I'm just so excited and
can't really tell anyone, especially as I haven't spoken to my other part time job yet.
Just hoping everything works out well, and that I have money for grad school in the fall!

Text length: 102 words

ANGER: 0.01
ANTICIPATION: 0.31
DISGUST: 0.01
FEAR: 0.11
JOY: 0.58
LOVE: 0.01
OPTIMISM: 0.93
PESSIMISM: 0.03
SADNESS: 0.05
SURPRISE: 0.01
TRUST: 0.01
```

Figure 5.6: Text CNN + BERT model on text with 102 words.

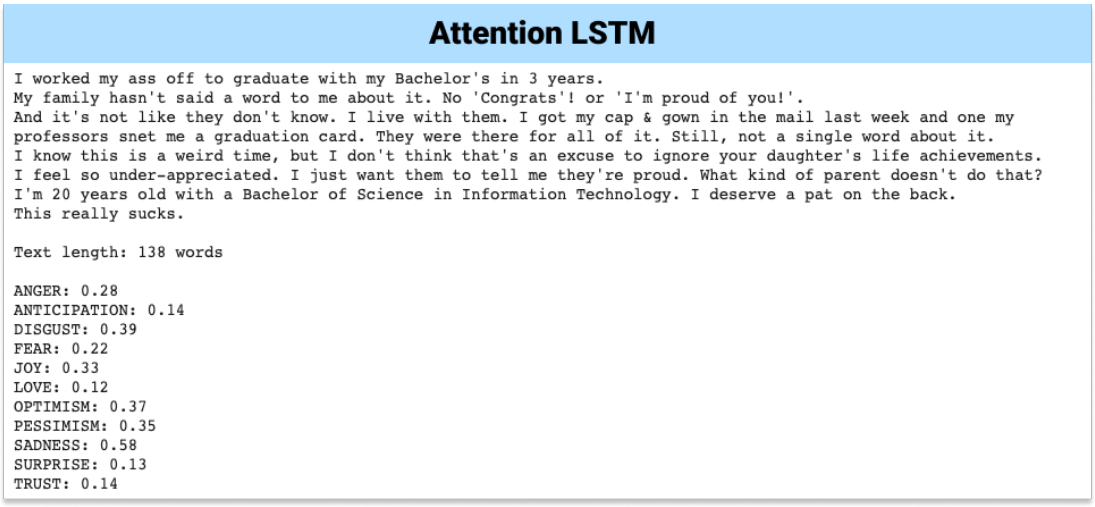


Figure 5.7: Attention LSTM + BERT model on text with 138 words.

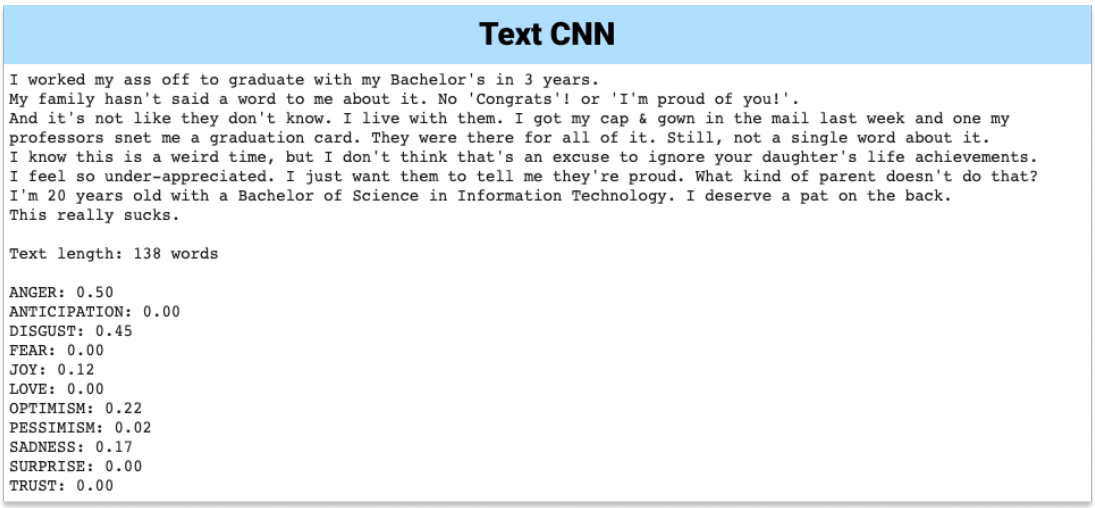


Figure 5.8: Text CNN + BERT model on text with 138 words.

By comparing the predictions for both the Attention LSTM and Text CNN models with BERT embeddings on longer pieces of text, a better understanding was gained in how the models were classifying the emotions. Interestingly, we found that the Attention LSTM provides more weight towards each of the emotion categories, whilst the Text CNN focuses the majority of its predictions towards just a few categories. For example, we can see from Figure 5.5 and Figure 5.6 that both models consider the text to mainly convey optimism, however, the Attention LSTM provides a sigmoid value of 0.55 whilst the Text CNN gives a value of 0.93. Moreover, the emotion that the Attention LSTM considers the second most significant was fear, whereas the Text CNN considered joy as the second most important emotion. Similarly, with regards to Figure 5.7 and Figure 5.8, we see that the Attention LSTM outputs values for each of the emotion categories, whilst the Text CNN doesn't output values for anticipation, love, surprise and trust. This indicates that the Attention LSTM is better able to depict multiple emotions out of longer pieces of text in comparison to the Text CNN. This correlates with the results in Table 4.3 where we observed that while both models have a difficult time generalizing to categories with an insufficient amount of training data, the Attention LSTM outputs a slightly higher F1 score for these categories, and therefore the Attention LSTM is better able to handle the label imbalance.

Chapter 6

Conclusion

This chapter summarizes the main outcomes and conclusions resulting from this study, and provides a detailed discussion of future work.

6.1 Conclusions

This report examined the ability of different deep learning models in detecting emotions from text. Through the experiments we observed that in general the Text CNN and Attention LSTM model prove better results in comparison to the baseline Logistic Regression and SVM machine learning algorithms. Due to the limited amount of training data, a transfer learning approach was made use of such that the deep learning models were injected with pre-trained word embeddings which have already captured the semantic meanings of words. In particular, the study made use of pre-trained GloVe embeddings and the learned representations from a frozen pre-trained BERT language model. The aim of this approach was to compare the use of the contextualized embeddings produced by BERT to the traditional context insensitive word vectors produced by GloVe, and the impact these have on model performance. We discovered that the models which made use of the contextualized embeddings produced by the pre-trained BERT transformers model outperformed the same models using GloVe word embeddings. This suggested that the ability for pre-trained language models such as BERT to be able to capture the context of a word proves a significant advantage with regards to classifying emotion from text.

When comparing both the Text CNN and Attention LSTM to emotion categories with an insufficient amount of training data as well as against longer pieces of text, we discovered that the Attention LSTM is better able to depict multiple emotions in comparison to the Text CNN. This indicated that the Attention LSTM does a better job in considering emotion categories with an insufficient amount of training data. When

visualizing the attention mechanism of the Attention LSTM model, we observed that for the majority of instances, the model is successfully able to capture the most important words which correlate with the model predictions.

The project compared the performance of the best model (Attention LSTM + BERT) to the current top 10 post-evaluation results for the SemEval Task 1: E-c (multi-label emotion classification) competition results, and showed that competitive results were achieved in terms of the micro-averaged F1 score. This was a satisfactory result as it shows that the model does a positive job in handling the label imbalance associated with the used dataset. Overall, this project has shown that deep learning along with transfer learning techniques can be used effectively for detecting emotions from text.

6.2 Future Work

Having developed a front-end interface where you can provide the Attention LSTM + BERT model with your own input text and mark the models predictions as correct or incorrect, the aim is that the performance of the model could be improved. Ideally, we would like to provide the model with more instances on categories with less training data, such as anticipation, surprise and trust, as we noticed that these were the categories that performed worse. Evidently, there are some challenges associated with this, for example: how do we gain enough traction on the application such that more instances could be provided to the model? Moreover, if the model already has trouble detecting emotions with less training data, it won't output these categories as predictions which can be marked as correct or incorrect as often as one would like to the user of the application. Thus, modifications need to be made to the application such that we find an accurate way of providing the model with more instances on these categories with an insufficient amount of training data.

Furthermore, while model training was done using the learned representations produced by the BERT Base, Uncased model, which is a 12-layer pre-trained model with 768 hidden states, 12 attention heads and 110M parameters, it would be interesting to carry out the same experiment using the larger variation of the transformer[1] as this would likely result in better model performance. Nevertheless, it's important to note that the use of the larger pre-trained model is extremely computationally demanding, therefore, there would be a trade-off between better model performance and training time. The Attention LSTM + BERT model is already approximately 500MB in size, thus, implementing a larger variation into a production-ready application would be

[1]BERT Large, Uncased: https://huggingface.co/transformers/pretrained_models.html

very costly.

Finally, the source code with regards to the experiments that have been carried out are available on github[2], such that the results are easily reproducible and can be used for further experimentation in the field.

[2] https://github.com/oaarnikoivu/ainoa

Bibliography

Alammar, J. (2020), 'The illustrated bert, elmo, and co. (how nlp cracked transfer learning)'.
URL: *https://jalammar.github.io/illustrated-bert/*

Baziotis, C., Athanasiou, N., Chronopoulou, A., Kolovou, A., Paraskevopoulos, G., Ellinas, N., Narayanan, S. & Potamianos, A. (2018), 'Ntua-slp at semeval-2018 task 1. predicting affective content in tweets with deep attentive rnns and transfer learning'.

Binali, H., wu, C. & Potdar, V. (2010), Computational approaches for emotion detection in text, pp. 172 – 177.

Böhm, T. (2018), 'The general ideas of word embeddings'.
URL: *https://towardsdatascience.com/the-three-main-branches-of-word-embeddings-7b90fa36dfb9*

Calefato, F., Lanubile, F. & Novielli, N. (2017), 'Emotxt: A toolkit for emotion recognition from text', *CoRR* **abs/1708.03892**.
URL: *http://arxiv.org/abs/1708.03892*

Chen, Q., Zhu, X., Ling, Z., Wei, S. & Jiang, H. (2016), 'Enhancing and combining sequential and tree LSTM for natural language inference', *CoRR* **abs/1609.06038**.
URL: *http://arxiv.org/abs/1609.06038*

Chung, J., Gülçehre, Ç., Cho, K. & Bengio, Y. (2014), 'Empirical evaluation of gated recurrent neural networks on sequence modeling', *CoRR* **abs/1412.3555**.
URL: *http://arxiv.org/abs/1412.3555*

Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V. & Salakhutdinov, R. (2019), 'Transformer-xl: Attentive language models beyond a fixed-length context', *CoRR* **abs/1901.02860**.
URL: *http://arxiv.org/abs/1901.02860*

Devlin, J., Chang, M., Lee, K. & Toutanova, K. (2018), 'BERT: pre-training of deep bidirectional transformers for language understanding', *CoRR* **abs/1810.04805**.
URL: *http://arxiv.org/abs/1810.04805*

Ekman, P. (1992), 'Are there basic emotions?'.
URL: *https://pdfs.semanticscholar.org/c2f4/41a578c1f2a5d147d3ac378454839a6cb217.pdf*

Ghelani, S. (2019), 'Medium'.
URL: *https://towardsdatascience.com/text-classification-rnns-or-cnn-s-98c86a0dd361*

Goodfellow, I., Bengio, Y. & Courville, A. (2016), *Deep Learning*, MIT Press. http://www.deeplearningbook.org.

Graves, A. (2013), 'Generating sequences with recurrent neural networks', *CoRR* **abs/1308.0850**.
URL: *http://arxiv.org/abs/1308.0850*

He, L., Lee, K., Lewis, M. & Zettlemoyer, L. (2017), Deep semantic role labeling: What works and what's next, *in* 'Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Association for Computational Linguistics, Vancouver, Canada, pp. 473–483.
URL: *https://www.aclweb.org/anthology/P17-1044*

Howard, J. & Ruder, S. (2018), 'Fine-tuned language models for text classification', *CoRR* **abs/1801.06146**.
URL: *http://arxiv.org/abs/1801.06146*

Jain, U. & Sandhu, A. (2015), A review on the emotion detection from text using machine learning techniques.

Kant, N., Puri, R., Yakovenko, N. & Catanzaro, B. (2018), 'Practical text classification with large pre-trained language models', *CoRR* **abs/1812.01207**.
URL: *http://arxiv.org/abs/1812.01207*

Kim, Y. (2014), Convolutional neural networks for sentence classification, *in* 'Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)', Association for Computational Linguistics, Doha, Qatar, pp. 1746–1751.
URL: *https://www.aclweb.org/anthology/D14-1181*

Kratzwald, B., Ilic, S., Kraus, M., Feuerriegel, S. & Prendinger, H. (2018), 'Decision support with text-based emotion recognition: Deep learning for affective computing', *CoRR* **abs/1803.06397**.
URL: *http://arxiv.org/abs/1803.06397*

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. & Dyer, C. (2016), 'Neural architectures for named entity recognition', *CoRR* **abs/1603.01360**.
URL: *http://arxiv.org/abs/1603.01360*

LeCun, Y., Bengio, Y. & Hinton, G. (2015), 'Deep learning', *Nature* **521**, 436 EP –.
URL: *https://doi.org/10.1038/nature14539*

Lee, K., He, L., Lewis, M. & Zettlemoyer, L. (2017), 'End-to-end neural coreference resolution', *CoRR* **abs/1707.07045**.
URL: *http://arxiv.org/abs/1707.07045*

Lipton, Z. C. (2015), 'A critical review of recurrent neural networks for sequence learning', *CoRR* **abs/1506.00019**.
URL: *http://arxiv.org/abs/1506.00019*

Martinez, D. L. (2017), 'Regularization approaches for support vector machines with applications to biomedical data', *CoRR* **abs/1710.10600**.
URL: *http://arxiv.org/abs/1710.10600*

McCann, B., Bradbury, J., Xiong, C. & Socher, R. (2017), 'Learned in translation: Contextualized word vectors', *CoRR* **abs/1708.00107**.
URL: *http://arxiv.org/abs/1708.00107*

Mikolov, T., Chen, K., Corrado, G. S. & Dean, J. (2013), 'Efficient estimation of word representations in vector space'.
URL: *http://arxiv.org/abs/1301.3781*

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C. & Joulin, A. (2018), Advances in pre-training distributed word representations, *in* 'Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)'.

Mohammad, S., Bravo-Marquez, F., Salameh, M. & Kiritchenko, S. (2018), SemEval-2018 task 1: Affect in tweets, *in* 'Proceedings of The 12th International Workshop on Semantic Evaluation', Association for Computational Linguistics, New Orleans, Louisiana, pp. 1–17.
URL: *https://www.aclweb.org/anthology/S18-1001*

Otter, D. W., Medina, J. R. & Kalita, J. K. (2018), 'A survey of the usages of deep learning in natural language processing', *CoRR* **abs/1807.10854**.
URL: *http://arxiv.org/abs/1807.10854*

Pan, S. J. & Yang, Q. (2010), 'A survey on transfer learning', *IEEE Transactions on Knowledge and Data Engineering* **22**(10), 1345–1359.

Pantic, M., Nijholt, A., Pentland, A. & Huanag, T. (2008), 'Human-centred intelligent human-computer interaction (hci2): How far are we from attaining it?', *International Journal of Autonomous and Adaptive Communications Systems* **1**, 168–187.

Pascanu, R., Mikolov, T. & Bengio, Y. (2012), 'Understanding the exploding gradient problem', *CoRR* **abs/1211.5063**.
URL: *http://arxiv.org/abs/1211.5063*

Pennington, J., Socher, R. & Manning, C. D. (2014), Glove: Global vectors for word representation, *in* 'In EMNLP'.

Peters, M. E., Ammar, W., Bhagavatula, C. & Power, R. (2017), 'Semi-supervised sequence tagging with bidirectional language models', *CoRR* **abs/1705.00108**.
URL: *http://arxiv.org/abs/1705.00108*

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. & Zettlemoyer, L. (2018), Deep contextualized word representations, *in* 'Proc. of NAACL'.

Plutchik, R. (1980), 'A general psychoevolutionary theory of emotion', *Theories of emotion* **1**, 3–31.

Poria, S., Cambria, E., Hazarika, D. & Vij, P. (2016), 'A deeper look into sarcastic tweets using deep convolutional neural networks', *CoRR* **abs/1610.08815**.
URL: *http://arxiv.org/abs/1610.08815*

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. (2019), 'Language models are unsupervised multitask learners'.
URL: *https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf*

Rao, P. (2019), 'Transfer learning in nlp for tweet stance classification'.
URL: *https://towardsdatascience.com/transfer-learning-in-nlp-for-tweet-stance-classification-8ab014da8dde*

Reimers, N., Schiller, B., Beck, T., Daxenberger, J., Stab, C. & Gurevych, I. (2019), 'Classification and clustering of arguments with contextualized word embeddings', *CoRR* **abs/1906.09821**.
URL: *http://arxiv.org/abs/1906.09821*

Ruder, S. (2018), 'Nlp's imagenet moment has arrived'.
URL: *https://thegradient.pub/nlp-imagenet/*

Ruder, S. (2019), 'Transfer learning machine - learning's next frontier'.
URL: *https://ruder.io/transfer-learning/index.html*

Ruder, S. & Eisenschlos, J. (2019), 'Efficient multi-lingual language model fine-tuning'.
URL: *http://nlp.fast.ai/*

Ruder, S., Ghaffari, P. & Breslin, J. G. (2016), 'Insight-1 at semeval-2016 task 5: Deep learning for multilingual aspect-based sentiment analysis'.

Senarath, Y. & Thayasivam, U. (2018), DataSEARCH at IEST 2018: Multiple word embedding based models for implicit emotion classification of tweets with deep learning, *in* 'Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis', Association for Computational Linguistics, Brussels, Belgium, pp. 211–216.

Seyeditabari, A., Tabari, N. & Zadrozny, W. (2018), 'Emotion detection in text: a review', *CoRR* **abs/1806.00674**.
URL: *http://arxiv.org/abs/1806.00674*

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C. & Liu, C. (2018), 'A survey on deep transfer learning', *CoRR* **abs/1808.01974**.
URL: *http://arxiv.org/abs/1808.01974*

Tao, J. & Tan, T. (2005), Affective computing: A review, pp. 981–995.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017), 'Attention is all you need'.

Wang, X., Liu, Y., Sun, C., Wang, B. & Wang, X. (2015), Predicting polarities of tweets by composing word embeddings with long short-term memory, *in* 'Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)', Association for Computational Linguistics, Beijing, China, pp. 1343–1353.

Yin, W., Kann, K., Yu, M. & Schütze, H. (2017), 'Comparative study of CNN and RNN for natural language processing', *CoRR* **abs/1702.01923**.
URL: *http://arxiv.org/abs/1702.01923*

Young, T., Hazarika, D., Poria, S. & Cambria, E. (2017), 'Recent trends in deep learning based natural language processing', *CoRR* **abs/1708.02709**.
URL: *http://arxiv.org/abs/1708.02709*

Zhong, P. & Miao, C. (2019), ntuer at SemEval-2019 task 3: Emotion classification with word and sentence representations in RCNN, *in* 'Proceedings of the 13th International Workshop on Semantic Evaluation', Association for Computational Linguistics, Minneapolis, Minnesota, USA, pp. 282–286.
URL: *https://www.aclweb.org/anthology/S19-2048*

Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H. & Xu, B. (2016), Attention-based bidirectional long short-term memory networks for relation classification, *in* 'ACL'.

Zhu, Y., Kiros, R., Zemel, R. S., Salakhutdinov, R., Urtasun, R., Torralba, A. & Fidler, S. (2015), 'Aligning books and movies: Towards story-like visual explanations by watching movies and reading books', *CoRR* **abs/1506.06724**.
URL: *http://arxiv.org/abs/1506.06724*

Chapter 7

Appendices

7.1 Appendix A - Project Log

#	Date	Activities
1	20/09/2019	Discuss possible project ideas.
2	27/09/2019	Decide the topic of project and prepare the project proposal.
3	30/09/2019	Submit the project proposal for review.
3	01/10/2019	Begin the literature review.
4	04/10/2019	Implement background research.
5	15/10/2019	Research on neural network architectures and pre-trained models.
6	20/10/2019	Research on transformers and word embedding techniques.
7	08/11/2019	Submit literature review.
8	12/11/2019	Setup Google Colaboratory notebooks. Experiment with the researched methods.
9	20/11/2019	Begin methods and experiment design.
10	27/11/2019	Submit initial draft on methods and experiment design.
11	06/12/2019	Begin the implementation of experimental methods. Play around with different datasets, namely the ISEAR and SemEval Task 1: E-c datasets.
12	18/12/2019	Christmas break.
13	04/01/2020	Continue with implementation of experimental methods. Initial commit to Github repository.
14	14/01/2020	Decide on SemEval dataset. Test using TF-IDF and BOW with Logistic Regression.
15	16/01/2020	Begin testing with the BERT transformer model from the HuggingFace transformers library.
16	24/01/2020	Successfully implement CNN model using BERT and GloVe embeddings. Begin implementing Flask server. Setup simple client to interact with the Flask server.
17	28/01/2020	Continue with research. Make modifications to methods and experimental setup.
18	07/02/2020	Discussion of implementation throughout Christmas break with supervisor. Discussion on initial model results and steps for further research.
19	14/02/2020	Implemented bidirectional Attention LSTM model with BERT and GloVe embeddings.
20	16/02/2020	Replace Flask CNN model with Attention LSTM due to better results.
21	19/02/2020	Modify client side code to visualize the weights of the Attention LSTM model.
22	28/02/2020	Show results and application to supervisor. Discussion on how to begin writing the experimental results and analysis section of report.
23	04/03/2020	Run deep learning and baseline models. Record the results.
24	12/03/2020	Modifications to experimental setup, and begin writing the results section of the report.
25	17/03/2020	Working on other coursework.
25	19/03/2020	All coursework including report gets extended due to the coronavirus. Work on other coursework.
26	02/04/2020	Progress with the report and begin working on the project poster.
27	06/04/2020	Run another baseline experiment using Linear Support Vector Machine and TF-IDF vectorization. Record the results.
28	15/04/2020	Write up experimental results. Continue working on poster.
29	29/04/2020	Submit poster and video demonstration.
30	04/05/2020	Do some final experiments. Write up last sections of the report. Make modifications to methods and experimental setup.
31	12/05/2020	Write up the evaluation and conclusion. Make some final modifications to the report.

7.2 Appendix B - Source Code

Source code is available at: https://github.com/oaarnikoivu/ainoa.

The commit log can be viewed at: https://github.com/oaarnikoivu/ainoa/commits/master.

Alternatively, the source code for the baseline algorithms and deep learning models can viewed and experimented with on Google Colaboratory:

Attention LSTM: https://colab.research.google.com/drive/1lZJpqv-38jmP7zCizHdy8mSu-IMlHlCZ?usp=sharing.

Text CNN: https://colab.research.google.com/drive/1wbjptVje2AIFjs_Ry-rSGAqq0ptO3dmI?usp=sharing.

Baselines: https://colab.research.google.com/drive/1useAkZrNN0IKXWCi3FJ3FIOH23EYBJ70?usp=sharing.

Dataset analysis: https://colab.research.google.com/drive/1cqbUipO2O55O4R2AezMFcyqNfMswN6dz?usp=sharing.

7.3 Appendix C - Application

The application can be accessed at: https://ainoa.netlify.app.

7.4 Appendix D - Poster

Detecting emotion from text using Deep Learning

Oliver Aarnikoivu & Eyad Elyan

Introduction

Currently, a vast majority of research which has been applied with regards to sentiment analysis has been focused on classifying text as either “positive” or “negative”. Evidently, if we can move from a binary classification task into analysing and detecting distinct emotions, this could lead to advancements in various fields. However, it’s difficult to gain an understanding on how we define ”emotion” due to the complexity of human behaviour. Emotion can be expressed in so many different ways, such as facial expressions, gestures, speech, text and even from less obvious indicators such as heart-rate, skin clamminess, temperature, and respiration velocity.

Nevertheless, since 1979, an illustration provided by the psychologist Robert Plutchik (Plutchik 1979) has been widely used to demonstrate how different emotions can blend into another creating new ones.

Figure 1: Plutchik Wheel of Emotions (Plutchik 1979)

These emotions are (joy, trust, fear, surprise, sadness, disgust, anger and anticipation). If we can agree that emotions can be categorised into these distinct labels, it begs the question on whether it is possible to convey these emotions through text?

Project Aim

This aim of this project is to assess the ability of different deep learning models to classify a text as having one, or more, emotions for eight of the (Plutchik 1979) categories plus Optimism, Pessimism, and Love. The model should be able to generalise adequately to unseen data.

Methods

This project uses the *SemEval Task 1: Affect in Tweets E-c* dataset. The dataset consists of 6838 training examples, 886 validation examples and 3259 testing examples. The experiment is tested using both a Text CNN (Convolutional Neural Network) and Attention LSTM (Long short term memory network) proposed by the authors (Kim 2014) and (Zhou. et al 2016) respectively. Due to the limited amount of training data, we make use of transfer learning such that the embedding layer of both models is initiated using both pre-trained GloVe (Global vectors for word representation) vectors and BERT (Bidirectional Encoder Representations from Transformers) embeddings, generated from a pre-trained BERT transformers model. While the two chosen model architectures are considerably different, they have been selected due to their ability to identify words within a sentence regardless of its position.

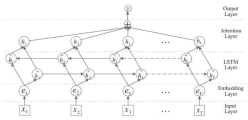


Figure 2: Attention LSTM Model Architecture (Zhou et al. 2016)

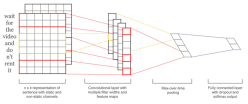


Figure 3: Text CNN Model Architecture Kim (2014)

Figures and Results

	Accuracy (Jaccard)	Micro F1	Macro F1
Attention LSTM + GloVe	0.4692	0.6010	0.4532
Attention LSTM + BERT	**0.5438**	**0.6684**	**0.4907**
Text CNN + GloVe	0.4066	0.5472	0.3874
Text CNN + BERT	0.5270	0.6592	0.4625

Table 1: Performance comparison of Attention LSTM and Text CNN.

Model	Anger	Anticipation	Disgust	Fear	Joy	Sadness	Surprise	Trust	Avg
Attention LSTM + GloVe	0.65	**0.25**	0.59	0.64	0.74	0.56	**0.14**	**0.16**	0.47
Attention LSTM + BERT	**0.76**	0.19	0.71	0.68	**0.82**	**0.65**	0.11	0.07	**0.50**
Text CNN + GloVe	0.63	0.00	0.58	0.55	0.72	0.46	0.11	0.00	0.38
Text CNN + BERT	**0.76**	0.03	**0.72**	**0.70**	**0.82**	0.62	0.00	0.05	0.46

Table 2: Attention LSTM vs. Text CNN on Plutchik Categories by F1 Score.

Based on our results, it’s evident that the Attention LSTM performs better for emotion detection. The Attention LSTM using BERT embeddings outperformed both Text CNN models as well as the Attention LSTM model using GloVe embeddings. Moreover, with regards to both the Attention LSTM and Text CNN, in both cases the model using the embeddings produced by the pre-trained BERT model proved better results. This suggests that the contextualised embeddings produced by BERT may be superior in comparison to the non-context dependent vectors produced by GloVe. Furthermore, as shown by the table above, we see that in terms of the Micro F1 score, the results achieved by the Attention LSTM model using BERT embeddings are comparable to the top 10 official SemEval Task 1 (multi-label emotion class.) competition results.

#	Team Name or User	Accuracy (Jaccard)	Micro F1	Macro F1
1	zimkjh	0.593	0.704	0.565
2	psyML	0.574	0.697	0.574
3	prabod	0.574	0.689	0.500
8	ventakesh-1729	0.558	0.677	0.476
9	Seekers7	0.555	0.667	0.505
10	ELiRF-UPV	0.552	0.658	0.512

Table 3: SemEval 2018: Task 1 E-c (multi-label emotion class.) English leaderboard, snippet of the top 10 results.

This suggests that the model does an adequate job taking into account the label imbalance of the dataset.

Figure 4: Example of attention visualisation for emotional classification

The figure above displays the words that the attention model considers the most significant with regards to it’s predictions. The color intensity and word size corresponds to the weight given to each word. We can see that the model is successfully able to “place attention” towards words which correlate to the predicted emotions.

Conclusion

This project compares an Attention-based bidirectional LSTM to a CNN using transfer learning such that the embedding layer is initialised with pre-trained GloVe and BERT embeddings. The results achieved by the Attention LSTM model using BERT embeddings proved comparable to the current top 10 official SemEval Task 1: E-c (multi-label label emotion class.) competition results. The results displayed in Table 2 indicate that both chosen models have a difficult time generalising to categories with just a few training examples, whereas categories with a sufficient amount of training data perform well. This suggests that the labels with a worse class imbalance would benefit from having a larger dataset.

Acknowledgments

I would like to give a special thank you to my honours supervisor Dr. Eyad Elyan, whose support and guidance throughout this project has been invaluable.

References

Kim, Y. (2014), Convolutional neural networks for sentence classification, in ‘Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)’, Association for Computational Linguistics, Doha, Qatar, pp. 1746–1751.
URL: https://www.aclweb.org/anthology/D14-1181

Plutchik, R. 1979. Emotions: A general psychoevolutionary theory. 1.

Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H. & Xu, B. (2016), Attention-based bidirectional longshort-term memory networks for relation classification ,in ‘ACL’.

ROBERT GORDON UNIVERSITY ABERDEEN
SCHOOL OF COMPUTING SCIENCE AND DIGITAL MEDIA

BSc (Hons) Computer Science

7.5 Appendix F - Project Proposal

Detailed Project Proposal

First Name:	Oliver
Last Name:	Aarnikoivu
Student Number:	1502639
Supervisor:	Eyad Elyan

Defining your Project

1.1 Project title

Help: a brief statement about what you are actually going to do.

Title: Detecting emotion from text using Deep Learning

My project will investigate the use of deep learning algorithms and natural language processing techniques to detect emotion from text.

1.2 Background

Help: Provide the background to your project. This section should highlight the main topics in the area you are going to research. Essentially what is the project about, what has been done before and why is this project important? ~500 words

My project aims to investigate the use of recent state of the art deep learning algorithms to detect emotion from text. Emotion detection has become popular due to the vast potential in applications that can be applied to areas such as political science, human-computer interaction, artificial intelligence, psychology and more.

Currently, the majority of research that has been applied to text involves sentiment analysis, such as classifying a specific set of text as either "positive" or "negative". Evidently moving from negative and positive sentiments into analysing distinct emotions can lead to advancements in various applications in the fields mentioned above and potentially open ways to new areas of research. For example, in recommender systems, emotion detection can be used to provide recommendations based on the emotional state of the user. In marketing, the ability to identify emotions produces a greater understanding towards the behaviour and satisfaction of a business's customers. Psychologists could study mental health problems and even detect depressive symptoms in people based on the type of emotion that an algorithm would detect. This could even extend to analysing the overall happiness of people on the planet as well as to other societal wellbeing metrics.

Up until recently, research and experimentation in NLP has been focused on the use of Recurrent Neural Networks (RNN) and word embedding techniques, however, research is trending towards the use of pre-trained models (Transfer Learning) and Transformer techniques. Various models have been released which have been trained on multilayer RNN's such that they can learn word embeddings from context. The weights of these models can then be applied to

problems of a different use case, such as detecting emotion from text. Using transformers and transfer learning has proved significant results. There are examples of the use of transformers and transfer learning on sentiment analysis and some other multi-label classification problems, nevertheless, there are scarce sources on the use of these techniques on detecting emotion. My project will focus on researching these techniques, however, it's important to note that they are very costly in terms of computation, thus, my project will also research other methods in NLP such as the use of Long short-term memory neural networks (LSTM's) and word embedding techniques such as Word2Vec and GloVe.

According to "Emotion Detection in Text: A Review" by Armin Seyeditabari, Narges Tabari and Wlodek Zadrozny, there is a limited amount of public data specifically classifying the emotion of text. The paper indicates that one of the most well-known sources for emotionally labelled text is ISEAR (International Survey on Emotion Antecedents and Reactions). It contains the responses of over 3000 people from the world who were asked to describe situations they have experienced and how they reacted to them. Thus, this dataset can prove to be suitable for the use case of my project. Another dataset to consider is "SemEval 2018 Task 1: Affect in Tweets" which contains tweets that have been classified into 1 out of the 11 emotions (anger, anticipation, disgust, fear, joy, optimism, pessimism, sadness, surprise and trust). The paper also suggests the use of emotion lexicons where specific words are associated with emotions. An emotion lexicon dataset to consider is the "NRC Word-Emotion Association Lexicon" by Saif Mohammad where words are linked with eight different emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust).

1.3 Motivation

Help: To whom is this project important? A project must address a question/problem that generates a small piece of new knowledge/solution. This new knowledge/solution must be important to a named group or to a specific client (such as a company, an academic audience, policy makers, people with disabilities) to make it worthwhile carrying out. This is the ***motivation*** for your project. In this section you should address who will benefit from your findings and how they will benefit. ~300 words

> **Example** 1: If you intend to demonstrate that a mobile application that automates class registers at RGU will be more efficient than paper-based registers - the group who would be interested in knowing/applying these findings would be both academic and administrative staff at RGU and they would benefit by time saved and a reduction in their administrative workload.
>
> **Example** 2: You are demonstrating that a particular 3D model design increases realism in 3D environments. The group that would be interested would be games designers or developers of 3D virtual environment applications. The would benefit from producing more realistic environments that could increase sales of their products.
>
> **Example** 3: You have designed a new network topology for IrishOil plc's new Aberdeen headquarters. The interested group would clearly be IrishOil. They would benefit from easier maintenance and improved security of their computer network.

As mentioned in section 1.2, the potential for new applications and areas of research is vast when it comes to understanding emotion from text. I believe the groups that would most benefit from this would be businesses that share a close relationship with their customers as the potential to tune a business's marketing tactics, customer relationship techniques and more would benefit from understanding the discrete emotion of a customer as opposed to just a

binary classification of sentiment, i.e. “positive” or “negative”. Of course, the implications of this can raise serious ethical concerns which I aim to discuss in my project. In addition, on a larger scale, the ability for a machine to be able to detect emotion from text can assist in a governments ability to measure people’s emotional, social and overall wellbeing. The potential for new applications and business’s is enormous. For example, we can imagine an application that can detect severe distress in people with mental health problems such as depression, anxiety etc and pinpoint these people to sources that could provide the necessary support, such as psychologists around their area, people with similar problems and educational resources.

1.4 Aim & Objectives

Help: Outline what are the main things your project is going to do and what steps or milestones will be used to achieve this aim. The Aim is unlikely to change throughout your project; however, the objectives are likely to adapt to your ongoing research and development. In particular it is highly likely that you may wish to split objectives into sub-objectives as work progresses. A good clear set of objectives give you something to evaluate your final project against.

Example : For the timetable app outlined above
Aim: To create a functioning attendance application that efficiently automates the taking of class registers.
Objective 1: study existing register system in place at RGU and identify weaknesses
Objective 2: research existing automation technology’s and identify and evaluate those that may be appropriate to taking in class registers
Objective 3: Implement chosen technologies to create prototype application
Objective 4: Conduct user trials to evaluate capabilities of prototype application
Objective 5: Create a refined application incorporating feedback from user trials

Aim: To develop an effective machine learning/deep learning model in detecting emotion from text.

Objective 1: Research existing papers on emotion detection and sentiment analysis to get an idea on the current techniques which are being used, and to identify any possible weaknesses.

Objective 2: Evaluate researched technologies and identify techniques appropriate to detecting emotion from text.

Objective 3: Explore publicly available datasets/emotion lexicons which can be applied to my deep learning model and evaluate deep learning libraries/frameworks to use.

Objective 4: Begin implementation and testing of researched techniques such as Recurrent neural networks, Transfer Learning and more on my chosen dataset.

Objective 5: Record and evaluate the results of my model. This will likely include a thorough review on the metrics my model received.

Objective 6: Evaluate possible ways to improve the model, such as dealing with underfitting or overfitting, imbalanced data, learning rate etc.

Objective 7: Visualise and discuss final results

1.5 Key Techniques

Help: Perform some initial research into the area and outline what techniques you my research in further detail here. The techniques you cover here should include references to the papers where you have sourced the information. The techniques mentioned here are very likely to become the section headers in your literature review.

Transfer Learning & The Transformer

If using a dataset such as ISEAR which only consists of approximately 3000 responses, it's evident that a dataset of this size is not enough information for a machine to be able to learn an entire language, let alone emotion. Reason for this is because language is extremely nuance. For example, on various online forums people like to make use of sarcasm and other forms of language which can make it quite difficult for a model to correctly "understand" the type of language that is being used. The key technique that I plan to use in order to counter this issue is Transfer Learning. The idea behind Transfer Learning is to begin with a pre-trained model that has been trained to do a different task yet is somewhat related to the task at hand. As I briefly described in section 1.2, sentiment analysis is quite similar to emotion detection, thus, using the pre-trained weights of a model trained on sentiment analysis on my model of emotion detection may prove significant improvements in terms of accuracy and other metrics.

Another technique to consider is the use of Transformer. The Transformer, as described in the paper "Attention Is All You Need", is a dominant sequence transduction model based on complex convolutional or recurrent neural networks which also include an encoder and a decoder.

Some of the key models that are being used in NLP currently are BERT (Bidirectional Encoder Representations from Transformers), ELMo (Embeddings from Language Models) and GPT-2.

Although the performance of my model may improve significantly if using a transformer model, it's important to note that this is computationally expensive at its current state, thus, I plan to investigate the use of "lighter" varieties on cloud based Jupyter notebooks such as Google Collab, which provides free access to a GPU.

In addition, it may also prove to improve metrics if using a Language Model as the pre-trained model. The idea behind a Language Model is to train a model to predict the next word in a sentence. This is beneficial because it provides context to differentiate between words and phrases that sound alike and allows my model to learn representations that contain information about the structure of natural language. It is able to predict the next word in a sentence mainly as language models are trained on extremely large datasets. An example of this is WikiText 103 which is a collection of tokens extracted from Wikipedia articles.

Recurrent neural networks:

Although the direction of research and experimentation in NLP is shifting towards transformer models and transfer learning, this doesn't completely eradicate the use of RNN's (Recurrent Neural Network's). RNN's are useful as they make use of sequential information. Specifically, a key RNN architecture to research is LSTM (Long short-term memory) neural networks. LSTM's differ from standard feedforward neural networks as they are able to learn long-term dependencies allowing them to maintain information in memory for longer periods of time.

Word embeddings:

Word embeddings are a form of word representation which allocates a human understanding of language to a computer. It represents the words in a coordinate system in which words that are closely related are placed closer together. Task performance has been shown to improve when using a representation that better understands the meaning of words as input. Examples of Word Embedding techniques include Word2Vec and GloVe.

Sources:

https://nlp.stanford.edu/pubs/glove.pdf

https://arxiv.org/pdf/1803.11175.pdf

https://gluon-nlp.mxnet.io/examples/sentiment_analysis/sentiment_analysis.html

https://towardsdatascience.com/machine-learning-text-classification-language-modelling-using-fast-ai-b1b334f2872d

https://medium.com/huggingface/multi-label-text-classification-using-bert-the-mighty-transformer-69714fa3fb3d

https://arxiv.org/pdf/1810.04805.pdf

https://arxiv.org/pdf/1801.06146.pdf

https://openai.com/blog/better-language-models/

https://github.com/huggingface/transformers

https://arxiv.org/pdf/1706.03762.pdf

https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html

https://www.ibm.com/developerworks/community/blogs/nlp/entry/tokenization?lang=en

https://thegradient.pub/nlp-imagenet/

https://medium.com/@Hironsan/why-is-word-embeddings-important-for-natural-language-processing-6b69dd384a77

https://arxiv.org/pdf/1301.3226.pdf

https://blog.einstein.ai/the-wikitext-long-term-dependency-language-modeling-dataset/

1.6 Legal, Social, Ethical, Professional and Security issues

Help: Here you should discuss any legal, social, profession and security issues that you believe may occur during the course of your project. It is not acceptable to write none in this box, all projects, regardless of focus will have to address issues in one, or more, of these categories. This is an extremely important part of your honours project to which there is no correct answer, this section must be fully discussed with your Honours Supervisor.

> **Example 1** : In the class register example above – there would be a Legal and Security issue with the gathering and storage of student data. There may be a social constraint as you may be relying on a user to have access to a specific technology. There will need to be consideration of user accessibility.
>
> **Example 2** : A 3D model design may have ethical considerations in its evaluation. What if your model made users feel nauseous. Social constrains may again be access to technology or accessibility issues.
>
> **Example 3** : You network design need to adhere to specific company policies. You would need to consider the possibility that your design could be wrong, compromising the company's security.

If my model proves to be accurate in detecting emotion from text, it's possible that this could be used in an unethical manner when testing on text made public by users on social media forums such as Reddit or Twitter.

As I plan on testing my model on text from Reddit or Twitter, it's important that I don't reveal any information about the user such as the name or username.

1.7 Project Plan

Help: This is the project plan as to how you will go about achieving the objectives of the project.

> **Example**: In the class register example above the research plan may involve:
> Collecting and analyzing paper-based registers in a given class on five occasions.
> Identifying the error rate average on these occasions
> Researching existing automation techniques
> Designing and implementing a mobile application that automatically records attendance in class.
> Deploying the application in the class on five occasions.
> Identifying the error rate average of the mobile application on these occasions.
> Comparison of data and summary of findings.

1. Background research

1.1. Project proposal

1.2. Literature review

1.3. Research on technical implementation (python libraries, methodologies, etc.)

2. In-depth review of previous work

2.1. Look at examples of Transfer Learning in similar domains such as sentiment analysis

2.2. Review examples of Language Models & Transformers

2.3. Review implementations of neural network architectures in NLP

3. Design

3.1. Begin design of program (UML Diagrams)

3.2. Evaluate libraries to use (PyTorch, Keras, Scikit Learn, etc…)

3.3. Evaluate Neural Network architectures for emotion classification. (LSTM, BiLSTM, CNN)

3.4. Investigate frameworks for Transformers & Transfer Learning (pytorch-transformers, transformers, BERT, ULMFit, etc…)

4. Implementation

4.1. Gather dataset

4.1.1. Begin implementing models based on researched techniques such as Transfer Learning, Transformers, etc.

4.1.2. Evaluate methods to improve performance of the models (underfitting, overfitting, learning rate, etc)

5. Testing

5.1. Test model efficiency on test data & other forms of text from online sources

6. Report

6.1. Discuss results and testing

6.2. Discuss literature review

6.3. Discuss any changes that could have been implemented

7. Visualisation

7.1. Visualise results

7.2. Implement poster & presentation

1.8 Ethics Form

You must include in your signed ethics form in this submission or you will not be able to continue the project.

Provided in separate file.

7.6 Appendix E - Ethics Form

ROBERT GORDON UNIVERSITY•ABERDEEN

STUDENT PROJECT ETHICAL REVIEW (SPER) FORM

The aim of the University's *Research Ethics Policy* is to establish and promote good ethical practice in the conduct of academic research. The questionnaire is intended to enable researchers to undertake an initial self-assessment of ethical issues in their research. Ethical conduct is not primarily a matter of following fixed rules; it depends on researchers developing a considered, flexible and thoughtful practice.

The questionnaire aims to engage researchers discursively with the ethical dimensions of their work and potential ethical issues, and the main focus of any subsequent review is not to 'approve' or 'disapprove' of a project but to make sure that this process has taken place.

The *Research Ethics Policy* is available at
www.intranet.rgu.ac.uk/credo/staff/page.cfm?pge=7060

Student Name	Oliver Aarnikoivu
Supervisor	Eyad Elyan
Project Title	Detecting emotion from text using Deep Learning
Course of Study	BSc (Hons) Computer Science
School/Department	School of Computing Science and Digital Media

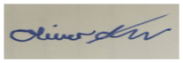
Part 1 : Descriptive Questions			
1	Does the research involve, or does information in the research relate to:	Yes	No
	(a) individual human subjects		X
	(b) groups (e.g. families, communities, crowds)		X
	(c) organisations		X
	(d) animals?		X
	Please provide further details:		
2	Will the research deal with information which is private or confidential?		X
			X
	Please provide further details:		

Part 2: The Impact of the Research			
3	In the process of doing the research, is there any potential for harm to be done to, or costs to be imposed on	Yes	No
	(a) research participants?		X
	(b) research subjects?		X
	(c) you, as the researcher?		X
	(d) third parties?		X
	Please state what you believe are the implications of the research:		
4	When the research is complete, could negative consequences follow:	Yes	No
	(a) for research subjects		X
	(b) or elsewhere?		X
	Please state what you believe are the consequences of the research:		

Part 3: Ethical Procedures			
5	Does the research require informed consent or approval from:	Yes	No
	(a) research participants?		X
	(b) research subjects		X
	(c) external bodies		X
	If you answered yes to any of the above, please explain your answer:		
6	Are there reasons why research subjects may need safeguards or protection?	Yes	No
			X
	If you answered yes to the above, please state the reasons and indicate the measures to be		
7	Has PVG membership status been considered?		X
	(a) PVG membership is not required.		X
	(b) PVG membership is required for working with children.		X
	(c) PVG membership is required for working with protected adults.		X
	(d) PVG membership is required for working with both children and protected		X
	If you answered yes to (b), (c) or (d) above, please give details:		
8	Are specified procedures or safeguards required for recording, management, or storage of data?	Yes	No
			X
	If you answered yes to the above, please outline the likely undertakings:		

Part 4: The Research Relationship			
9	Does the research require you to give or make undertakings to research participants or subjects about the use of data?	Yes	No
			X
	If you answered yes to the above, please outline the likely undertakings:		
10	Is the research likely to be affected by the relationship with a sponsor, funder or employer?	Yes	No
			X
	If you answered yes to the above, please identify how the research may be affected:		

Part 5: Other Issues			
11	Are there any other ethical issues not covered by this form which you believe you should raise?	Yes	No
			X

Statement by Student I believe that the information I have given in this form is correct, and that I have addressed the ethical issues as fully as possible at this stage.			
Signature		Date	30/09/2019

If any ethical issues arise during the course of the research, students should complete a further Student Project Ethical Review (SPER) form.

The *Research Ethics Policy* is available at
www.intranet.rgu.ac.uk/credo/staff/page.cfm?pge=7060

Part 6: To be completed by the supervisor			
12	Does the research have potentially negative implications for the University?	Yes	No
			X
	If you answered yes to the above, please explain your answer:		
13	Are any potential conflicts of interest likely to arise in the course of the research?		No
	If you answered yes to the above, please identify the potential conflicts:		
14	Are you satisfied that the student has engaged adequately with the ethical implications of the work? [In signifying agreement, supervisors are accepting part of the ethical responsibility for the project]	Yes	
	If you answered no to the above, please identify the potential issues:		
15	**Appraisal:** Please select one of the following		
	The research project should proceed in its present form – no further action is required	X	
	The research project requires ethical approval by the School Ethics Review Panel		
	The research project needs to be returned to the student for modification prior to further action		
	The research project requires ethical review by an external body. If this applies please give details		
	Title of External Body providing ethical review		
	Address of External Body		
	Anticipated date when External Body may consider project		

Affirmation by Supervisor			
I have read the student's responses and have discussed ethical issues arising with the student. I can confirm that, to the best of my understanding, the information presented by the student is correct and appropriate to allow an informed judgement on whether further ethical approval is required.			
Signature	Eyad Elyan	**Date**	29.09.2019