Master Thesis

Target-Based Sentiment Analysis as a Sequence Tagging Task

Zoe Gerolemou

Master Thesis DKE-19-13

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science of Artificial Intelligence
at the Department of Data Science and Knowledge Engineering
of the Maastricht University

Thesis Committee:

Dr. Johannes C. Scholtes
Dr. Jerry Spanakis

Maastricht University
Faculty of Science and Engineering
Department of Data Science and Knowledge Engineering

June 20, 2019

Acknowledgements

I would like to express my gratitude and appreciation to Dr. Johannes C. Scholtes and Mr. Jeroen Smeets for their excellent guidance and inspiring mentorship that helped me enrich my knowledge and grow as an Information Retrieval and Text Mining enthusiast. Along with them, I want to thank everyone at ZyLAB who welcomed me to their team so warmly.

Finally, I would like to thank my family and friends for standing next to me every step of the way. I, wholeheartedly, dedicate this thesis to them. Nothing would have been possible without their priceless support.

Abstract

With focus on the online reviews domain, this thesis attempts to provide a complete solution to the sentiment analysis task (involving the extraction of the opinion holder, the polarity of the underlying sentiment and the target) by treating it as a sequence tagging problem. For the first component, a rule-based approach is devised which results to an F1-Score of 0.9200. The other two subtasks are tackled using a machine learning approach. To support the training process, a new dataset, which combined sentences from two different review-based datasets, was constructed and partially annotated to indicate the target of sentiment. The best performing models are LSTM-based and make use of the newly-introduced context-sensitive BERT embeddings. The polarity and target components yield F1-Scores of 0.9001 and 0.8959, respectively, over all the possible classes. For the purposes of this research, many challenges and issues related to the nature of the problem are addressed such as class imbalance and the need for meaningful data augmentation techniques to increase the size of the training set and make the use of LSTMs possible. For both of them, new effective approaches are proposed and evaluated.

Contents

1 Introduction 3

2 Related Work 6
2.1 Sequence Tagging . . . 6
2.1.1 Hidden Markov Models . . . 6
2.1.2 Maximum Entropy Markov Models . . . 7
2.1.3 Conditional Random Fields . . . 8
2.1.4 Deep Learning . . . 9
2.2 Language Modelling . . . 10
2.2.1 Word2Vec and GloVe . . . 10
2.2.2 Bidirectional Encoder Representations from Transformers 11
2.2.3 Contextual String Embeddings . . . 12
2.3 Natural Language Processing Techniques . . . 14
2.3.1 Co-Reference Resolution . . . 14
2.3.2 Dependency Parsing . . . 14
2.4 Sentiment Analysis . . . 14
2.4.1 Sentiment Polarity . . . 14
2.4.2 Sentiment Target . . . 17

3 Research Questions 20

4 Dataset 21
4.1 Existing Dataset Availability . . . 21
4.2 Constructing a New Dataset . . . 22
4.3 Annotation . . . 23
4.4 Structure and Format . . . 25

5 Implementation 26
5.1 Proposed Approach . . . 26
5.2 Overall Architecture . . . 27
5.3 Preprocessing . . . 27
5.3.1 Sentence Representation . . . 27
5.3.2 Handling Class Imbalance . . . 28
5.3.3 Data Augmentation . . . 32

5.4 Feature Extraction 33
5.4.1 Lexicographical and Syntactical Features 33
5.4.2 Word Embeddings 34
5.5 Sentiment Triple Extraction 35
5.5.1 Target and Polarity Prediction 35
5.5.2 Opinion Holder Extraction 36
5.6 Visualisation 40

6 Results **44**
6.1 Target and Polarity Prediction 44
6.2 Performance Evaluation 44
6.2.1 CRF-Based Models 46
6.2.2 LSTM-Based Models 49
6.3 Opinion Holder Extraction 57
6.3.1 Performance Evaluation 57
6.3.2 Error Analysis 57

7 Conclusions **60**

8 Future Work **63**

Chapter 1

Introduction

Being able to detect the sentiment behind verbal and written speech had always been a quite difficult and challenging task for humans. As languages were evolving over time and more complex lexical and syntactical structures were making their appearance, people started having trouble to precisely interpret underlying emotions. This was even more acute in written text where ambiguities arose very frequently with the absence of any facial expressions, gestures and voice tone fluctuations that accompany the oral communication.

The aforementioned challenging nature of this task is a result of both the subjectivity, but also the ambiguity that characterises natural language. Depending on the person who assesses a language extract, this could be interpreted in many different ways. Factors that usually affect the way people perceive language range from cultural and age-related to their own individual emotional skills and capabilities. Phenomena like irony and sarcasm further increase the difficulty due to the complexity behind their formation. These structures often alter or even invert the meaning of words in a sentence; something that makes their interpretation harder.

Amid the rise of social media and the different kinds of microblogging, people are finally offered the means and the ways to share their thoughts publicly in online platforms without the need of building and maintaining their own websites as it was the norm in the early 2000s. Some of these services are solely dedicated to the expression of opinions about products and services (e.g. Yelp!, Trustpilot) in the form of reviews whilst others (e.g. Twitter) allow the more general exchange of thoughts between their users regarding certain entities or topics. The spontaneity behind these postings, which usually do not undergo any particular polishing by their authors, are important sources of information for how the targets of these opinions are received and appreciated by the general public so that they can adjust their strategies accordingly.

After the major advancements in Natural Language Processing (NLP) that allowed the employment of different computational techniques for analysing human language, it was only natural to proceed with the investigation of the effectiveness of these approaches into detecting sentiment in text. Researchers introduced efficient ways of representing text corpora which preserved the linguistic and contextual relationships between words and prepared them to be used for training the appropriate models and accomplishing the aforementioned goal. Most prominent of them are, undoubtedly, the different forms of word embeddings which are frequently part of the state-of-the-art solutions in different NLP problems. Sentiment analysis does not only concern the polarity of an expressed emotion but also denotes a relationship between two entities: the opinion holder and the target of this opinion. Apart from the academic interest behind this task, it has found a lot of applications in various fields in the industry, too. Companies use these approaches to evaluate the performance of their products based on their appreciation by the public. Politicians can get an overview of what the public thinks of them or about certain topics that affect the society.

One other domain which can be significantly supported by the use of sentiment analysis is criminal and fraud investigation. Intense emotional patterns are noticed to emerge in relevant communication whenever there is some kind of obstruction in the criminal scene. This observation makes the creation of effective and efficient techniques for quickly and accurately extracting the sentiment from text corpora to be particularly important. Unfortunately, due to the sensitive personal data and the confidentiality involved in this form of communication, there is limited or even non-existent dataset availability. Thus, sentiment analysis methodologies especially targeted to it cannot be investigated as extensively as they would on text from other domains. However, it is highly likely that methods conceived and successfully applied in different contexts where similar language is used could be leveraged for these purposes, too.

A significant portion of the studies conducted involve the instances of Microblogging that are shared in platforms like Twitter or Facebook. Especially in the Twitter domain, numerous researches have attempted to shed a light to the attributes of a tweet that act as good indicators for understanding its underlying sentiment so that they can eventually construct automatic systems to analyse them from an emotional perspective. However, these observations are not always relevant to all kinds of text. The way this form of microblogging evolves over time is based on the comprehensiveness behind these messages. The case of Twitter in particular, where there is a strict limit of 140 characters per message, gave the first spark for the creation of abbreviations and shorthands that aim to convey the same meaning as other words without taking up much of the message quota. In addition to this, the grammatical and syntactical rules that are formally applied on the written form of languages are not always properly followed in a further attempt to save up space.

This particular interest in the second form of microblogging has imposed several significant consequences on the overall field. Since it is more widely researched, there are more datasets that allow the experimentation on sentiment analysis approaches for this form of text. This is an important limitation for studies that are interested to mine sentiment from more formal extracts of text since they cannot base the evaluation of the effectiveness of their approaches on Twitter-based datasets, for example.

To investigate whether it is possible to generalise these approaches to other more formal and structured forms of text, this study will approach the sentiment analysis problem from the perspective of the first class of microblogging text with the use of sentences extracted from reviews.

The main aim of this thesis is to introduce a complete solution to the sentiment mining problem by abstracting it as sequence tagging task. The ultimate goal will be to label each individual word in a sequence based on whether it belongs to any of the sentiment components. For example, by giving the sentence "*I, really, enjoyed watching Bohemian Rhapsody last night.* as input, the proposed system will have to detect 'I' as the *opinion holder*, 'Bohemian Rhapsody' as the *target* and determine that the *sentiment polarity* is 'Positive'. These three elements together will form the so-called 'sentiment triple' which can be thought as a summarisation of this sentence from a sentimental point-of-view.

In addition to the primary goals of this research, some elementary issues will be addressed; firstly, an attempt will be made to overcome the problem of dataset shortage by following a process of target-oriented annotation on a dataset consisted of reviews about products and services. Secondly, the issue of class imbalance and its effects on this particular context will be thoroughly investigated and a number of effective solutions for mitigating it will be presented and assessed. Amongst them, two novel approaches of augmentative nature will be proposed that can be employed for both the construction of a more-balanced dataset but also for increasing the number of training samples. Later, BERT and flair, two newly-introduced types of contextual embeddings, will be used for sentiment analysis for the first time as part of this research.

Chapter 2

Related Work

2.1 Sequence Tagging

An important characteristic of natural language is its hierarchical structure. A standalone piece of text such as an essay can be broken down into paragraphs and even further into sentences. A sentence consists of a set of tokens and, while being complete on itself, it conveys a single statement or idea. It follows that, from a top-to-down perspective, documents are sequences of paragraphs and paragraphs are sequences of sentences.

Therefore, an expected consequence from the above is that the different NLP tasks which involve the tagging of the individual constituents of these language sequences can be viewed as sequence tagging problems. *Sequence tagging* (or alternatively *Sequence labelling*) is defined as the *assignment of tags to discrete elements ('words') in a sequence*[10]. Some of the most popular sequence tagging tasks in NLP are Named-Entity Recognition (NER), Part-of-Speech (PoS) tagging and chunking. Sequence labelling is also quite successfully employed in sentiment analysis for the extraction of emotions and sentiment aspects.

2.1.1 Hidden Markov Models

The vast majority of approaches that are currently employed for this class of problems are based on linear statistical models. The foundations were set by the Hidden Markov Models (HMMs) which found a wide range of applications; some of the most important being speech processing and language modelling. For PoS Tagging in particular, two successful HMM-based methods were proposed in 1985 and 1992 by Jelinek[16] and Kupiec[18], respectively.

However, these methods are restricted by some inherent limitations of the HMMs. Firstly, they suffer from poor representation capabilities, especially when it comes to observations that need to be described in terms of numerous features which overlap with each other. Their second problem lies on the nature of the underlying model. Despite the fact that their aim is to solve a problem of conditional nature in which the observations are known, they use a generative joint model that attempts to maximise the likelihood of the observations[21] instead. Therefore, in addition to the prediction of labels, they also estimate the likelihood of a word ('observation') sequence to be generated via this model.

The way these models estimate the possible transitions is by focusing solely on the observed words themselves. This means that, in order to be durable enough to detect different transitional patterns, they need sufficient data demonstrating each and every one of them. However, this is not always feasible.

2.1.2 Maximum Entropy Markov Models

In an attempt to overcome the above shortcomings, McCallum et *al.*[21] introduced Maximum Entropy Markov Models (MEMMs). An MEMM acts as an extension to the plain HMM framework in the sense that it additionally employs *Maximum Entropy* (also known as *Multinomial Logistic Regression*). This modification offers the desirable freedom when it comes to the number and type of features employed. In particular, it allows the use of feature functions since, in contrast with the separate transition and observation functions utilised by HMMs, MEMMs employ combined state-observation transition functions. These functions express the possible transitions in terms of many features which may have dependencies between them. The use of such high level features eliminates the need for more training data which is evident in HMMs. This is because the different transitional patterns can now be extracted and learned by simply looking at the feature representations of the different words.

This new model shifts towards a conditional environment where the probability to end up in a particular state is represented given a specific observation and the previous state. A direct consequence of this is that the Markov Independence assumption is no longer preserved. The authors report that the results of the experimental process followed to evaluate their effectiveness are positive.

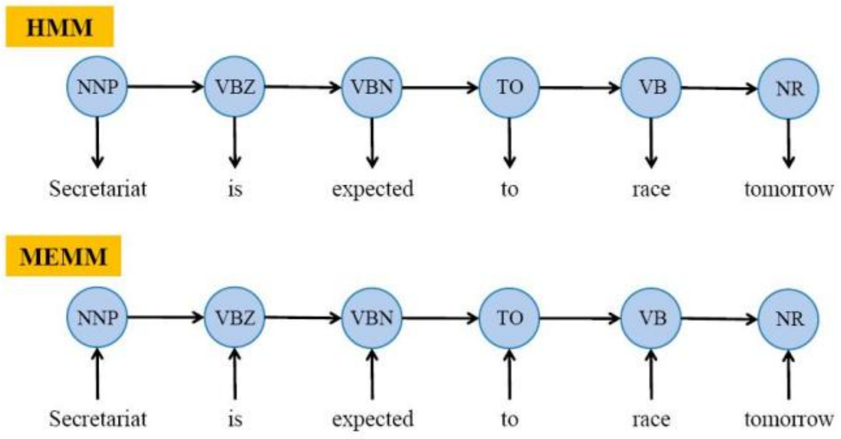


Figure 2.1: A diagram that juxtaposes the differences in the way HMMs and MEMMs model the same input sentence.

On their turn, MEMMs are attributed an important weakness: the so-called *label bias* problem. This concept was introduced and described by Leon Bottou in 1991[5]. The transitions that leave a state (i.e. outgoing transitions) are only compared with each other instead of competing with all the other transitions in the model. Thus, the score which each transition receives represents only its relative importance with respect to this particular state and not the overall, absolute one. By keeping in mind that the sum of the probabilities of all transitions in a state should be equal to one, it follows that states with fewer outgoing links are associated with higher probabilities and, therefore, they are given an unfair advantage.

2.1.3 Conditional Random Fields

Lafferty et *al.*[19] address the issue of label bias by introducing Conditional Random Fields (CRFs) which carry over all the positive attributes of MEMMs without being susceptible to this problem. They can be thought of as a sequential version of MEMMs which has the maximisation of the conditional probability of the output sequence as its optimisation objective. Due to their nature, they also consider the nearby points whenever they are trying to predict the output label for a given instance; meaning that they have the capacity to take context into account. What makes them immune to the label bias problem is that, instead of calculating the individual maximal entropy values for each state, they make use of its cumulative distribution over the entire sequence. The results of their experimental evaluation show that they generally outperform MEMMs with a margin of 10-20% relative error.

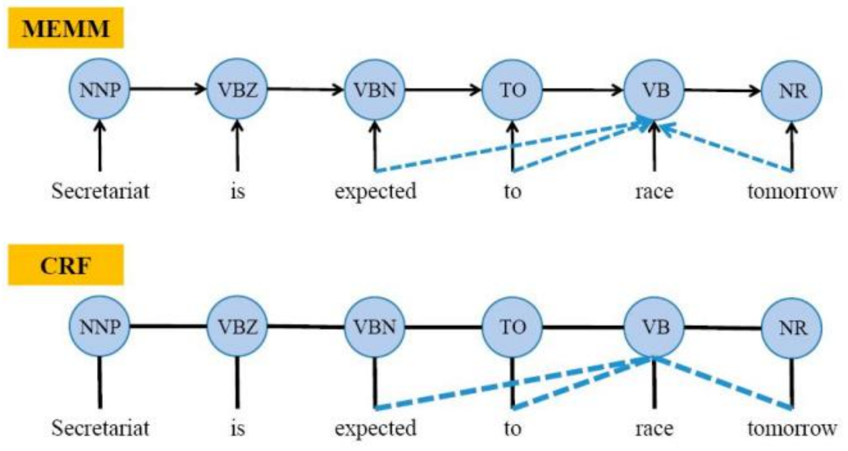


Figure 2.2: A diagram that juxtaposes the differences in the way MEMMs and CRFs model the same input sentence.

2.1.4 Deep Learning

The advances that followed the rise of deep learning in the last years brought a new perspective to many branches of machine learning. NLP and sequence tagging, in particular, were no exceptions to this. Apart from powerful and innovative architectures, ideal for application on different NLP tasks, they introduced new ways of representing the inputs in a dense manner. Some examples of these new forms of language modelling will be presented in Section 2.2.

Architectures that are consisted of convolutional and pooling layers provide a way to encode large feature sets into fixed-sized vectors, capable of capturing their most important and salient properties[13]. Collobert et *al.*[8] present a model that is based on a Convolutional Neural Network (CNN) which is complemented by an output layer that denotes the sentence level log-likelihood. As the authors themselves state, the configuration of this last layer in combination with the overall convex architecture are equivalent to the training of a CRF model.

Another collection of deep network architectures which attract a significant amount of attention by the Natural Language Understanding (NLU) research community are the Recurrent Neural Networks (RNNs). The reason behind this is their inherent ability to deal with sequential data. Additionally, they overcome an important limitation of CNNs which is the elimination of any structural information from the input data.

Irsoy and Cardie[15] investigate the effect from the use of deep narrow RNNs instead of shallow and wide ones in the context of opinion mining. They formulate the opinion extraction task as a sequence tagging problem on the token-level. Their work proves that their proposed configuration outperforms the others as well as several previous CRF-based baselines without the need to use any external resources or features of syntactic nature.

In their study, Mesnil et *al.*[22] experiment with various RNN configurations including the Elman-type and Jordan-type ones and evaluate their performance on the standard Airline Travel Information Systems (ATIS) benchmark. Both configurations manage to outperform the CRF baseline. After the introduction of the Long-Short Term Memory (LSTM) Networks to the literature and the appreciation of their advanced properties and indisputable superiority over the vanilla RNNs, it was only natural to be utilised for sequence labelling. Yao et *al.*[40] validate their performance on the ATIS benchmark and report their effectiveness over previous approaches.

Based on the above observations, Huang et *al.*[14] propose a variety of models for sequence labelling based on different (unidirectional or bidirectional) LSTM and CRF configurations. The combination of a Bidirectional LSTM with a CRF layer (called BI-LSTM-CRF) yields the best results by leveraging the ability of the bidirectional component to use both past and future features at each moment in time but also the tag information in the sentence level provided by the CRF layer. This configuration exceeds the previous state-of-the-art accuracy when applied on different PoS tagging, NER and chunking datasets.

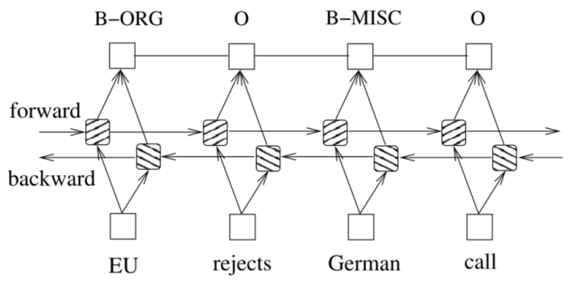


Figure 2.3: A diagram showcasing the architecture of the Bidirectional Long-Short Term Memory Network with a Conditional Random Field layer (BI-LSTM-CRF) model [14]

2.2 Language Modelling

In the past few years, a field of machine learning which received a significant portion of attention from the research community was, undoubtedly, transfer learning (TL). TL is defined as the class of approaches in which a model is developed and trained over a specific task and then used as the starting point for tackling others. It can be viewed as a clear demonstration of how useful the knowledge developed in a particular task of one domain could be when employed for other tasks in different domains.

NLP leverages techniques of TL to construct pre-trained language models which on their turn are fine-tuned for use in other downstream tasks. Some of the most prominent language models are currently widely-used for different purposes and will be introduced and explained in the following subsections.

2.2.1 Word2Vec and GloVe

Many attempts have been made in the past towards the invention of ways to represent words or phrases as vectors of real numbers so that they can be used as input to models for NLP purposes in an efficient manner. Some of the most prominent of them are Word embeddings. Being a prime example of TL, they provide the means for understanding the basic concepts and foundations of language and thus eliminate the need for more data when attempting to tackle specific linguistic tasks such as NER and sentiment analysis.

Of particular importance are Word2Vec[23] embeddings which are capable of reflecting the linguistic context of words. Each word is associated with a dense vector constructed in such way that words which frequently appear in the same context end up in close proximity to each other in the vector space.

An interesting alternative to Word2Vec are GloVe embeddings introduced by Pennington et *al.* [26]. GloVe capture the context in a count-based manner by considering the ratio of the co-occurrence probabilities of two words. Subsequently, they construct the corresponding co-occurrence matrix which carries information about the frequency of words in a given context. It is worth noting that GloVe have shown more aptitude in apprehending the global context than Word2Vec and are, therefore, preferred for tasks in which word analogy and similarity are heavily involved.

These representations are quite popular in the literature since they are pretty powerful and often achieve state-of-the-art results on various NLP tasks. Most of the times, they outperform older approaches for statistical language modeling because they provide more expressiveness and preserve the notion of ordering that characterises the natural language context. For example, 'Bag-of-Words' (BoW) is a widely-used statistical model which depicts the frequency distribution of the different n-grams in a text corpus. Usually, each n-gram is assigned with a single value related to some measure of frequency. For example, this can be a binary value denoting its presence/absence, the term frequency or term frequency - inverse document frequency (TF-IDF). The last one highlights the importance of a term in a text collection by calculating the ratio of its frequency in a document over its frequency in the entire corpus. As its name suggests, BoW ignores the ordering as well as any underlying relationships between the words. On the other hand, word embeddings associate each word with a vector that denotes their semantic similarity with the other words in this vocabulary. This allows for richer representation capabilities in a lower dimensional space.

However, both the abovementioned types of embeddings are subject to a number of important limitations. Firstly, they are language-dependent and context-independent. The same representation is produced for words associated with different meanings (i.e. polysemous words) and thus their contextual background is completely ignored. Their other weakness affects highly frequent words of opposite meanings which often appear together in the same context. Because these models maintain small windows of surrounding tokens, these words end up being encoded in a similar way (e.g. the words 'good' and 'bad'). As a direct consequence of these two characteristics, the two models underperform when they are used for tasks in which the actual contextual meaning words is of significant importance (e.g. sentiment analysis). Additionally, they cannot handle any words that are out-of-vocabulary. Last but not least, multi-token terms containing words that frequently co-occur in text because of this cannot be interpreted appropriately and encoded together in a composite way as they would have been expected to.

2.2.2 Bidirectional Encoder Representations from Transformers

One of the first approaches towards context-dependent word embeddings was presented by Peters et *al.* in 2018[27]. It involves the concatenation of two language models trained independently with two different directions in a shallow manner: left-to-right and right-to-left. They have named their embeddings *ELMo* which stands for *Embeddings from Language Models.* By being evaluated across six challenging NLP tasks such as question answering and sentiment analysis, it is experimentally proven that their new representation can be added to existing models and boost their performance. In all those tasks, they manage to clearly outperform the existing state-of-the-art.

In the same year, Devlin et *al.*[9] introduced an alternative approach to this. In contrast with ELMo, their representation (*Bidirectional Encoder Representations from Transformers* or *BERT*) utilises bidirectional pre-training based on masked language models. BERT represent the given input in the form of subwords for which they learn the embeddings. They belong to the class of fine-tuning methods meaning that they make use of the minimal amount of task-specific parameters. The *Generative Pre-trained Transformer* (OpenAI GPT) by Radford et *al.*[31] belongs to the same category; it is, however, limited by the fact that its pre-training is based on unidirectional language models. This means that it treats language modelling in a traditional manner by looking at it as a ”next-word-prediction task”. However, the most modern approach, which is reflected by the use of bidirectional representations, interprets it as a ”fill-in-the-gaps task” that can predict a masked word by looking at both the previous and the next tokens in the sequence.

The implementation of the BERT representation is based on TL. Firstly, it is pre-trained using two unsupervised prediction tasks; namely ”Masked Language Modelling” and ”Next Sentence Prediction”. For the former, a percentage of the input tokens is masked in random and then predicted. The latter investigates the relationship between two sentences. BERT's architecture is consisted of a Multi-Layer Bidirectional Transformer Encoder that uses the same configuration as the one presented by Vaswani et *al.*[37].

Devlin et *al.* evaluate the performance of these new embeddings empirically by applying them on eleven NLP tasks. The BERT fine-tuning approach outperforms the state-of-the-art for all these tasks. It is worth noting that they result in a 0.076 absolute improvement on the GLUE benchmark (i.e. receive a score of 0.804) and of 0.15 on the F1-Score of the SQuAD v1.1 Question Answering (i.e. 0.935).

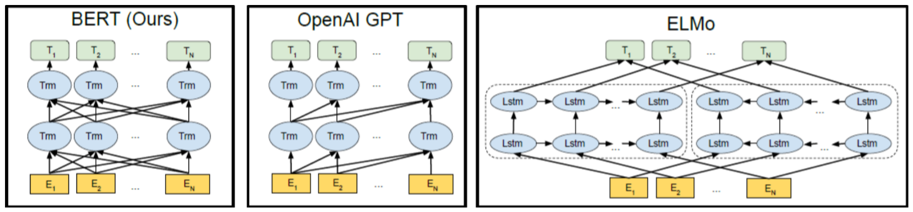


Figure 2.4: A diagram that highlights the differences between three context-sensitive architectures (BERT, OpenAI GPT and ELMo) [9]

2.2.3 Contextual String Embeddings

An important milestone in NLP was the ability to model language using distributions over sequences of characters instead of strictly over words[34, 17]. Despite the fact that these models are not trained on the premises of utilising any word- or sentence-level boundaries, they are still capable of understanding the different syntactic and semantic properties. An additional benefit from the use of these models is their strong capacity in capturing the exact word meaning in the specific context.

All the above observations are leveraged by Akbik et *al.*[2] who introduced the Contextual String Embeddings (also known as 'flair'). These are produced by employing the internal states of a trained character-based language model. Its architecture is consisted of two individual language models with different directions ('Forward' and 'Backward'). During the extraction of embeddings for a particular word, the 'Forward' model propagates information from the beginning of the sentence up to the last character of the word whilst the 'Backward' model propagates information from the end of the sentence up to the first character of the aforementioned word. The extracted information from both directions forms the output hidden states which are eventually concatenated to construct the final embedding. This process is visualised on Figure 2.5.

To evaluate the performance and the effectiveness of the aforementioned embeddings, the authors incorporate them as features on models for three classic downstream sequence tagging tasks; namely English and German Named-Entity Recognition (NER), chunking and Part-of-Speech (PoS) tagging. They use the Bidirectional LSTM-CRF (BI-LSTM-CRF) configuration introduced by Huang et *al.*[14] as the underlying sequence labelling architecture. For all these tasks, the new embeddings manage to outperform the reported state-of-the-art solutions. An additional benefit from the use of flair is that, due to the utilisation of a reduced vocabulary, they are characterised by a significant improvement on the computational time when compared to other types of embeddings.

Another useful observation that results from the evaluation is that the concatenation of the contextual string embeddings with context-insensitive ones (e.g. GloVe[26]) leads to a significant increase on the average F1-Score for the different tasks. The authors justify this by the fact that the semantics extracted in the word-level possibly act as a supplement to the character-level semantics which are captured by their own embeddings.

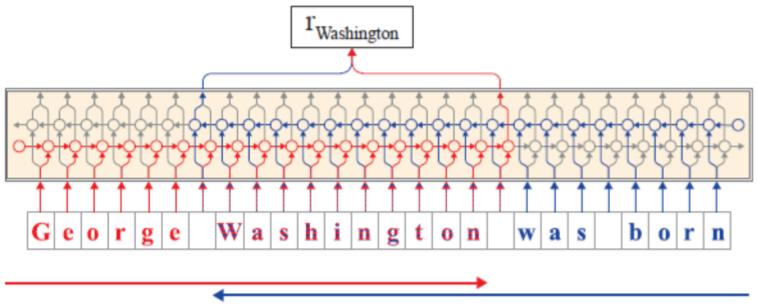


Figure 2.5: Extraction of Contextual String Embeddings (flair) for the word 'Washington' [2]

2.3 Natural Language Processing Techniques

2.3.1 Co-Reference Resolution

Co-reference resolution is defined as *the task of finding all expressions that refer to the same entity in the text*[1]. In normal speech, an entity usually appears on its full form just once; when it is introduced in the text for the first time. Most of the following references to this entity are made by means of co-references or anaphora. A direct consequence is that the vast majority of relations between entities in human communication are expressed in this form. The ubiquity of this phenomenon highlights why its resolution is considered an important step of preprocessing which leads to performance improvements for several information extraction tasks.

The neuralcoref module [2] of the Python-based spaCy NLP engine (whose syntactic parser is considered to be the fastest in the word with an accuracy of 0.9260)[3] provides an implementation for the co-reference resolution task by employing a scoring model based on a neural network as proposed by Clark and Manning[7]. In contrast with other older methods which involve training with heuristic loss functions, the authors make use of two different reinforcement learning approaches with the aim of optimising a neural mention-ranking model to be utilised for co-reference evaluation metrics. Their solution reports significant improvements over the previous state-of-the-art solutions on the English and Chinese sets of the *CoNLL 2012 Shared Task* dataset.

2.3.2 Dependency Parsing

Dependency Parsing is concerned with the analysis of sentences in terms of the 'attachments' between their different constituents. It can be depicted in the form of a directed graph over words. It represents the structure of a sentence by defining the relationships between 'head' tokens and their 'modifiers' (or 'dependents'). The edges connecting these two kinds of tokens denote the syntactic dependencies between them which form the overall dependency grammar[10]. Dependency grammars are widely used in the field of Text Mining for different tasks that involve the identification of relationships between words in speech such as co-reference and anaphora resolution.

2.4 Sentiment Analysis

2.4.1 Sentiment Polarity

Undoubtedly, the component of sentiment analysis that has attracted the greatest share of research interest so far is polarity. This can be explained by the fact

[1]The Stanford Natural Language Processing Group - Coreference Resolution - https://nlp.stanford.edu/projects/coref.shtml

[2]spaCy neuralcoref - https://spacy.io/universe/project/neuralcoref

[3]spaCy - Facts & Figures - https://spacy.io/usage/facts-figures

that it is indeed its most important constituent; an opinionated sentence may lack a specific target but it cannot be considered as subjective without some form of polarity towards either the positive or the negative end.

Some of the earlier works are based on the exploitation of syntactic word patterns and sentiment dictionaries to perform classification in an unsupervised manner. These approaches assume that the presence of words which inherently bear an amount of sentiment as well as the occurrence of certain syntactic patterns are usually dominating factors for the existence of an opinion or sentiment.

An example of the utilisation of syntactic patterns for detecting expressed sentiment is presented by Turney[36]. This technique uses PoS tags to construct different patterns which are employed for the extraction of all sentence slices that match them. Then, using *Pointwise Mutual Information* (PMI) as a metric, the Sentiment Orientation (SO) of the extracted phrases is calculated and eventually averaged. The polarity of this cumulative SO denotes the overall polarity of the inspected text extract. This approach is evaluated on reviews which originate from different domains with a resulting accuracy that ranges between 0.6600 and 0.8400.

On similar grounds but using sentiment dictionaries instead, Taboada et *al.*[35] compute a sentiment score for each document which is under assessment. For the calculation of this score, they look up the aforementioned dictionary to identify any sentiment words or phrases along with their orientations and strengths and incorporate them on the current score after considering possible intensifications or negations. Amongst the dictionaries that they experiment with is SentiWordNet[11] which is currently one of the most popular lexical resources for this form of keyword-based sentiment analysis in the literature.

However, a straightforward observation regarding the aforementioned approaches is that, because they focus on particular chunks from the sentences which according to pattern-based rules or dictionaries are indications of sentiment but they may not accurately reflect the sentiment for the entire sequence of interest. For instance, methods that involve averaging polarity scores may be susceptible to extreme values of sentiment which may incorrectly skew the overall mean value towards one direction. If, according to a sentiment dictionary, a phrase is generally associated with a large amount of sentiment of a certain polarity but, in this particular case, it appears in a mostly-neutral sentence where the overall context does not imply this intensity of sentiment, it may affect the final score in an undesirable way which will alter its estimated polarity.

As an immediate consequence of their evident success in other text classification tasks, Support Vector Machines (SVMs) algorithm is also a quite popular option for sentiment polarity detection. Pang et *al.* experiment with it along with two other traditional classification approaches (Naive Bayes and Maximum Entropy) to distinguish between positive and negative reviews in the Movie's domain. They make use of unigram/bigram features as well as PoS tags. Their best-performing system is the SVM-based Classifier which employs only unigram features with an average three-fold cross-validation accuracy of 0.8290.

Experimenting with the same classification approaches but with a focus on the Twitter domain, Go et *al.*[12] utilise distant supervision to avoid hand-labelling their training data. They propose the use of emoticons corresponding to positive or negative sentiments as query terms in order to extract large amount of tweets containing them for which these emoticons serve as noisy labels. Their experimental process shows that the training of the abovementioned three classification algorithms with their training data managed to reach an accuracy of over 0.8000.

Barbosa and Feng[3] present an approach for the automatic detection of sentiments on tweets that leverages their structure and the meta-information of their constituent tokens. For the former class of features, the authors analyse the syntax of the tweets and extract information regarding the their nature (i.e. whether they are plain tweets, retweets or replies to other tweets) and the presence of specific elements such as hashtags, links, punctuation, emoticons and upper-case letters in their body. The second set of features considers each word individually and associates it with its PoS tag, prior subjectivity (either weak or strong) and finally its prior polarity. Both prior subjectivity and polarity are obtained by looking up the subjectivity lexicon that is also used by Riloff and Wiebe[32]. The selection of these meta-information features is based on their effectiveness as demonstrated in previous researches[38, 33].

In order to avoid the manual annotation of tweets, they make use of external resources (i.e. dedicated sentiment detection websites for Twitter data) whose results combine and cross-validate using various strategies. Their method is comprised of two concrete subtasks: firstly they decide whether a tweet is subjective or not and, if the former is the case, they further classify the tweet as positive or negative. After experimentation with various learning algorithms, the one that yields the best results is SVM. Their system outperforms the baselines (one of them being the simple unigram model by Pang et *al.*[25] that was mentioned earlier) in both subtasks.

Agarwal et *al.*[1] employ data scraped from Twitter in order to build models capable of classifying tweets into positive, negative or neutral based on their sentiment. They split their problem into two different classification tasks; one for binary classification into positive or negative and one for multi-class classification into either positive, negative or neutral. Their dataset is consisted of 5127 tweets collected by archiving the real-time stream, manually annotated using a commercial resource and finally sampled in a stratified manner by the authors themselves to assert class balance. In addition to their models, they introduce two new resources aimed to support the process of preprocessing twitter data; namely an emoticon-to-emotional-state dictionary and an acronym resolution dictionary for translating frequently-used abbreviations in microblogging. In addition to these, they apply some extra preprocessing steps including the replacement of URLs, User Mentions (@User) and negations with fixed tags as well as the reduction of sequences of repeated characters in words into just three to reflect the emphasised use of the word.

Starting with a simple unigram model as the baseline, the authors experiment with a feature-based and a tree-kernel-based model as well as with combinations of the aforementioned configurations. For the feature-based model, not only do they employ features introduced in previous works but also define new ones. On the other hand, the tree-kernel-based model involves the representation of numerous feature categories with a more comprehensive tree structure. The concept of Partial Tree (PT) kernel is used to calculate the similarity between trees. All of their models yield a better performance than the compared baseline. However, their best-performing configuration with the most significant effect on the performance of the classifier is the one that combines the prior polarity of words with their PoS tags which achieves an accuracy of 0.7539.

2.4.2 Sentiment Target

On the other hand, sentiment target detection has started to receive more attention in the last years. Different entities who find themselves interested in the opinions of people on different matters appreciate the importance of being able to analyse an opinion instance automatically and deduce the entities toward which the sentiment is directed.

Mitchell et *al.*[24] conducted an elaborated study on the targeted sentiment in an open domain environment and present a novel approach based on the intuition that the problem can be abstracted as a sequence labelling task. Considering this, they employ a CRF-based architecture that leverages linguistically-informed features on the word level. These features are either surficial (e.g. message length), linguistic (e.g. whether it is a function word or not), brown clustering (as an alternative to PoS Tags in order to keep the approach language-independent) or sentimental (e.g. prior-polarity) and they are extracted for both the current word but, also, for a context-window of three words in either direction. Additionally, they break down the task into two smaller ones: 1) Denoting whether there is an entity to which sentiment is targeted, 2) identifying the polarity of the sentiment towards it/them. For the first task, they use the *BIO* encoding (*B* - For the beginning of a Named Entity, *I* - For the rest of the tokens that consist a Named Entity and *O* - For any token that does not belong to a Named Entity) whilst for the latter the values are either POS, NEG or NOT-TARG.

Regarding the model configurations, the authors experiment with three different strategies. In the first one ('PIPE'), where the model acts as a pipeline, the entities are firstly learned followed by the targeted sentiment towards them. The second model ('JOINT') learns the entities along with the sentiment. Finally, the third structure ('COLL') learns the entities and the sentiment together using combined labels. Each of these configurations holds certain assumptions regarding the possible couplings between the two aforementioned subtasks. The 'PIPE' model is based on the rationale that these are completely independent and therefore they cannot utilise any relationships between them. On the other hand, a more integrated approach such as the 'JOINT' model can handle and leverage any dependencies between the two smaller subtasks. It is quite plausible to assume that they are indeed not completely independent since the association of a token with some kind of sentiment is an indication that it is a target entity at the same time and conversely.

The authors evaluate the three strategies on both English and Spanish microblogging data. Their models managed to reach around 0.9000 accuracy for the second dataset. However, when it comes to the English dataset, the resulted gain against the baselines is not significant enough to be considered.

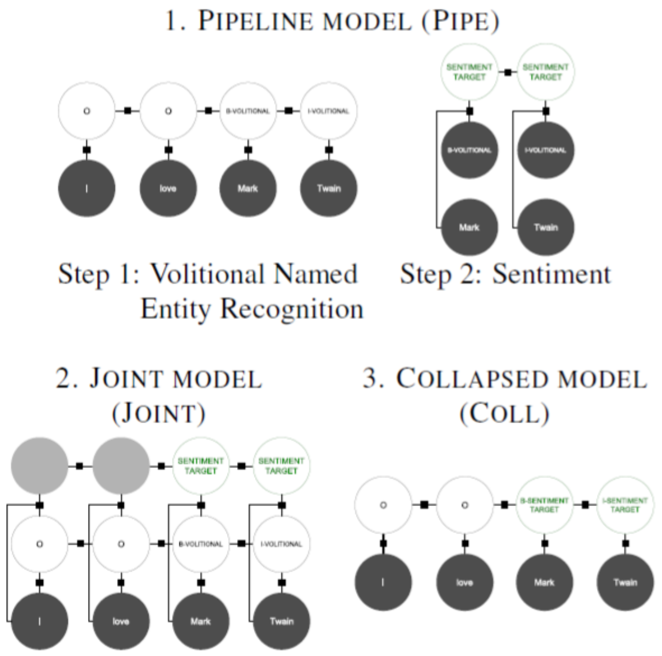


Figure 2.6: Visual demonstration of the way the three models ('PIPE', 'JOINT' and 'COLL') process a sample sentence [24]

Yang and Cardie[39] attempt to perform fine-grained opinion extraction by identifying the opinion entities (holder and target), the opinion expression as well as the relationships between them. Aiming to differentiate their approach from previous ones, they make use of a joint inference model instead of a pipeline. Their model is capable of capturing the internal dependencies between the subtasks that consist their problem. The features they extract reflect the local properties of the possible opinion expressions as well as their syntactic and semantic characteristics. The results of the experimental process show that this joint model is significantly more powerful than the traditional pipeline ones as well as other models that are focused solely on a single aspect of the opinion extraction process.

A quite recent study on target-based sentiment analysis is presented by Li et *al.*[20]. Their work aims to solve the complete sentiment analysis task by proposing a model which treats the problem in an end-to-end manner using a unified tagging scheme. The implemented system is composed of two stacked RNNs. It is worth noting that they make use of LSTMs to benefit from any underlying connections between the words in a sequence. The bottom layer attempts the auxiliary task of predicting the target's boundaries. Its output acts as a guideline for the upper layer which performs the primary task of estimating the appropriate labels for target-based sentiment analysis.

In addition to the above, the architecture is enriched by three custom-made components: Boundary Guidance (**BG**), Sentiment Consistency (**SC**) and Opinion-Enhanced (**OE**) Target Word Detection. **BG** utilises information about the target boundaries by modeling all the constraint transitions from target boundaries to target polarities explicitly, in the form of a transition matrix which is used for determining their proportions based on the underlying confidence of the target boundary tagger later on. **SC** makes use of a gate mechanism which combines the features extracted from the current word with its predecessor ones in the sequence so that it is less likely for tokens belonging to the same target to end up being allocated with different sentiments. Finally, **OE** attempts to improve the quality of the boundary prediction for the opinion targets by looking for words that indicate the expression of an opinion in a certain window around each token and subsequently training an elementary classifier which distinguishes between target and non-target words based on distant supervision.

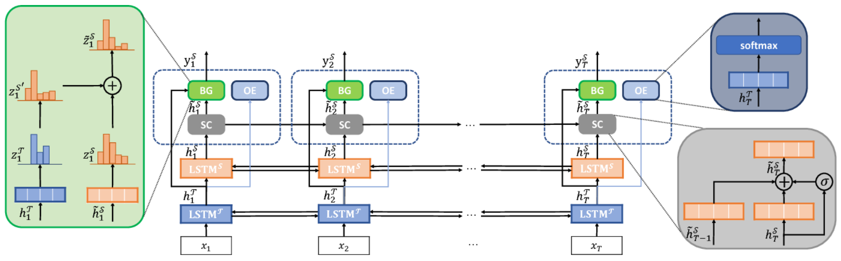


Figure 2.7: A diagram depicting the architecture proposed by Li et *al.*[20]

To evaluate the performance of their model, the contributors of this paper merge the training sets from the datasets provided in the three years of the SemEval ABSA challenge[28, 29, 30] which belong to the 'Laptop' and 'Restaurants' domain with the tweets collected and used by Mitchell et *al.*[24]. GLoVE embeddings[26] are extracted and fine-tuned during training to be employed as features. Finally, they compare their proposed architecture with several baselines introduced in previous researches and report that it is capable of achieving state-of-the-art results. Their full model yields F1-Scores of 0.5790, 0.6980 and 0.4801 over three datasets and exceeds the corresponding results from the previous best-performing configuration (i.e. 0.5619, 0.6638 and 0.4735, respectively).

Chapter 3

Research Questions

Originating from the task definition itself and supported by the different findings presented in Chapter 2, the following research questions are formulated with the aim of being addressed throughout this study:

- (*Main Goal*) Is it possible to provide a complete and novel solution to the sentiment analysis task involving the detection of the *opinion holder*, the *polarity* and the *target*?
- Do the newly-introduced contextual embeddings (i.e. BERT and flair) which have already yielded some remarkable results on other NLP tasks provide any performance gains when used for sentiment analysis for the first time?
- How to cope with the lack of target-oriented annotated datasets?
- How to quantify and assess the briefly-explored issue of class imbalance in a sequence tagging dataset? Which approaches can be used for mitigating this problem?
- How can meaningful data augmentation take place in a sequence tagging context considering the different constraints involved? What will be the effects from the inclusion of augmented data in the training set for its class balance and the performance of the resulting model?

Chapter 4

Dataset

4.1 Existing Dataset Availability

Due to the abundance of previous research in sentiment analysis and polarity detection in particular, there is a large amount of datasets with either sentences or larger text extracts annotated based on their sentiment which are publicly available for experimentation. These datasets are either manually annotated by human assessors or they are subject of automatic annotation based on certain heuristics. Depending on the purpose behind their collection (for example, some of them were compiled as part of related studies), the annotated polarity could either take labels corresponding to the two extremes (i.e. positive or negative) or, additionally, consider the neutral case.

As mentioned in the Introduction, one popular domain for the application of sentiment analysis is Twitter. Thus, a lot of datasets consisted of tweets have been made available in the past few years. Sentiment140[1] dataset contains 1.6 million tweets that were automatically annotated as it was explained in Subsection 2.4.1. This dataset was collected by Go et *al.*[12] for a study that is considered to be one of the initial works related to the application of sentiment classification on Twitter data.

Apart from Twitter datasets related to various topics, there are also collections of tweets that originate from a specific domain. An example is the *Airline Twitter sentiment*[2]. It includes scraped tweets from February 2015 in which customers praise or complain about their experience with some of the major U.S. airlines.

For the purposes of their studies on sentiment classification[4], a research team from Johns Hopkins University's Department of Computer Science has collected Amazon Reviews from several product domains (four in the first release[3] and twenty-five in the second one[4]). This dataset was named *Multi-Domain*

[1]Sentiment140 Dataset - http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip
[2]Airline Twitter Sentiment - https://data.world/crowdflower/airline-twitter-sentiment
[3]Multi-Domain Sentiment Dataset - http://www.cs.jhu.edu/ mdredze/datasets/sentiment/index2.html
[4]Multi-Domain Sentiment Dataset (Version 2.0) - http://www.cs.jhu.edu/ mdredze/datasets/sentiment/

Sentiment Dataset (thus, referred to as *Multi-Domain* dataset). The reviews are accompanied by their star rating (from one up to five stars). Reviews with less than three stars are labelled as 'Negative' whilst the ones that contain with more than three stars are marked as 'Positive'.

The only publicly-available[5] dataset that includes target-oriented annotations is the 'Restaurant' domain of the *SemEval-2016 Task 5: "Aspect-Based Sentiment Analysis"* dataset [28, 29, 30] (thus, referred to as *SemEval-2016-Task-5*). These target-based annotations made the specific dataset an ideal choice for the purposes of this research. Unfortunately, it was not possible to solely rely on it, due to a variety of reasons. Firstly, the dataset is consisted of sentences that come only from the restaurant domain. Therefore, they contain words from a vocabulary related to this specific context. If they were used to train a model, they would most certainly affect its ability to generalise over opinionated sentences from different domains. Secondly, due to its relatively small size, it could not be used for training Deep Learning models effectively.

Additionally, numerous attempts were made to be granted access to datasets used by other target-oriented studies. The purpose of this was to both use them for training and testing the proposed models but to also validate the performance of previously-implemented systems. Unfortunately, this effort was unsuccessful.

4.2 Constructing a New Dataset

Despite the fact that the aforementioned datasets were not suitable as standalone solutions on their own, their union could prove to be useful for training an effective model. In particular, the last two datasets (*SemEval-2016-Task-5* and *Multidomain sentiment analysis*) are pretty similar in the sense that they both originate from reviews, but from different domains, which means that they could introduce some variety to the models during training. Apart from this, reviews usually abide more to the rules of written speech. Unlike any social media postings which are written in a more quick and informal way, reviews tend to resemble the formal way of writing to a higher degree due to the fact that their authors aim to associate their writing with a touch of credibility so that they can be taken seriously by the readers. At the same time, though, they don't lack any sentiment as a more formal piece of writing would probably do (e.g. company documents, e-mails). This provides an ideal balance between the objective/factual and the subjective/sentimental forms of speech. Therefore, based on the above factors, a new dataset (thus, referred to as *ZyLAB-Targeted-Sentiment-Reviews* or *ZyLABSent*) was compiled which consists of subsets from these two existing datasets.

It was important to ensure that there is an equal amount of samples from different domains, so that the models would not be biased towards any of them. To accomplish this, one thousand sentences were randomly chosen from both the training and test sets of *SemEval-2016-Task-5* dataset. Furthermore, for the *Multi - Domain* dataset, the counts of reviews for all the available domains

[5]SemEval-2016 Task 5 Dataset - http://alt.qcri.org/semeval2016/task5/

were calculated and the four of them with the largest number of samples were selected. Every particular domain was further divided into positive and negative groups of sentences. Out of each of these subdivisions, five hundred sentences were sampled to additionally enforce a relative balance on the polarity.

4.3 Annotation

Unfortunately, the *Multi-Domain* dataset is characterised by two important limitations. The first one had to do with the scope of sentiment annotations. These are provided in a per-review and not in a per-sentence manner. However, when considering each sentence individually, it is highly possible to come across sentences whose polarity disagree with the overall review polarity. For example, in a negative review, there can be a sentence which expresses a positive opinion, in contrast with the rest. This may become even more acute with the use of cynical language. Thus, it is not safe to deduce that the polarity of each sentence matches the global polarity of the review. The second issue is analogous to the former. The dataset states the product for which a review is about but the same does not hold for every individual sentence. It is plausible that a sentence may include an opinion related to a different entity than the rest of the review.

In order to overcome these restrictions, an annotation process was initiated on the set of sentences extracted from the aforementioned dataset. The sentences were shuffled and divided into eighty separate annotation tasks. For the given sentences, the annotators were asked to detect the entities that appear to be targets of sentiment; either positive or negative. Additionally, they were informed on the polarity of the review from which a sentence originated from, as an indication of the overall context. It was stressed that the "entities to-be-detected" should appear as tokens in the sentence. For example, if a sentence was associated with a target entity of a certain polarity that could be inferred from the context but did appear *explicitly* in the sentence, the corresponding section should have been marked by '[Unknown]' token. Also, in case a sentence did not include an element of certain polarity, this section should have been filled in with the '[None]' token.

Each task was scheduled to be completed three times so that the responses could be cross-checked and validated before the final annotations were generated. Several interpretation conventions were established during the process to provide as much consistency between the annotators' responses as possible.

As soon as the annotation phase was completed, the two polarity components of each sentence would be collected and analysed. By taking the nature of the task into consideration, one would expect three forms of disagreement between the responses of the different annotators. Firstly, in the case when an entity consisted of multiple tokens (e.g. 'United States of America'), one of the annotators may have chosen to enter all the tokens comprising this entity in his/her response whilst another may have chosen to simply use a shortened version (e.g. 'States' or 'United States'). Secondly, disagreements may arise from typos or mistakes when entering the response to the system. Despite the fact that the annotators were encouraged to copy and paste the segment of the sentence that corresponded to an entity, it was still possible that mistakes were made. Lastly, the annotators may indeed disagree on which the target entity is and, thus, enter completely different responses.

A simple clustering approach was employed to process the different responses and decide upon the final annotations. Due to the fact that the annotations originate from a limited dictionary (i.e. the words of each particular sentence), responses from different annotators for the individual components could be grouped together based on their similarity. Firstly, as an initial means of clustering, co-reference resolution was applied on the input sentence to detect groups of expressions that appeared to correspond to the same entity. If the different annotators' responses found themselves in the same group, they were replaced by the entity to which they were resolved.

Subsequently, the responses were represented as BoW consisting of their TF-IDF scores and clustered based on their similarity using distances in the vector space. Non-identical responses sharing common tokens or almost identical tokens would end up together in the same cluster. The resulting groups would correspond to the responses that really differ from each other. Therefore, up to this point, the first two types of disagreement were mitigated. An important next step was to choose which member of each cluster would be chosen to represent the rest during the final decision process. Firstly, by iterating over all the members of each cluster, the ones which did not appear in the sentence in their exact form (i.e. typos) were disqualified immediately. Then, the entity with the largest amount of tokens from the remaining options was selected as the cluster identifier.

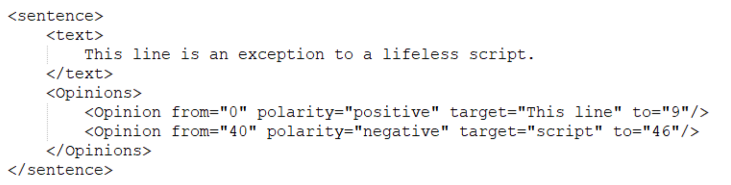
The last type of disagreement is purely related to how the annotators interpreted the sentences and perceived their target entities. A majority voting scheme was used between all cluster identifiers to decide which of them would be labelled as targets in the final version of the annotated set. Clusters whose size was equal to or greater than two (i.e. entities that were detected by the majority of annotators) were considered as the final target entities for each sentiment. In case no clusters had more than one members, n cluster representatives were randomly drawn from each polarity set as the target entities. n was defined as the mode of the count of detected entities for this specific polarity between the three annotators. If the statistic mode could not be calculated (i.e. all annotators detected a different number of entities) a single cluster would be chosen instead. This action would have the exact same effect as the case of a single annotator; full faith would be put on the responses of a single (artificial) assessor.

Unfortunately, due to time constraints, it was not possible to use a cross-validated version of the annotations that would result from the use of responses from three different people. Instead, for the purposes of the conducted experiments, the responses submitted by the first annotator were solely used. However, a future plan is to make the target-oriented version of this dataset with the cross-checked annotations publicly available.

4.4 Structure and Format

After the annotation extraction was finalised, all bits and pieces had to be put together to form the new dataset. To maintain some consistency between the structure of the *SemEval-2016-Task-5* and the subset of the *Multi - Domain* dataset chosen to be included in *ZyLABSent* so that they could be simply concatenated into a single file, the same XML tree-based format was employed. This specific encoding allows the hierarchical parsing of documents. This is particularly useful in this case since the dataset can be thought of as a tree-like structure whose traversal would start off from the 'root' (i.e. the set of all sentences), progress through each 'internal node' (i.e. sentence) and reach the individual 'leaves' (i.e. opinions exported from each sentence).

To facilitate this, each sentence was stored into a separate 'sentence' element. This element could include one or more 'opinion' child elements which corresponded to the expression of a positive or a negative sentiment. These 'opinion' instances included information about the particular target entities that were determined during the annotation process. If an entity of a specific Polarity was not '[None]' or '[Unknown]', the entry specified the starting ('from') and ending ('to') indices of the occurrence of this entity in the sentence string to make its identification easier. It is worth noting that a special convention was followed for this: in case the target entity appeared more than once in the sentence, the indices of its first occurrence were used. If the target is marked as either '[None]' or '[Unknown]', both the 'from' and 'to' properties of the 'Opinion' element were assigned a value of zero to indicate the absence of a target entity token with this polarity in the sentence. An extract from the XML-formatted dataset file showing a sentence and its targets for both polarities is shown on Figure 4.1.



```
<sentence>
    <text>
        This line is an exception to a lifeless script.
    </text>
    <Opinions>
        <Opinion from="0" polarity="positive" target="This line" to="9"/>
        <Opinion from="40" polarity="negative" target="script" to="46"/>
    </Opinions>
</sentence>
```

Figure 4.1: A sample annotated sentence from the newly-constructed dataset

Chapter 5

Implementation

5.1 Proposed Approach

According to Mitchell et *al.*[24], an occurrence of sentiment can be broken down into three concrete components: the person who expressed it ('Experiencer' or 'Opinion Holder'), its polarity (or 'Attitude') and the entity to which it is directed ('Target'). In accordance with this, a system that aims to solve the complete sentiment analysis task should be able to detect any opinions appearing in a given sentence and extract these three components out of them.

The proposed approach aims to generalise the definition of Target-Based Sentiment Analysis (TBSA) task as it was formulated by Li et *al.*[20] so that it incorporates all the elements of sentiment analysis:

> "...a sequence labelling problem which employs a tagging scheme consisted of three components: Y^S = {<Opinion Holder>, <Polarity>, <Target>}. The first one denotes whether the current token is part of an 'Opinion Holder' entity (NH - No, H - Yes), the second highlights the polarity of sentiment (NEU - if it is not part of a target entity/no sentiment is expressed towards it, POS - Positive and NEG - Negative) and the last one to indicate whether it belongs to a target entity (NT - No, T - Yes). For a given input sequence X = $\{x_1, ..., x_T\}$ with length T, the goal is to predict a tag sequence $Y^S = \{y_1^S, ..., y_T^S\}$, where $y_i^S \in Y^S$."

This thesis attempted to approach the overall task by dividing it into three smaller subtasks; each corresponding to one of the sentiment components described earlier. The last two of them ('Sentiment Polarity' and 'Target') were tackled using supervised means. For the first one, a simple, rule-based approach was utilised instead.

5.2 Overall Architecture

The overall system architecture can be thought of as a pipeline consisted of five concrete layers (Figure 5.1). Starting off with the Preprocessing layer, the sentence was converted into a token-based representation before it entered the next layer where features were extracted out of these tokens. Depending on the model chosen to be used for predicting the 'Sentiment Polarity' (thus referred as 'Polarity') and 'Target' components of the problem (CRF- or LSTM-based ones), either word/contextual embeddings or lexicographical/syntactical features were collected.

As soon as the feature matrix for the sentence was constructed, it was passed through the third ('Prediction') layer. The desired model was used to predict the two aforementioned components which along with the original sentence eventually reached the penultimate layer of 'Opinion Holder Extraction'. There, the last element of the sentiment analysis task was detected. Finally, when all the constituents of the 'Sentiment Triple' were accumulated, it was visually represented in the form of a network graph on the last ('Visualisation') layer of the system.

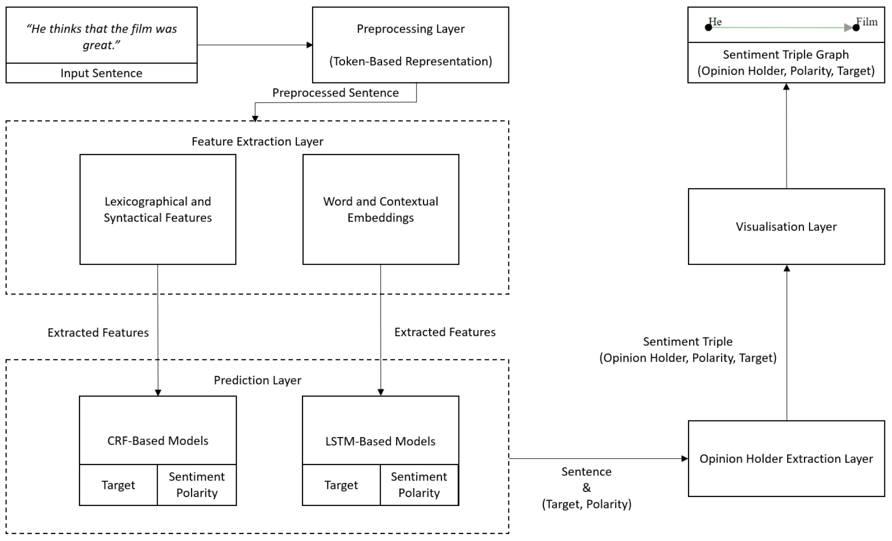

"He thinks that the film was great."
Input Sentence
Preprocessing Layer
(Token-Based Representation)
Preprocessed Sentence
Feature Extraction Layer
Lexicographical and Syntactical Features
Word and Contextual Embeddings
Extracted Features
Extracted Features
Prediction Layer
CRF-Based Models
Target
Sentiment Polarity
LSTM-Based Models
Target
Sentiment Polarity
Sentence & (Target, Polarity)
Opinion Holder Extraction Layer
Sentiment Triple (Opinion Holder, Polarity, Target)
Visualisation Layer
He
Film
Sentiment Triple Graph (Opinion Holder, Polarity, Target)

Figure 5.1: The overall system architecture

5.3 Preprocessing

5.3.1 Sentence Representation

Despite the fact that a sequence (or sentence, in this case) tagging task has the sentences themselves as its starting point, the real focus is on the individual tokens. Thus usually token-based approaches are used when it comes to the representation of the input sentences in a way so that they can support the capturing of information regarding the tokens themselves as well as their relationship with the surrounding context in the sentence.

Every sentence in this dataset was represented as a list of tuples; each tuple corresponding to a word. These tuples included the words themselves as well as two labels that determined the classes associated with them during the annotation phase. The first one was the 'Target' tag denoting whether this was a target token which could take the values 'T' or 'NT' corresponding to 'Target' and 'Not-Target, respectively. The latter could be either 'POS', 'NEG' or 'NEU' depending on the polarity of the sentiment. These two labels were intended to be used for the supervised components of the Sentiment Analysis task.

So to complement the definition from an earlier subsection:

> ...If x originates from a corpus c consisted of m sentences and it is indexed as its i^{th} element, the representation of this sentence for the supervised components of Sentiment Analysis can defined as a list of tuples of the form $c_i = x = [(c_{0,0},\ z_{0,1},\ z_{0,2}),\ (c_{1,0},\ z_{1,1},\ z_{1,2}),$ ..., $(c_{n-1,0},\ z_{n-1,1},\ z_{n-1,2}]$ where z is a set of tuples containing the 'Target' and 'Polarity' tags for each token in the sentence. The overall corpus can be represented as a list of these lists: c = $[x_i$, ..., $x_{m-1}]$.

5.3.2 Handling Class Imbalance

Approaching the task as a sequence tagging problem and representing the input accordingly resulted to certain interesting effects related to the distribution of the different class labels. Originating from the specifics of the problem domain itself, the fact that the focus of the different subtasks is only on certain sentence tokens (i.e. words that represent the target entities) means that the vast majority of tokens are expected to be classified as 'Non-Token' or 'Neutral'. A serious consequence of this, from a machine-learning point-of-view, is class imbalance.

The usual approach when it comes to the selection of dataset instances for training or testing is random sampling. The dataset is shuffled and instances of it are randomly drawn and added to either the training or the testing subset. The amount of samples to be chosen for each split is determined based on the Training-to-Test ratio decided by the machine learning engineer beforehand and it is something along the lines of 4:1 or 9:1. Sometimes, a third subset (validation set) is constructed which, often, has the same size as the test set to be employed for tuning the internal parameters of the chosen model.

The aforementioned sampling process when applied on a balanced dataset ensures that the samples from all classes are given equal opportunities of being involved in the training process. However, when the dataset is highly imbalanced, this same approach gives an unfair advantage to samples belonging to the majority class since its instances have larger probabilities to be selected during random sampling.

Several methods are widely-used to handle class imbalance in machine learning. Some of them are directly applied on the dataset and attempt to balance its class distribution or, simply, make the probabilities of being sampled even for the all classes; irrespective of their size. The first one is the application of undersampling on the majority class. This approach suggests the elimination of its numerical superiority by either removing some of its instances from the dataset or reducing the probability of their selection using some strategy-based way of sampling. Looking at the issue from the opposite point-of-view, another technique is the use of oversampling on the instances of the minority class(es). This again can be performed randomly by replicating their instances in the dataset multiple times to provide an equal amount of probabilities to all classes or by following methods that involve the use of synthetic data. A popular example of the latter is the SMOTE technique[6] which creates new instances of the minority class by interpolating between existing ones. A final approach which is not concerned directly with the dataset itself is the introduction of a weighting scheme on the loss function used during training. This scheme aims to boost the minority classes by assigning higher weights to their individual performance metrics while reducing the impact of the majority class by allocating a smaller weight to it.

However, before proceeding with following any approaches towards tackling class imbalance, it was important to assess its severity on this specific case and estimate any direct impacts it could have on the performance of the prediction processes. The first step was to ensure whether the dataset was indeed imbalanced and to which degree whilst the latter could confirm whether the possible class imbalance would significantly hinder the classification procedures and determine whether it was worth investing time handling it or not.

Since, by the time this research was conducted, there was no standard way to measure class imbalance for sequence tagging tasks in the literature, a new metric was defined aiming to quantify this phenomenon:

> Let s be a sentence consisted of n tokens t originating from a text corpus c. c contains m sentences. The index of each sentence in the corpus is denoted by i. Each of this sentence tokens is associated with one out of l labels corresponding to the possible output classes of a sequence tagging task. maj_s is the set of sentence tokens which are associated with a label belonging to the majority class and min_s is the union of tokens allocated with labels originating from the other $l-1$ classes that together consist the minority classes. The **Minority-to-Majority Ratio (M2MR)** measure of a sentence ($M2MR_s$) is defined as the ratio of the size of the former versus the size of the latter (Formula 5.1). To construct a global metric that quantifies the overall balance of a text corpus, the $M2MR_c$ is, also, defined which contains an additional multiplicative parameter α corresponding to the inverse of m (Formula 5.2).

$$M2MR_s = \begin{cases} \frac{|min_s|}{|maj_s|}, & maj_s >= min_s > 0 \\ 0, & otherwise \end{cases} \tag{5.1}$$

$$M2MR_c = \alpha \times \sum_{i=0}^{m-1} M2MR_{c_i} = \frac{1}{m} \times \sum_{i=0}^{m-1} M2MR_{c_i} \tag{5.2}$$

By definition, the size of the majority class should be greater than or equal to the size of the union set of the minority classes. This case corresponds to the scenario of a balanced sentence that would yield an $M2MR_s$ of one. In the case when min_s is equal to zero, the $M2MR_s$ becomes zero immediately due to the fact that the complete absence of minority class tokens from a sentence corresponds to a state of full imbalance. If maj_s is zero, it means that min_s is also zero which is only possible when an empty sentence with no tokens is concerned. In this scenario the $M2MR_s$ becomes zero, too.

According to the corresponding formula, as the value of the $M2MR_s$ approximates one, the more balanced the sentence under investigation is (i.e. the difference between the majority and minority classes is minimised). On the contrary, as it tends to become zero, the more imbalanced it is. Similarly, the use of the α parameter in the corpus-wide version guarantees that a perfectly-balanced dataset (i.e. a dataset consisted solely of balanced sentences) would receive an $M2MR_c$ score of one which is the maximum value this formula can take. Therefore, the optimisation objective when trying to construct a balanced dataset can be thought of as the attempt to maximise the score of $M2MR_s$ for each individual sentence or the $M2MR_c$ for the entire dataset.

The $M2MR_c$ score for the *ZyLABSent* dataset is **0.1536**. Since this value lies very close to 0, it can be deduced that the dataset is highly imbalanced. In the following divisions of this subsection, a non-generative approach aiming to solve the issue of class imbalance that often affects sequence tagging tasks will be introduced and explained. The newly compiled and annotated dataset will be used as the base for the corresponding experiments. Before that, though, a walk-through on the construction of an ideally-balanced dataset will be provided which is going to be used as the baseline for evaluating the performance of the proposed approaches.

Ideally-Balanced Dataset Construction ('Ideally-Balanced')

As it was stressed earlier, based on the definition of $M2MR$, if one aims to build a balanced dataset, he/she has to attempt to maximise the value that this metric can take.

A sequence tagging dataset is consisted of several sentences containing a varying amount of tokens (i.e. words) each. These tokens, being annotated for a particular purpose, are highly likely to be associated with a class that significantly outnumbers the rest. However, using this $M2MR$ metric, it is possible to select the slice of this sentence which offers the most balanced configuration using the available tokens. Attempting to balance each sentence individually can help mitigating the overall class imbalance which characterises the entire dataset.

In order to find the optimal slice for each sentence, all possible ordered slices with sizes ranging from two up to the length of the entire sentence were generated. Slices of size one were not considered due to the fact that such slice is basically a word and thus it no longer corresponds to a proper sequence. Subsequently, the $M2MR_s$ value of each slice was calculated and, after sorting the slices in descending order of it, a ranking was constructed. To provide an additional boost to slices which were both balanced but, also, retained the largest amount of original tokens possible, slices were grouped based on their $M2MR_s$ score and internally re-sorted based on their length. The slice that made it to the top of this refined ranking bore the maximum $M2MR_s$ of all and it was chosen to represent the sentence in the corpus. This process was repeated for all the sentences resulting to a new balanced version of the original dataset.

However, this approach could be associated with an important limitation. Since, it aimed to arrange the class distribution in the most optimal way, it could demonstrate a tendency towards building a dataset which was ideally balanced but not realistic anymore. For example, if most of the sentences originally consisted of only a single token allocated with a minority class label then a direct consequence of this would be that most of the sentences would end up being replaced by a slice of size two. Thus, if this dataset was used for any training purposes, it might result to a biased model which would demonstrate an aptitude towards labelling unseen sentences of this size but not when applied on sentences of different lengths.

Majority Class Token Reduction ('Majority Reduction')

Unfortunately, the nature of sequence tagging and the corresponding form of representation which it employs, poses some difficulties on the use of the general approaches in handling class imbalance. The dominating class in this scenario usually appears in the vast majority (even the entirety) of sentences in the dataset. Thus, if random undersampling was applied and whole sentences containing instances of it were removed then there would also be a decrease in the amount of instances belonging to the minority classes. This would be counter-effective for the final goal of this solution.

Therefore, a different methodology was followed to mitigate this problem. As a first step, all sentences that did not include any targets (either they were not opinionated or the targets did not appear explicitly as tokens in the sentence) were removed completely. After iterating over each sentence in the training set, sequences of consecutive tokens belonging to the majority class were reduced to a maximum length of two and thus ensured that the majority class tokens which were immediate neighbours to any minority ones were preserved. This idea was based on the fact that that these tokens were more useful when they appeared in close proximity to tokens from the minority classes, since they may provide some information on the surrounding context and act as indications that their nearby tokens are indeed targets of sentiment. However, the further away such tokens are located, the less useful or relevant may be to the target entities. Thus, by following this approach, the amount of majority class samples was reduced while as many of their intrinsic contextual relationships with the tokens of interest as possible were preserved.

5.3.3 Data Augmentation

In addition to the above approach towards handling class imbalance, two more methods were devised for tackling this task: 'Proportion-Based Probabilistic Sampling' and 'Synonym-Based Minority Class Sample Augmentation'. Both approaches were based on the idea of oversampling; adapted to accommodate the different assumptions which characterise sequence tagging.

There are two major things differentiating these two approaches from the two presented earlier. Firstly, both of them have an augmentative (i.e. generative) nature, meaning that part of their process is the creation and incorporation of new samples in the dataset. Secondly, these approaches are not solely targeted for tackling class imbalance; they can also be used for simply enriching a dataset with more data. The need for more data always persists in machine learning and, especially, in the NLP domain where there is a lack of established methodologies for doing so. The techniques to be presented aim to address this limitation. It is important to note that both of them can be adapted easily for use in any other problem domain which involves token-based representation of data.

Proportion-Based Probabilistic Sampling ('Proportion-Based')

A more generalised version of the approach followed to construct an ideally-balanced dataset could prove itself capable to mitigate the issues resulting from the lack of the realistic element in the resulting dataset. Instead of constantly favouring the slices with the highest $M2MR_s$, these slices were simply associated with more chances to be drawn whilst the selection was kept completely random.

The steps up to and including the ranking process that were described in the previous balancing technique were followed for this approach. Then, the ranking was split into four leagues based on the sorting order and the members of each league were replicated in the dataset based on them. More specifically, samples belonging to the top league were copied four times, tripled in the second league, doubled in the third and remained unchanged in the last league.

Finally, a random slice (or number of slices in the general case) was chosen from the list. This approach still benefited the balanced slices by giving them more chances of being selected while introducing the probabilistic element which would lead to a more 'realistic' setting.

Synonym-Based Minority Class Sample Augmentation ('Minority Augmentation')

The second approach was based on the creation of new samples by means of data synthesis. To accomplish this, all sentences were iterated over and their tokens which were annotated as targets of sentiment (i.e. assigned labels from the minority class) were detected. Based on the idea that a synonym is, most of the times, capable of conveying the same meaning as the original word without altering the quality of the overall sentence, new copies of the sentences were fabricated by simply replacing the target tokens with their synonyms.

To be able to identify all the possible synonyms for the different target words, the WordNet lexical database for English [1] was utilised. The target tokens were looked up in the aforementioned resource for possible synonyms. However, there was some variety on the lemmas returned for some of the words. In the English Language, words such as 'work' can be used in either Verb or Noun form. In order to tackle this, extracted lemmas that were associated with a different PoS tag than the original target token were discarded. The remaining lemmas belonged to synonyms matching the syntactic role of the initial word. As an additional clearance measure, any synonyms which appeared in a different form than the original word (i.e. synonyms in plural while the word itself was in singular and the converse) were disposed. For additional variety, different forms could have also been used but that could have led to some grammatical inconsistencies with the rest of the tokens in the sentence. Subsequently, the selected synonyms were incorporated on the original sentences and allocated with the same labels as their predecessors.

5.4 Feature Extraction

5.4.1 Lexicographical and Syntactical Features

Following the example of Mitchell et *al.*[24], different lexicographical and syntactical features were used as input to the various CRF-based models. The selection of this kind of features aimed to investigate whether the different properties associated with the words themselves could be proved important for the classification of tokens as possible targets receiving a sentiment of a certain polarity.

The features were extracted in a per-token manner for both the current token itself but for the neighbouring tokens, too. The number of nearby tokens to be used was determined, by means of experimentation. Since CRF-based models are highly competent in detecting patterns in the transitions between tokens and their associated labels, they could possibly identify any features that are important for this process. A list of the different features that were extracted can be found below:

[1]WordNet - https://wordnet.princeton.edu/

- PoS tag
- Prior polarity score
- Sentence position
- Length
- Is it in upper-case form?
- Is the first letter capitalised?
- Is it a digit?
- Does it include punctuation?
- Is it the first sentence token?
- Is it the last sentence token?

5.4.2 Word Embeddings

As mentioned in the 'Related Work' chapter of this thesis, currently, the state-of-the-art solutions for NLP tasks (one of them being sentiment analysis) utilise dense representations of words and their relations in the vector space; namely word embeddings. Even in their simplest, context-independent form, Word Embeddings had always been a source of traction for application in different kinds of problems that require an efficient way of representing corpora of words and sentences.

The recent advances in the implementation of contextual word embeddings was proved to be of a particular importance for tasks in which the context surrounding the words really mattered. An example for this class of problems is, undoubtedly, sentiment analysis where words may be associated with a different meaning based on the context they appear in. Also, depending on the word arrangement, there can be sudden changes of meaning and sentiment in the same sentence.

Thus, a natural approach was to employ context-sensitive word embeddings as features in order to be able to capture intrinsic properties and internal relationships between the words that consist the dataset. The latest additions to these state-of-the-art contextual-sensitive representations were BERT and flair and, due to this very recentness of their introduction, they have not been employed for sentiment analysis tasks yet; especially in the token level.

For the extraction of these Embeddings from the sentences, the Python-based **flair** framework[2] that Zalando Research has quite recently outsourced was employed. This library enables the use of various pre-trained models for the different forms of embeddings it supports. For BERT embeddings, the selected model was trained on 104 languages using an 12-layer architecture with 768-hidden layers and 110 million parameters. For the flair embeddings, a stack of two models with different directions (forward and backward) trained on a mix of corpora (Web, Wikipedia, Subtitles, News) in six different languages (English, German, French, Italian, Dutch, Polish) was chosen. To verify and utilise the observation made by Akbik et *al.*[2] that was mentioned in Subsection 2.2.3 about the extra gains resulting from the use of context-independent word embeddings in addition to the contextual ones, experiments were additionally conducted with stacked configurations combining the aforementioned embeddings with GloVe[26].

[2]flair Framework - https://github.com/zalandoresearch/flair

5.5 Sentiment Triple Extraction

5.5.1 Target and Polarity Prediction

As mentioned earlier, target and polarity prediction subtasks were tackled together by means of supervised classification. The *ZyLABSent* dataset was used as the base corpus for training the different models. Aiming to leverage their various benefits and compare their performance, two different forms of architectures were tested: CRF- and LSTM-based ones.

CRF-Based Models

In order to investigate the effectiveness of the different models introduced by Mitchell et *al.*[24] on this dataset and use them as a reference point during the evaluation of more complex architectures, adaptations of these models were implemented using the **sklearn-crfsuite** Python library[3]. Different lexicographical and syntactical features were employed as input to these models; namely the ones introduced in Subsection 5.4.1.

For the first model ('INDEP')(Figure 5.2), two separately-trained CRFs were used. The first one was dedicated to the determination of polarity whilst the latter was dealing with the detection of target. However, it is worth noting that since the two models did not interfere with each other, the order did not really matter.

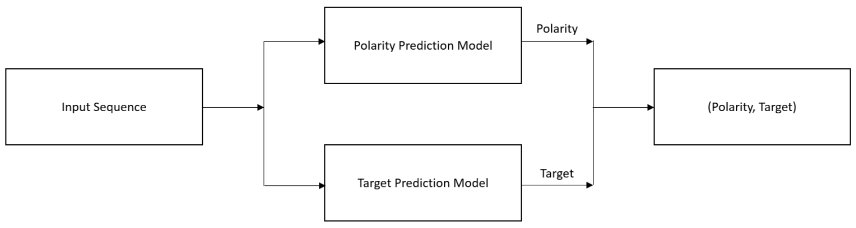


Figure 5.2: The architecture of the 'INDEP' model

The second model ('PIPE')(Figure 5.3) had a pipeline-like structure and leveraged the internal dependencies between the tasks of polarity and target prediction. Similarly with 'INDEP', two separate CRF models were implemented. However, they were not independent this time. The former was used to estimate the labels for the first task whilst the latter made use of these predicted labels along with the rest of the extracted features to determine the outputs for the second task.

Finally, the third model ('COLL')(Figure 5.4) tackled both tasks together using a single model. This model could generate predictions originating from a unified set of labels which included all the possible combinations of output labels from both tasks.

[3]sklearn-crfsuite - https://sklearn-crfsuite.readthedocs.io/en/latest/

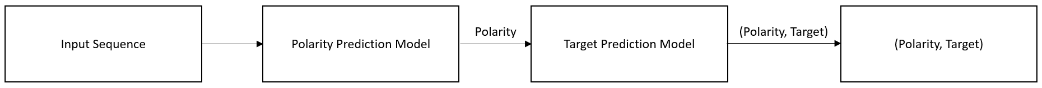


Figure 5.3: The architecture of the 'PIPE' model

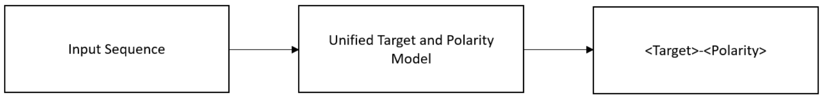


Figure 5.4: The architecture of the 'COLL' model

The rationale behind the use of these CRF-based approaches was their simplicity along with a demonstrated efficiency when it comes to the resources they require. This is highlighted by their fast training as well as their ability to extract meaningful transitional relationships between sequence tokens even when applied on a relatively small dataset.

LSTM-Based Models

According to the work by Huang et *al.* [14], the current-state-of-the-art architecture for sequence tagging tasks leverages Bidirectional LSTMs followed by a CRF layer. This model combines both the strong abilities of LSTMs to capture dependencies in the dimension of time but also the aptitude of CRFs to identify possible transitions in sequences. It follows that experimentation with this model configuration was considered to be a good starting point for the construction of predictive models for these two components of the sentiment analysis task.

Akbik et *al.*[2] who developed the **flair** framework mentioned in Subsection 5.4.2 have also enriched their library with the ability to perform sequence tagging tasks using this very architecture by employing different word embeddings as features. The library simply requires a text corpus to be used as input which is structured and formatted in a specific way and includes annotations for each token individually.

This provided the means and the ways for experimentation with the aforementioned architecture for the purposes of this thesis. The different class imbalance handling techniques, various combinations of word embeddings as well as some tweaks on the parameters of the models were applied and compared in order to determine the optimal configuration for the purposes of this task.

5.5.2 Opinion Holder Extraction

In contrast with the detection of the sentiment target and polarity, the identification of the opinion holder can be thought as a fairly simple process. Instead of following a similar supervised approach as with the other two sentiment constituents, a rule-based heuristic procedure was followed with the expectation of receiving results of high quality.

Purely from a high-level perspective, a sentence that involves the expression of an opinion without a direct reference to the entity from which it originates, is usually attributed to the author of the text itself. For example, in the sentence "This film was simply amazing", the entity that expresses this positive opinion is not mentioned explicitly anywhere in the sentence. Therefore, it is safe to infer that the opinion holder here is the author of this sentence. The author could be either a single person or multiple ones but, since there is no clear indication on their number, the generic '(Author)' token covers both scenarios.

On the other hand, there are sentences in which the opinion holders are explicitly referenced. These situations can be generally broken down into two categories: first-person thoughts or experiences and other-people thoughts or perceptions. An instance of the first class is a sentence in which an opinion or experience is described by the speakers themselves. The difference between them and the sentences described in the previous paragraph is that the identity of the speaker is evident by the appearance of first-person pronouns. Finally, the second category includes all sentences in which opinions or perceptions of other people are quoted by their author. Therefore, the opinion holder (or *'Subject'*) in this scenario are these third-party entities.

Based on the conceptual hierarchy of the three aforementioned scenarios for the identity of the opinion holders, a rule-based methodology was devised and applied to accomplish their efficient and precise extraction. Each sentence was initially allocated with '(Author)' as its opinion holder. Subsequently, it was tested against each of the other possible cases starting from the most general up to the most specific ones. Whenever a sentence satisfied an additional set of requirements, its allocated opinion holder changed to something more specific after discarding the previous more generic assignment. The result of this process was that the sentences eventually ended up allocated with the most specific opinion holder they could possibly be assigned based on the satisfied criteria. Even if a sentence failed to fulfill any of these more specific criteria, it remained assigned with the generic '(Author)' token which is always correct from a more general point-of-view.

When the opinion holder extraction process was initiated, each sentence was allocated with the '(Author)' token as its subject. Then, the sentence was iterated over in a per-token manner. To check whether the sentence conformed to the scenario of *"thoughts/experiences in the first-person form"*, each token whose PoS tag denoted that it was either a personal or a possessive pronoun (marked with **PRON** by the PoS tagger component of spaCy) was resolved into its subjective form to discover the underlying person. In the case when the pronoun was regarded to correspond to the first person (i.e. either 'I' or 'We'), the opinion holder changed to be this first-person personal pronoun. This category did not require the satisfaction of any additional conditions considering that usually when one talks from a first-person perspective, his/her tone fundamentally enriches the statements with an empirical or opinionated essence which makes him/her a highly plausible (or even the prime) candidate for being the opinion holder. Of course, this was subject to change, if there were any indications later on that this sentence could actually satisfy additional criteria that may lead to the detection of a different opinion holder.

As the iteration over the tokens progressed, an additional workflow was followed for all instances associated with a 'VERB' PoS tag. The stems of verbs appearing in the sentences were looked up in a lexicon of verbs denoting thought, opinion, or perception. If they indeed matched, it was also checked whether this verb was in active or passive voice. The rationale behind this extra requirement is that if these verbs appeared in Passive Voice form, it would mean that an opinion is expressed *about* this entity and not *by* this entity. If the verb was in passive voice, the opinion holder remained as it was, until any additional condition was satisfied. Otherwise, another mechanism was triggered for detecting the subject that was coupled with the thinking/perceiving verb. The reason why this distinction was made is that the occurrence of such words is associated with the explicit appearance of an opinion holder who is usually their subject when the syntactic structure of a sentence is concerned. Thus, resolving who this subject was could be considered as a key step. For example, in the sentence "I saw that he enjoyed this film.", it can be clearly observed that the entity expressing a positive opinion towards 'this film' was 'he' and not 'I' which is the first entity appearing in the sentence.

As soon as a verb of this type appeared, the words consisting the sentence up to that point were cut off from the rest and subsequently analysed by spaCy to detect its subject. spaCy is capable of resolving syntactic dependencies between words in sentences; one of them being the Subject-Verb relationship. In particular, spaCy marks instances of this dependency with the **'nsubj'** label. On the abovementioned example, 'he' would have been detected as the subject of the isolated subsentence. This can be seen on Figure 5.5 which shows a Syntactic Dependency Tree (SDT) for this sentence exactly as it was constructed and visualised by spaCy.

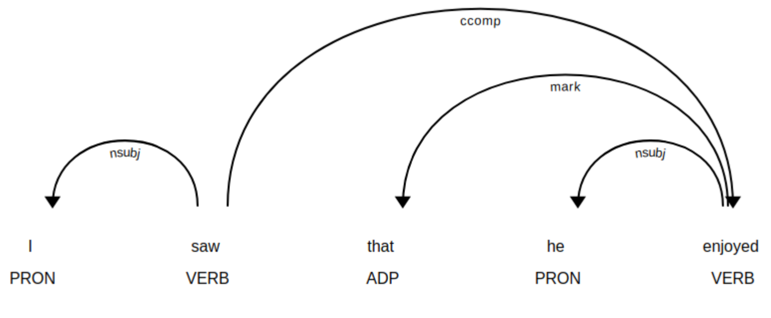


Figure 5.5: An example of how dependency parsing when applied on (a slice of) an opinionated sentence using spaCy can detect the subject by visualising the sentence in the form of a Syntactic Dependency Tree.

However, after empirical evaluation of this approach with a few sentences, an important limitation of it was discovered: it was not able to detect more complex subject configurations such as those consisted of multiple tokens or conjugations of entities. This can be depicted by demonstrating some the altered versions of the above sentence: "I saw that a group of children enjoyed this film." and "I saw that my brother and his girlfriend enjoyed this film.". For the former, the subject as it would have been denoted on the SDT (Figure 5.6) is simply 'group' which is correct but not specific enough. The full subject here consists of multiple tokens (i.e. 'a group of children') and so this special case will be, thus, referred to as 'Multi-Token Entity' scenario.

For the latter, the SDT (Figure 5.7) would have denoted that the 'brother' token as the subject which would be partly-correct and incomplete since the opinion was expressed by both him and his girlfriend. Due to the fact that the full subject to be extracted here is a conjunction of two entities that could have been considered as individual subjects but, in this case, they acted together as one, this scenario will be named as 'Conjugated Entities' case. Although this was somewhat expected by keeping in mind that a dependency parser considers the sentence in a per-token manner and thus cannot correctly interpret any relationships between words in higher levels, it was something that still needed to be tackled since it was tampering with the accurate extraction of the sentences' opinion holders.

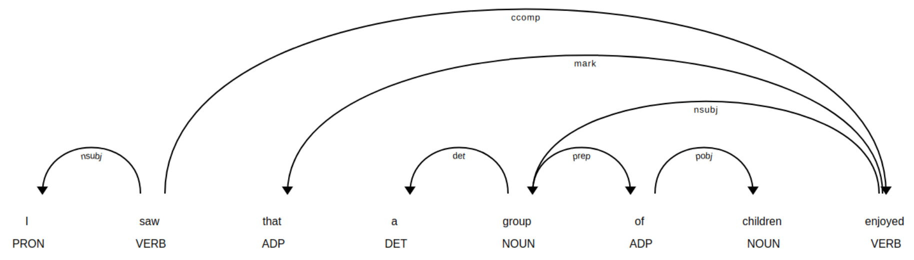


Figure 5.6: An example of a sentence slice that demonstrates the 'Multiple-Token Entity' scenario.

To tackle these issues, it was necessary to go one level higher in the syntactic hierarchy and identify the noun phrases included in the sentence slice of interest. These chunks of text contained both the noun which acted as the 'head' but also the words that described or specified it further. Then, the extracted noun phrases were cross-checked with the previously extracted subjects and combined in a union-like manner. If a subject was a substring of a noun phrase, this noun phrase replaced it in the list of possible subjects for the given thinking/perceiving verb. This process on its own was capable to serve as a solution to the 'Multiple-Token Entity' scenario. Furthermore, as a step towards solving the 'Conjugated Entities' problem as well, the rest of the identified noun phrases were added to the aforementioned list in order of appearance in the sentence. The purpose of doing so was to check later on whether a conjugated version of

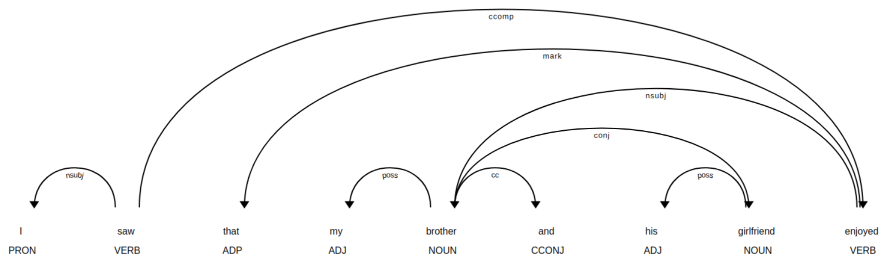


Figure 5.7: An example of a sentence slice that demonstrates the 'Conjugated Entities' scenario.

any elements of this list consisted the actual subject of the sentence.

Focusing on the 'Conjugated Entities' issue, all the possible combinations between the listed noun phrases and the subject token/noun phrase were generated and checked for appearance in the original sentence. If that was the case then the conjugation which satisfied this requirement was considered to be the opinion holder of this sentence. If not, the final opinion holder was declared to be the subject/noun phrase solely. As an additional verification checkpoint, the extracted opinion holder was compared against the predicted Sentiment Target and, if they were identical, the former was changed back to the original '(Author)' token. Finally, using spaCy neuralcoref, co-reference resolution was applied to detect possible co-referring clusters and if the opinion holder belonged to one, it was replaced by the noun entity representing it.

5.6 Visualisation

Apart of the actual process of extracting the different sentiment components in each sentence and in order to be able to get a better sense on how emotion was distributed around the entities of a text corpus, it was important to find a visualisation approach capable of providing this kind of insights to the viewer. The natural choice when it comes to the representation of the relationships between entities in the same context is a network graph. Networks have been successfully used to represent the interaction of entities on different levels with one of their most popular applications being the detection of entity communities.

However, the most crucial property of a Network representation is the way it presents the individual interaction between entities. As explained earlier, the problem of complete sentiment extraction on a sentence can be represented as a triple of the form ('Opinion Holder', 'Sentiment', 'Target'). Based on this idea, a straightforward way to represent the sentimental relationship between two entities in an opinionated sentence was in the form of a directed graph consisted of two adjacent nodes. The properties of their shared edge represented characteristics of the expressed emotion. Concretely, the direction of the edge showed the flow whilst its colour depicted the polarity (green for positive, red for negative). Therefore, the set of all these smaller graphs extracted from a textual corpus and formed a network structure were named *Sentiment Flow Network* (SFN). Each entity could have one or more emotional interactions with the other entities in the network which could be positive or negative.

To further elaborate on this representational approach, let's consider a sentence that contains an element of sentiment: "He thinks that the film was great." This sentence conveys the positive sentiment that an opinion holder (i.e. 'He') expresses towards a target entity (i.e. 'film'). Therefore, by following the aforementioned approach, this relationship will be visualised using two nodes ('He' and 'film') connected with each other using a directed edge (i.e. 'link') from the former to the latter. This is depicted on Figure 5.8.

Figure 5.8: A diagram that visualises a sentiment triple

This visualisation technique when applied on a corpus is capable of highlighting the sentimental dynamics between the different entities of a dataset. Nodes with the most outgoing edges represent entities which are the most active opinion holders. On the other hand, nodes with a large number of incoming edges correspond to entities which are quite popular opinion receivers (i.e. targets). More generally, nodes with a large number of incident edges (or in terms of graph theory, the vertices with the highest degrees) are the 'hotspots' of the network; meaning that the entities they represent appear frequently in opinion-bearing sentences.

It is also possible to use the aforementioned observations for drawing conclusions regarding the overall polarity flow within a dataset. By looking at either the incoming or the outgoing edges of a node, the corresponding entity may be thought of as either a strong source of positivity or negativity or equivalently as a popular target of a particular sentiment. However, investigating relationships of entities with neutral polarity may not be particularly interesting since the sentences from which they originate usually transmit information of more objective and factual nature.

From a broader point of view, a network-based representation like this can additionally provide some useful insights regarding the interaction between the different entities that constitute it. For example, looking at groups of entities which are characterised by high interconnectivity may lead to some very useful observations. Since they seem to sentimentally interact to a remarkable degree, it is highly possible that these entities are part of some form of a community (i.e. subgraph) which will be, from now, on referred to '*Sentiment Exchange Neighbourhood*' (SEN). From a graph theory point-of-view, this graph class is called a *Weakly Connected Graph* since if all of its edges are replaced by their undirected version, it is possible to define a path that connects any pair of its vertices. Subsequently, if among these entities, there is a particular node that stands out in terms of the number of incoming and outgoing links, it can be considered as the central point ('centre') of this community. It is the centroid around which the different entities in this subgraph seem to revolve and, if it was detached, most of the other entities would no longer seem to interact in a sentimental way.

All the above points are illustrated in Figure 5.9. Firstly, it can be easily deduced that the nodes appearing in the middle cluster of the figure are part of a SEN with strong sentimental activity. The '(Author)' is by far the most active opinion holder in this network as well as the node with the largest amount of links of both directions. Therefore, this particular node can be thought of as the center of this SEN. On the other hand, the most popular target of sentiment is the entity 'Apocalypse Now' which has two incoming links. The overall sentiment flow seems to be quite balanced in this SEN since there is an equal amount instances of sentiment of each polarity.

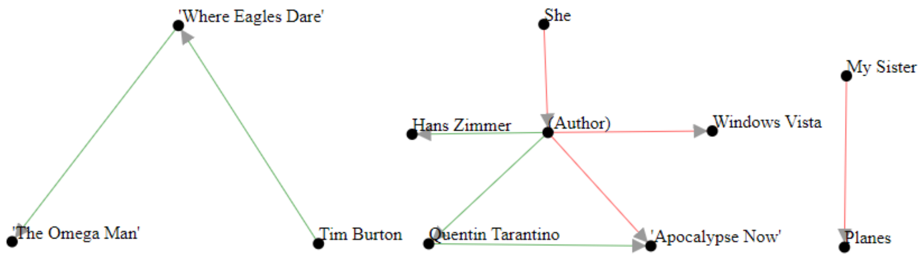


Figure 5.9: A diagram which visualises a given text corpus in the form of a *Sentiment Flow Network* where a *Sentiment Exchange Neighbourhood* is formed around the *(Author)* entity

Nevertheless, such a representation can be susceptible to several limitations that tend to become more acute as the size of the dataset increases. Firstly, a general issue associated with networked representations, the information overload, can make the visualisation too crowded and hardly readable for the viewer. Also, the same entities when appearing in multiple sentences may be interpreted as separate, unique instances. In addition to being a source of possible misinterpretations, this could be a worsening factor for the abovementioned overloading problem. So suggestions for future work to address these issues will be presented in Chapter 8.

The actual implementation of this visualisation technique was accompanied by some additional features to make its interpretation easier for the viewer. Firstly, the visualisations were fully interactive making possible to rearrange the nodes and edges in the provided space. This was, particularly, useful, especially if a large amount of nodes and edges were shown on the graph. Lastly, hovering over any nodes or edges presented some additional information for each of the corresponding instances. In the case of nodes, the number of incoming and outgoing links was presented whose sum corresponded to the degree of each vertex (See Figure 5.10). When the edges were concerned, the original sentence was projected to allow the viewer to understand the actual formation context of the particular subgraph (See Figure 5.11).

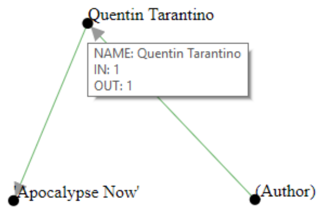
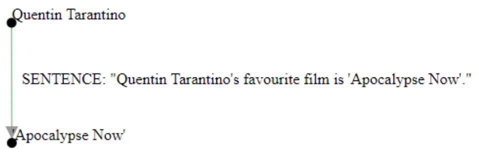


Figure 5.10: The textual description displayed after hovering the mouse over a node.



SENTENCE: "Quentin Tarantino's favourite film is 'Apocalypse Now'."

Figure 5.11: The opinionated sentence that associates the two entities which appears after hovering the mouse over the corresponding edge.

Chapter 6

Results

6.1 Target and Polarity Prediction

The first part of the experimental process was concerned with the evaluation of the two supervised components of the complete sentiment analysis problem: target and polarity detection. These two components were tackled together using the same approaches but with the aim of eventually determining the configuration that was optimal for each one of them individually. For the purposes of these experiments, 80% of the dataset was part of the training process and the remaining 20% was used for generating the results to be reported during testing.

6.2 Performance Evaluation

Due to the classification nature of these two subtasks, F1-Score was used as the evaluation metric. For the overall performance of each model, Micro-Averaged F1-Scores ($F1_\mu$) were calculated because, by definition, this measure takes class imbalance into consideration and reflects it accordingly on the results (Formula 6.2). Additionally, this metric was also used as the loss function during training for all the LSTM-based models.

Its Macro-Averaged ($F1_m$) equivalent does not involve the actual class sizes anywhere; it only sums up the individual F1-Scores and averages them over the total number of classes (Formula 6.1). This means that bad performance on a minority class would lower the value of the metric in the exact same proportion as the majority class would. Thus, a generally accurate model which underperforms in a single class would be overpenalised.

However, it is important to note that although the Micro-Averaged F1-Score considers the class distribution, it can completely overshadow the poor performance of smaller classes, if the majority class performs significantly well. Of course, this is unfair to the minority classes since their low individual scores would be masked by a high overall metric and thus they would end up overlooked during the optimisation of a model.

Therefore, a more fair, Weighted F1-Score was also considered which assigned weights to every class. These weights were inversely proportional to their frequency in the dataset. This means that the smaller the number of instances from a given class appearing in the dataset was, the larger the weight it was associated with it. The class with the largest number of tokens was considered as the 'majority' (maj) class whilst the union of the rest of the classes was the 'minority' (min). The exact weights were calculated using Formulae 6.3 and 6.4. This metric aimed to boost the impact of the F1-Score for the minority classes and award any configurations that yielded higher scores for these particular classes. The formulae used to calculate this Weighted F1-Score for either task can be seen on Equations 6.5 and 6.6, respectively and the chosen values for the different weights are listed on Table 6.1.

The two aforementioned metrics were calculated with different intentions. The Weighted score was used to compare the proposed configurations in each experimental iteration and choose the one which underlies a better performance for the minority class(es). The Micro-Averaged one was indicative for the overall performance of a model and how this compares to systems proposed in other studies.

$$F1_m = \frac{\sum_{i=0}^{l-1} F1_i}{|l|} \tag{6.1}$$

$$F1_\mu = \frac{\sum_{i=0}^{l-1} F1_i}{|all|} \tag{6.2}$$

$$W_{maj} = 1 - \frac{|maj|}{|all|} \tag{6.3}$$

$$W_{min} = 1 - \frac{|min|}{|all|} = 1 - \frac{|all| - |maj|}{|all|} \tag{6.4}$$

$$WF1_{TG} = F1_{NT} \times W_{maj} + F1_T \times W_{min} \tag{6.5}$$

$$WF1_{PL} = F1_{NEU} \times W_{maj} + (F1_{NEG} + F1_{POS}) \times W_{min} \tag{6.6}$$

where:

$F1_m$= Macro-Averaged F1-Score, $F1_\mu$= Micro-Averaged F1-Score
$WF1_{TG}$ = 'Target' Weighted F1-Score
$WF1_{PL}$ = 'Polarity' Weighted F1-Score
W_{maj} = majority class weight, W_{min} = minority class(es) weight
l: # classes, $|all|$ = total # tokens
$|maj|$ = # majority class tokens, $|min|$ = # minority class(es) tokens

Task	Class	Dataset Count	Weight
Target	**Non-Target (NT)**	9009	0.1082
	Target (T)	1093	0.8918
	Total	**10102**	
Polarity	**Negative (NEG)**	452	0.8926
	Neutral (NEU)	9017	0.1074
	Positive (POS)	633	0.8926
	Total	**10102**	

Table 6.1: The weights associated with each class based on their frequency in the dataset which were used for the calculation of the Weighted F1-Scores

As far as the different counts are concerned (Table 6.1, it is worth noting that there was some overlap between the classes of the two tasks. Specifically, with the exception of 8 neutral instances from *SemEval-2016-Task-5*, the 'T' class is consisted of the samples belonging to the 'NEG' and 'POS' classes whilst the 'NT' class contains the vast majority of the 'NEU' tokens.

6.2.1 CRF-Based Models

The first category of experiments attempted to adapt and reuse the different CRF-based architectures proposed by Mitchell et *al.*[24] (See Subsection 5.5.1) on the original imbalanced *ZyLABSent* dataset and see whether their observations can generalise enough to perform well for it, too. Each model was trained for a maximum of 100 epochs using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm with Elastic Net (L1 + L2) regularisation. Features were extracted from both the current word as well as from the neighbouring tokens in each direction that lied within a predefined distance (i.e. 'word scope') from it.

Model Comparison

The first experimental session aimed to determine which of the aforementioned model configurations was the most promising in terms of performance. During feature extraction, a word scope of 1 was used meaning that features were collected only from the current token and its immediate neighbours in either direction. The Average Weighted F1-Score of the two models were calculated to be used for comparison between the overall performance of the different architectures.

As the results on Tables 6.2 and 6.3 demonstrate, the observation made by Mitchell et *al.*[24] regarding the importance of the internal dependencies between the tasks of the target and polarity extraction was verified experimentally. The **PIPE** model which made use of the polarity predictions as features for target extraction managed to yield the best performance as reflected by its Average Weighted F1-Score which was significantly higher than the other

Model	**Micro-Averaged**	**Weighted**	**NT**	**T**
INDEP	0.9000	0.3369	0.9460	0.2630
PIPE	0.9990	0.9964	1.0000	0.8930

Model	**Micro-Averaged**	**Weighted**	**NEG**	**NEU**	**POS**
INDEP	0.8930	0.2622	0.0000	0.9450	0.1800
PIPE	0.8930	0.2622	0.0000	0.9450	0.1800

Model	**Micro-Averaged**	**Weighted**	**T-NEG**	**N-NEU**	**T-NEU**	**T-POS**
COLL	0.8920	0.4254	0.0000	0.9450	0.0000	0.1820

Table 6.2: The resulting F1-Scores from the experimentation with different CRF-based models

Model	**Average Micro-Averaged**	**Average Weighted**
INDEP	0.8965	0.2996
PIPE	0.9975	0.6293
COLL	0.8920	0.4254

Table 6.3: The average F1-Scores of the 'Target' and 'Polarity' components of the different CRF-based models

two architectures. As far as the rest are concerned, the **INDEP** model was the worst-performing configuration with an Average Weighted F1-Score that was less than half the winning **PIPE** system's. The third model (**COLL** lied somewhere in the middle in terms of performance.

Another important observation that could be made by looking at the results is related to the apparent inability of all models to correctly classify samples with negative instances of sentiment and target instances of neutral polarity for the 'COLL' model, specifically, which both resulted to F1-Scores of 0.000. This may be a side-effect from the fact that these classes were consisted of a very small number of samples and, due to the lack of data, the models might have not been able to be trained well enough to detect them appropriately. Alternatively, this result may denote that the strongest indicators for the classification of a sample into either of these classes might have been the features of tokens or even the tokens themselves which lied more than one word away from the token of interest. Subsequently, since those were not considered in this setting, instances belonging to these classes were not classified successfully.

Nearby Word Scope

However, looking solely at the performance of the 'Polarity' model, there was clearly room for improvement since it was underperfroming significantly when compared to its target prediction counterpart. Thus, the next step was to include more surrounding context to the training process. Up to this point, only features extracted from the immediate neighbouring tokens were incorporated to the feature vector of each word. For the following set of experiments, different neighbouring word scopes were used (from 2 up to 10 words in either direction) to determine whether they could lead to a possible increase to the model's performance.

Word Scope	Micro-Averaged	Weighted	NEG	NEU	POS
1	0.8930	0.2622	0.0000	0.9450	0.1800
2	0.8970	0.3552	0.0540	0.9470	0.2300
3	0.8960	0.3909	0.0750	0.9470	0.2490
4	0.8980	0.4123	0.0770	0.9470	0.2710
5	0.8960	0.4149	0.0910	0.9460	0.2600
6	0.8970	0.4320	0.0980	0.9470	0.2720
7	0.8930	0.4273	0.0830	0.9450	0.2820
8	0.8950	0.4122	0.0770	0.9460	0.2710
9	0.8940	0.4112	0.0960	0.9450	0.2510
10	0.8940	0.4514	0.0840	0.9450	0.3080

Table 6.4: The resulting F1-Scores from the experimentation with different word scopes for the 'Polarity' component of the 'PIPE' architecture

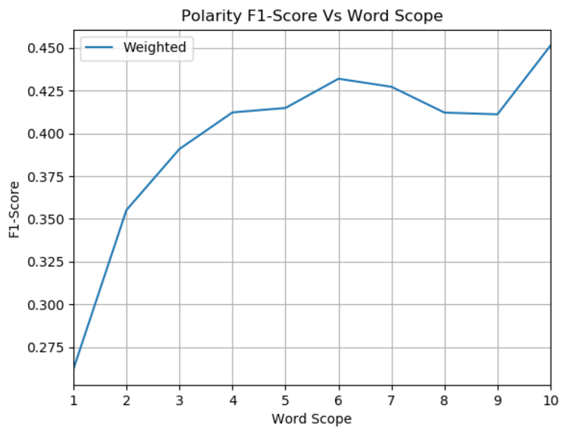


Figure 6.1: A plot that shows how the Weighted F1-Score of the 'Polarity' component of the 'PIPE' architecture was affected by the change of word scope

As both Table 6.4 and Figure 6.1 show, increasing the word scope indeed results to better performance. As far as the optimal value for it is concerned, there were two values of word scope for which there was a local maximum on the plotted line of the Weighted F1-Score: 6 and 10 (the last one was also the global maximum). It is worth noting that for each increasing step of word scope value, the dimensionality of every token's feature vector could be rise by at most 18 units at a time. This means that the largest the word scope value was the higher the dimensionality of the final feature matrix would be. A direct consequence of this was the notably larger training time and the slower convergence of the model. To find the sweet spot between high performance and train efficiency, the word scope of 6 was eventually selected to be part of the model configuration. It led to a Weighted F1-Score of **0.4273**; a value not not much lower than the **0.4514** produced by the best-performing model of word scope 10.

Useful Features

Using a built-in functionality offered by the **sklearn-crfsuite**, it was possible to determine which lexicographical or syntactical features were proved to be the most useful during the training process for the prediction of the different classes. This was achieved by sorting the different state features in descending order based on the weights they received. The ones associated with the largest positive weights were ranked at the top denoting high positive correlation between the appearance of a specific feature value and the prediction of a certain class.

Task	Class	Most Indicating State Feature	Weight	Second Most Indicating State Feature	Weight
Target	**Non-Token (NT)**	word.polarity(): NEU	6.5601	+6:word.istitle()	0.8060
	Target (T)	word.polarity(): NEG	5.2524	word.polarity(): POS	5.2136
Polarity	**Negative (NEG)**	-3:postag: NNPS	2.0747	+1:postag: SYM	1.8945
	Neutral (NEU)	postag: WP	3.0416	postag: EX	2.7424
	Positive (POS)	-3:postag: UH	1.8839	-3:postag: RP	1.6661

Table 6.5: The two most indicative state features for each class of the CRF model for 'Polarity' along with their associated weights. The signed index which precedes some of them denotes that they were extracted from a token positioned either before or after the token of interest in a distance equal to its numerical value.

For the 'Polarity' model, the state features ranked first were different values of either the current word's or its neighbours' PoS tags such as 'WP' (wh-pronoun - e,g. who, what) for 'NEU', 'NNPS' (proper noun in plural form) for 'NEG' or 'UH' (interjection) for 'POS'. As far as the 'Target' model is concerned, the best-indicating feature values were the predicted polarities of 'NEG' and 'POS' for the class 'T' and 'NEU' for the class 'NT'. This acted as an additional verification for the reported benefits from the use of the predicted polarity class as part of the feature set for the target detection subtask. It is also important to note that the best indicating state feature for 'NEG' partially confirmed one of the hypotheses made earlier about the reason why this class was underperforming when the word scope was equal to 1. Due to the fact that this state feature was the PoS tag of the token located three positions before the token of interest, this relation could not be leveraged by the model before, since tokens located that far were not concerned.

6.2.2 LSTM-Based Models

To facilitate the purposes of the experiments to be conducted for determining the best configuration for the two supervised components of the complete sentiment analysis problem, a basic baseline system (model) was implemented. For each experiment, a particular aspect of this system was modified to investigate the resulting impact on the performance. After comparing the results of the different configurations, the one that yielded the best performance persisted and was carried over as the baseline for the next set of experiments.

The baseline system was built upon the sequence tagging model based on a Bidirectional LSTM network as described in Subsection 5.5.1. Each model was trained for a maximum of 200 epochs using Stochastic Gradient Descent (SGD) with a mini-batch size of 32. The initial learning rate was set to 0.5 but it was adapted accordingly as the training process progressed. Finally, the hidden layer was consisted of 256 neurons.

Experiment 1 - Class Balancing

In Chapter 5, $M2MR$, a metric for quantifying the level of class balance in a dataset was introduced and justified. To demonstrate its use in an actual experimental setup, the $M2MR_c$ scores for the training and test splits of *ZyLABSent* as well as for the entire dataset were calculated and they were equal to **0.1536**, **0.1535** and **0.1540**, respectively. Considering that, according to its definition, $M2MR_c$ ranges from 0 to 1 where the latter corresponds to a completely balanced dataset, the *ZyLABSent* appears to be highly imbalanced on its original form.

Aiming to mitigate this, different techniques were implemented and evaluated. These techniques were described in detail in Subsections 5.3.2 and 5.3.3. To measure their effectiveness in accomplishing this goal, the $M2MR_c$ scores of the training split after applying each technique on it were calculated and they are listed on Table 6.6. It is worth mentioning that for the two augmentative techniques (i.e. 'Proportion-Based' and 'Minority Augmentation'), a single generated instance was included on the new version of the dataset.

These strategies had to be applied only on this particular split so that it can be used for training models which were less susceptible to the possible effects of class imbalance. However, these models would eventually be used on the test data which would not be necessarily balanced. Thus, the models' performance should be assessed on the basis of how well they would perform on a more realistic setting; one that would quite probably include imbalanced data. This is why none of these methods was applied on the test split.

Dataset Configuration	**Training**
Original Imbalanced	0.1535
Ideally-Balanced	1.0000
Proportion-Based	0.4227
Majority Reduction	0.6942
Minority Augmentation	0.2303

Table 6.6: The $M2MR_c$ scores for the training split after applying the different class balancing techniques

As it can be deduced from the results, using the approach described in Subsection 5.3.2, it was possible to generate a fully balanced dataset capable of achieving a perfect $M2MR_c$ score of 1. However, this dataset could be considered as idealistic since, due to the nature of sentences, it is hardly possible to have a dataset which even approximates complete class balance. When the three other more realistic approaches were compared, the one which yielded the most balanced dataset was 'Majority Reduction'. Despite the fact that both the other two approaches ('Proportion-Based' and 'Minority Augmentation') resulted to a more balanced dataset than the original setting, their $M2MR_c$ scores indicated that they were still closer to a state of imbalance rather than the opposite end. This was expected since 'Majority Reduction' was the only method which limited the occurrences of the majority class in a deterministic way without affecting the counts of the minority classes. This was a guarantee that $M2MR$, being the ratio of these two counts, would increase.

The 'Proportion-Based' approach had a strong probabilistic element which affected its effectiveness. Although balanced settings were indeed rewarded by receiving more probabilities of being drawn still the actual sampling process remained random. Albeit being highly unlikely, it was possible that all the sentence slices to be drawn would be imbalanced; even in a larger degree than the original sentences. At the same time, though, it was also capable of selecting the most optimal dataset configuration (practically identical to the product of the ideal dataset construction above) which would result to a $M2MR_c$ of 1. This broad range of possible outcomes was what made this solution the one which resembled reality to the greatest extent.

On the other hand, 'Minority Augmentation' acted more as a way to boost the minority classes rather than actually balancing the dataset. This is evident on the resulting $M2MR_c$ score which was just slightly better than the original one. Since it incorporated more (artificial) sentences with new minority class instances, it certainly boosted their overall presence in the dataset. However, at the same time, it additionally enriched the dataset with a proportional amount of majority class tokens. This simultaneous increase would have possibly prevented $M2MR_c$ from changing too much.

However, as it is reflected on the F1-Scores of Table 6.7, class balance in the training set is not always correlated with high classification performance on the test set. Starting off with the most balanced setting (**CB1**), it can be seen that the aptitude towards detecting target tokens was almost half than the original dataset (**CB0**) and was accompanied with an important decrease in the individual F1-Scores of the individual classes. The only exception to this was the performance on the minority class of the 'Polarity' model (i.e. 'NEG') whose performance increased. Similar behaviour can be observed for the second most balanced setting; 'Majority Reduction' (**CB3**). A significant drop can be seen both on the per-class but, also, for the overall scores. This can be explained by the fact that although the training set was now more balanced, the test set still resembled the class distribution of the original dataset. Therefore, a model trained over balanced data might not be as robust when tested on an imbalanced dataset.

Index	**Dataset Configuration**	**Micro-Averaged**	**Weighted**	**NT**	**T**
CB0	**Original Imbalanced**	0.9227	0.6395	0.9572	0.6010
CB1	**Ideally-Balanced**	0.6806	0.4335	0.7835	0.3910
CB2	**Proportion-Based**	0.9001	0.6397	0.9429	0.6029
CB3	**Majority Reduction**	0.6538	0.4051	0.7626	0.3617
CB4	**Minority Augmentation**	0.9108	0.5845	0.9506	0.5401

Index	**Dataset Configuration**	**Micro-Averaged**	**Weighted**	**NEG**	**NEU**	**POS**
CI0	**Original Imbalanced**	0.9063	0.5219	0.0624	0.9528	0.4076
CI1	**Ideally-Balanced**	0.6449	0.5202	0.1859	0.7808	0.3029
CI2	**Proportion-Based**	0.8792	0.5465	0.1085	0.9447	0.3901
CI3	**Majority Reduction**	0.5958	0.2948	0.0169	0.7476	0.2234
CI4	**Minority Augmentation**	0.9051	0.6538	0.1536	0.9538	0.4641

Table 6.7: The resulting F1-Scores from the experimentation with the different balancing techniques

On the contrary, the other two balancing techniques yielded some interesting results. They both managed to outperform the baseline (**CB0**) but for different model each. Specifically, the 'Proportion-Based' (**CB2**) one improved the Weighted F1-Score of the first model whilst the 'Minority Augmentation' (**CB4**) boosted the performance of each individual class of the 'Polarity' subtask. Thus, for the following experiments, **CB2** and **CB4** replaced **CB0** in the baseline model configuration. It should be noted, though, that the effects of the best-performing balancing techniques on the 'Target' task were significantly less evident than the 'Polarity' model.

Experiment 2 - Embeddings Configuration

To investigate the effects from the use of various types of embeddings, five models using different embedding configurations were trained and compared. The first one (**E0**) was the best-performing model from the previous section that made use of BERT embeddings. Flair embeddings (bidirectional, trained on a mix of corpora from six different languages) were part of the second model (**E1**) to determine which of the two recently-invented contextual embeddings yielded the best results. Additionally, a third model (**E2**) utilising GloVe embeddings was introduced as a reference point for the impact of contextual embeddings over simple context-insensitive ones. Finally, in an attempt to validate the observation of Akbik et *al.*[2] as stated in Subsection 5.4.2 and unravel any possible performance gains from the use of combined context-sensitive and context-insensitive embeddings, two composite configurations of embeddings were employed consisting of either BERT (**E3**) or Flair (**E4**) with GloVe stacked on top them.

Between the two types of contextual embeddings, BERT (**E0**) was the one that resulted to the best overall performance for both tasks. While it offered a slight increase in the F1-Score of the 'NEG' class of the 'Polarity' model, Flair (**E1**) clearly underperformed in the prediction of all the other classes. It is worth mentioning that, in the case of 'POS' class specifically, it led to a drop of

Index	Embeddings Configuration	Micro-Averaged	Weighted	NT	T
E0	**BERT**	0.9001	0.6397	0.9429	0.6029
E1	**Flair**	0.8803	0.5966	0.9308	0.5560
E2	**GloVe**	0.8928	0.6022	0.9390	0.5613
E3	**BERT + GloVe**	0.8950	0.6319	0.9397	0.5946
E4	**Flair + GloVe**	0.8875	0.6197	0.9350	0.5814

Index	Embeddings Configuration	Micro-Averaged	Weighted	NEG	NEU	POS
E0	**BERT**	0.9051	0.6538	0.1536	0.9538	0.4641
E1	**Flair**	0.8612	0.5745	0.1697	0.9364	0.3613
E2	**GloVe**	0.8789	0.5819	0.1285	0.9455	0.4096
E3	**BERT + GloVe**	0.8861	0.8261	0.3306	0.9452	0.4812
E4	**Flair + GloVe**	0.9027	0.6718	0.1674	0.9523	0.4706

Table 6.8: The resulting F1-Scores from the experimentation with different word embeddings

0.10 when compared to BERT.

Mixed results were produced by the stacked configurations of embeddings. For the 'Target' model, they did not manage to outperform BERT despite the fact that they had a comparable performance. On the contrary, for the 'Polarity' component, both stacked embeddings performed better in the prediction of the minority classes than their single-embeddings counterparts and generated higher weighted scores. In particular, the stacked combination of BERT with GloVe doubled the F1-Score for the 'NEG' class and led to a significant increase for the 'POS' one, too. Thus, the statement by Akbik et *al.*'s[2] was, partially, verified based on its effects on the 'Polarity' model.

Experiment 3 - Use of a CRF Layer

The next couple of experiments aimed to investigate whether the inclusion of an extra CRF layer on top of the Bidirectional LSTM would result to any additional performance gains for the classification process. Without the use of this layer, the classification was performed using softmax. Otherwise, the CRF unit undertook the classification process itself.

Index	CRF Layer	Micro-Averaged	Weighted	NT	T
CRF0	**No**	0.9001	0.6397	0.9429	0.6029
CRF1	**Yes**	0.8900	0.5961	0.9373	0.5547

Index	CRF Layer	Micro-Averaged	Weighted	NEG	NEU	POS
CRF0	**No**	0.8861	0.8261	0.3306	0.9452	0.4812
CRF1	**Yes**	0.8918	0.5515	0.0874	0.9470	0.4165

Table 6.9: The resulting F1-Scores from the experimentation with the use of a CRF layer

As the experiments demonstrated, the system configuration which included a CRF component (**CRF1**) resulted to a decreased performance for almost all the individual classes; with a small exception being the 'NEU' class. Thus, the best-performing configuration from the previous experimental iteration remained intact.

Experiment 4 - Data Augmentation

The fourth and final group of experiments attempted to unveil whether enriching the training dataset with more generated data would lead to any improvements in performance. Both the balancing techniques that were employed on the two different models ('Proportion-Based' and 'Minority Augmentation') have an underlying augmentative nature. However, up to this point, they were used only for balancing the dataset and not for providing more training samples. Thus, they were parametrised to return just one instance per sentence.

For the following experiments, the class balancing algorithms which tuned the dataset configuration were instructed to generate various numbers of samples. Starting off with 10 slices and reaching up to 40, each experimental setup aimed to determine whether increasing the dataset by varying amounts of samples would help the model become more robust in identifying interesting patterns and, therefore, be more precise in its predictions. The parameter which determined the number of instances was considered as an upper bound to the amount of samples to be used; if for a given sentence less slices were generated, all of them were added to the training set.

Index	# Slices	Micro-Averaged	Weighted	NT	T
DA0	**1**	0.9001	0.6397	0.9429	0.6029
DA1	**10**	0.8886	0.6135	0.9359	0.5744
DA2	**20**	0.8807	0.6118	0.9307	0.5731
DA3	**40**	0.8922	0.6261	0.938	0.5883

Index	# Slices	Micro-Averaged	Weighted	NEG	NEU	POS
DA0	**1**	0.8861	0.8261	0.3306	0.9452	0.4812
DA1	**10**	0.8959	0.8422	0.3325	0.9494	0.4968
DA2	**20**	0.8890	0.8108	0.3089	0.9463	0.4856
DA3	**40**	0.8899	0.8379	0.3352	0.9459	0.4897

Table 6.10: The resulting F1-Scores from the experimentation with augmenting different numbers of slices to the training set

As it can be spotted on the results table, augmenting the first model ('Target') with additional data did not result to any improvements on its performance. Although when looking at the Weighted F1-Scores for all the new configurations, it was possible to observe a pattern of performance growth for the prediction of the minority classes, yet, the highest-performer of them (**DA3**) still produced inferior results when compared to the original setting. On the contrary, for the second model ('Polarity'), the augmentation with more training data had immediate positive effects on the prediction of the minority classes as it was reflected on both their individual scores and the Weighted metric. However, this trend did not persist for all settings; the use of ten slices (**DA1**) appeared to be the most optimal out of them. Thus, for the final version of the models (shown in Table 6.11), the configuration of the first model did not change from the previous experiment (**DA0**) whilst, for the second one, the augmentation of the training set with at most 10 slices per sentence (**DA1**) was now incorporated.

Setting	**Target Model**	**Polarity Model**
Dataset Configuration	Proportion-Based	Minority Augmentation
Type of Embeddings	BERT	BERT + GloVe
CRF Layer	No	No
Number of Slices	1	10
Micro-Averaged F1-Score	**0.9001**	**0.8959**

Table 6.11: The final configuration of the two models based on the results of the experimental process

Error Analysis

Since the optimal configurations for both models were determined based on the results of the experiments, a more qualitative method of reviewing and evaluating their performance was, additionally, employed to justify their behaviour. In this section, a number of examples with misclassified instances are listed and analysed with the aim of determining the source of errors.

EXAMPLE #1
Sentence: "*'Breakfast At Tiffany's' is perhaps the best 'date' movie ever conceived.*"
Ground Truth: T: [Breakfast at Tiffany's], NEU: [], NEG: [], POS: [Breakfast at Tiffany's]
Prediction: T: [Breakfast at Tiffany's], NEU: [Breakfast at], NEG: [], POS: [Tiffany's]
****
This sentence showcases a possible shortcoming of the 'Polarity' model when dealing with targets consisted of multiple tokens. Instead of treating the target entity segment of the sentence as a contiguous sequence of the same polarity, it treats all but the last token as neutral entities. It seems to assume that the first few tokens just provide an additional description of the target entities instead of actually being part of them. A possible cause for this behaviour could be the existence of the word 'at' which is frequently used between two separate entities to highlight a positional relationship between them.

EXAMPLE #2
Sentence: "*Do not buy this camera.*"
Ground Truth: T: [this camera], NEU: [], NEG: [this camera], POS: []
Prediction: T: [this camera], NEU: [this], NEG: [camera], POS: []

****
Here, a similar issue to the abovementioned one is demonstrated but, this time, occurring on a target entity with negative polarity. Despite the fact that the target entity was correctly predicted to be 'this camera', the tokens which it is comprised of were labelled with sentiments of different polarity ('NEU' and 'NEG', respectively). In this case, the incorrectly-tagged token was 'this'. During annotation, it was labelled as part of the target entity since it acted as an additional specification for it despite being a demonstrative pronoun on its own right. This might be the reason why it was perceived to be neutral. It would be quite justified to expect that most of its occurrences in the training set would have been outside the span of a target entity and, therefore, considered to be neutral. Thus, the 'Polarity' model may have now failed to recognise its special role as part of a target of negative sentiment.

EXAMPLE #3
Sentence: "*Try the lobster teriyaki and the rose special roll.*"
Ground Truth: T: [lobster teriyaki, rose special roll], NEU: [], NEG: [], POS: [lobster teriyaki, rose special roll]
Prediction: T: [lobster teriyaki, rose special roll], NEU: [lobster teriyaki], NEG: [], POS: [rose special roll]
****
In the above sentence, the 'Polarity' model, unlike the 'Target' one, did not manage to associate a positive polarity to both the detected target entities. Like demonstrated in a previous example, there seemed to be a tendency to associate an expressed sentiment to either the last token of a single entity or the last target entity of many.

A possible solution to this could be to feed the model with more training samples consisted of either multiple target entities or multi-token entities, ideally with differences in polarity. It is safe to assume that the vast majority of sentences in the dataset were consisted of single-token target entities. It appears to be quite unlikely to find multiple target entities appearing in the same sentence; especially in the context of online reviews which tend to be comprised of relatively short sentences. Thus, it is plausible that the models did not develop the same aptitude towards classifying sentences which did not fit this specification.

EXAMPLE #4
Sentence: "*noodles with shrimp and chicken and coconut juice is the MUST!*"
Ground Truth: T: [noodles with shrimp and chicken and coconut juice], NEU: [], NEG: [], POS: [noodles with shrimp and chicken and coconut juice]
Prediction: T: [noodles, shrimp, chicken, coconut juice], NEU: [], NEG: [], POS: [noodles, shrimp, chicken, coconut juice]
****
This is an example of a sentence in which both classifiers failed to detect the actual boundaries of a multi-token target entity. It is worth noting, though, that there was a correspondence between what the two models considered to be

the target. Additionally, the 'Polarity' model, correctly, regarded them as being targets of positive sentiment individually.

One possible source of confusion might have been the existence of the words 'with' and 'and'. In the formation of speech, these words are used to denote some sort of relationship between entities. In this case, 'with' associated 'noodles' with 'shrimp' whilst 'and' related 'chicken' and 'coconut juice'. Looking from one level higher, the intermediate 'and' connects the composite entities of 'noodles with shrimp' and 'chicken and coconut juice' together to form the final target entity (i.e. 'noodles with shrimp and chicken and coconut juice'). However, the model failed to spot these complex relationships between the entities and, thus, considered the appearance of the functional words as an indication for the beginning of a new target entity.

6.3 Opinion Holder Extraction

6.3.1 Performance Evaluation

In contrast with the other two sentiment analysis elements which were predicted by means of supervised learning, this third ('Opinion Holder') component was not associated with a set of truth values which could be used for testing. The main means of understanding how well it performed during implementation was with the use of a qualitative, sentence-by-sentence manual assessment.

In order to be able to evaluate the performance of this component in a similar manner as the other two, a set of 100 randomly sampled sentences from *ZyLABSent* were manually annotated with respect to the person who expressed the opinions they included. Firstly, to get an idea of its performance in a per-sentence manner, the ground truth values along with the predicted ones were compared one by one. For 96 out of the 100 sentences (i.e. a success rate of **96%**), these two values matched completely. The four sentences for which the system detected different results will be presented and analysed in Subsection 6.3.2.

Furthermore, the performance was evaluated in a per-token basis. A token-based representation of the sentences was tagged with either 'H' or 'NH' depending on whether the token was part of a detected entity string or not. Finally, the sentence was compared with the ground-truth version of it. The per-token results showed F1-Scores of **0.9200** and **0.9970** for the 'H' and 'NH' classes, respectively and a Micro-Averaged score of **0.9940**.

6.3.2 Error Analysis

To be able to precisely detect the shortcomings of the devised approach, an error analysis process was applied on sentences whose opinion holder was not correctly identified by the algorithm. This procedure aimed to uncover any aspects of this rule-based approach which do not work in practice and attempt to justify why.

EXAMPLE #1
Sentence: "*Friendly staff that actually lets you enjoy your meal and the company you are with.*"
Ground Truth: (Author)
Prediction: You
****
This is an example of a sentence in which the actual opinion holder (Author) conveys his/her own opinion through the use of the second person to help the reader relate more with the described experience. As it can be understood, this is a pretty complex language technique which a human assessor learns to detect and interpret by taking the overall context into consideration and it cannot be expressed by any rule based on words or their functionality in a sentence. The algorithm considered 'enjoy' as a thinking/perceiving word and resolved 'you' as its subject on the the sentence's SDT and thus declared it to be the opinion holder.

EXAMPLE #2
Sentence: "*I/we will never go back to this place again.*"
Ground Truth: I/We
Prediction: We
****
Here, an instance of a partially correct extraction is provided. 'I/We' which, according to the ground truth, is the actual opinion holder, was overlooked by the system due to its token-by-token form of traversal. It tokenised the sentences and processed each word individually. Thus, 'I/We' was separated and perceived as two different tokens, instead of a conjugated entity like it is in reality. As a consequence, the system initially perceived 'I' as the opinion holder and then switched to 'We' as soon as it iterated over the next token. This incident was a side-effect from the use of abbreviative symbols like '' whose meaning corresponds to an instance of conjunction that, if expressed in full-word form, could have been easily perceived as so by the system.

EXAMPLE #3
Sentence: "*My boyfriend had Prime Rib it was good .*"
Ground Truth: (Author)
Prediction: I
****
This error can be attributed to the incorrect satisfaction of the "Thoughts/Experiences in the First-Person Form" rule. The occurrence of 'My' at the beginning of the sentence and its resolution to the subjective 'I' indicated the existence of an underlying person sharing his/her own experience on something. In reality, this person was just relaying an experience of somebody else along with a remark about it. However, it is not clear from the provided context whether the opinion was formed by this other person or the speaker. Thus, it should have been assigned the generic '(Author)' token instead.

EXAMPLE #4
Sentence: "*A little noise but I think that was because of our party!*"
Ground Truth: I
Prediction: We
****
Here, this example of misdetection originated from the incorrect application of the "Thoughts/Experiences in the First-Person Form" rule which regarded the appearance of any first-person pronoun as a trigger for changing the opinion holder to the corresponding first-person subjective pronoun. In this case, 'I' was initially the opinion holder but as soon as the 'our' token was detected and resolved to its subjective form ('We'), the subject was changed to this instead. In general, the rules defined in this system enforced the concept of 'We-Over-I', in the sense that, if the possibility of a more collective opinion holder emerged, it immediately overrode the 'individual-person-opinion' scenario. This action could have been correct if 'our' was not preceded by 'I think' which gave a clear hint on to who the actual subject was but, due to the nature of this rule-based system as a one-pass technique, when a condition had been satisfied and a change of opinion holder was triggered, it was no longer possible to look back.

As the above examples demonstrated together with the numerical performance scores, the implemented opinion holder extraction component was proved to be quite robust and effective in general. Unfortunately, it failed to detect the correct entities in situations where the contextual information provided solely by the sentence tokens was ambiguous or not sufficient on its own and thus some form of deductive thinking was required. The same holds for the cases when the opinion holder appeared in some non-standard form. However, considering the remarkable results it achieved with relatively limited resources and without the need of annotated data, the performance of the system can be considered more than satisfactory.

Chapter 7

Conclusions

Research Question: "(*Main Goal*) Is it possible to provide a complete and novel solution to the sentiment analysis task involving the detection of the *opinion holder*, the *polarity* and the *target*?"

This thesis aimed to provide a complete solution to the challenging task of sentiment analysis by breaking it down to its three fundamental components: opinion holder, sentiment polarity and target. The first element was tackled by employing a simple rule-based method. The last two were approached by means of supervised learning. The result from the application of full sentiment analysis on a sentence was a sentiment triple consisted of the aforementioned elements which could be visualised in the form of a graph of two adjacent nodes in a network-based representation.

For the two supervised subtasks, two different types of architectures were considered. The first one was based on CRFs and built up on a previous study which used various configurations of them for similar purposes. This experimental iteration aimed to investigate whether these models could, also, be used in this context and additionally verify some of the observations made about them as well as regarding the internal dependencies between the tasks of target and polarity detection.

The second class of approaches involved the employment of deep learning by means of Bidirectional LSTMs which are currently the state-of-the-art for several sequence tagging problems. Different experiments were performed aiming to assess the effect of various model parameters on the model's performance. The experimental variables included different dataset configurations, various types and combinations of contextual and context-insensitive embeddings, the inclusion of an extra CRF decoding layer on top of the existing architecture and the augmentation of additional data to enrich the training set. The best-performing models for the 'Target' and 'Polarity' subtasks have reached a Micro-Averaged F1-Score of **0.9001** and **0.8959**, respectively.

Finally, the 'Opinion Holder' subtask was tackled by devising a rule-based approach that leveraged syntactical and grammatical patterns which frequently appear in opinionated sentences. To evaluate its performance quantitatively, a

subset of sentences was manually annotated with respect to their opinion holder and given as input to this component. The system managed to correctly identify the opinion holder in **96%** of the sentences. From the sentence-tokens point-of-view, this performance was translated in a Micro-Averaged F1-Score of **0.9940**.
****

Research Question: "Do the newly-introduced contextual embeddings (i.e. BERT and flair) which have already yielded some remarkable results on other NLP tasks provide any performance gains when used for sentiment analysis for the first time?"

Amongst the different configurations of embeddings employed in the experimental process were four which involved the use of BERT and flair either solely or as on a stacked architecture along with GloVe. For both 'Target' and 'Polarity' models, BERT embeddings were part of the best-performing system; alone for the former and together with GloVe for the latter.
****

Research Question: "How to cope with the lack of target-oriented annotated datasets?"

An amalgamation of two publicly available review-based datasets from various domains was used as the development corpus for the purposes of this study. Before being able to use these two datasets together, one of them had to be subject of an annotation process that aimed to associate its sentences with information regarding the target entities that appear in their body as detected by the human assessors.
****

Research Question: "How to quantify and assess the briefly-explored issue of class imbalance in a sequence tagging dataset? Which approaches can be used for mitigating this problem?"

Instead of treating the input sentences as a whole, the proposed solution represented them as sequences of tokens which were destined to be allocated with three different labels indicating their sentimental polarity and whether they were opinion holding or target entities. This meant that the overall problem was abstracted as a sequence tagging task. However, this form of representation resulted to the emergence of class imbalance as a serious issue which affected the two supervised subtasks and required further study and mitigation. Consequently, different class balancing approaches were introduced and explained thoroughly in this study as well as evaluated via an experimental procedure. These solutions were accompanied with the definition of several metrics that aimed to quantify and evaluate the effect of class imbalance on a given dataset.
****

Research Question: "How can meaningful data augmentation take place in a sequence tagging context considering the different constraints involved? What will be the effects from the inclusion of augmented data in the training set for its class balance and the performance of the resulting model?"

Two of the proposed class-balancing approaches ('Proportion-based' and 'Minority Augmentation') could be alternatively used for augmenting data to an existing sequence tagging dataset to incorporate more variety to the trained models. The first one augmented different slices of the original sentences whilst the other constructed artificial data by replacing the minority tokens of existing sentences with their synonyms. Both approaches were carefully implemented to ensure that the generated slices were both meaningful and adequately balanced. These two approaches were quite novel for their kind since data augmentation in NLP and sequence tagging in particular is still in very early stages. It is worth noting that the 'Minority Augmentation' approach resulted to significant improvement in the classification performance for the 'Polarity' model.

Chapter 8

Future Work

The experiments conducted for the purposes of this study had led to a few interesting results and fruitful conclusions. Some of them could be used as the starting point for further investigation and future experimental iterations. This chapter aims to provide suggestions for follow-up work which may result in the conception of even more refined solutions for the interesting and challenging task of complete sentiment analysis.

Firstly, for class imbalance handling, the only widely-used technique which was neither applied directly nor adapted to the context of sequence tagging was the training loss function modification using weights that increase the impact of minority classes and reduce the effect of majority. A weighted metric such the one introduced in Section 6.1 could be employed for optimising the model so that it could demonstrate an aptitude in classifying minority instances correctly.

Secondly, more attention could be invested on the 'Polarity' model of the sentiment analysis task since, according to the results and the error analysis, it is the one which has the biggest room for improvement. Since data augmentation was proved to be beneficial for it, new ways of augmenting data could be devised which would introduce even more variety to the model. More focus should be given to the negative class which underperformed compared to the rest. Additionally, as the error analysis has shown, this model was characterised from a weakness in recognising the correct boundaries of multi-token target entities. Although, most of the times, it successfully detected at least one of these tokens and assigned a correct polarity, it often failed to identify the entire span of tokens. This could possibly be resolved using the significantly-more-powerful 'Target' model as an extra source of boundary information for adjusting its own estimations.

The proposed solution for complete sentiment analysis is tailored for use in text corpora written in one particular language. The 'Opinion Holder Extraction' component was constructed by putting together different syntactical rules that characterise the english language. Likewise, the different machine learning models were trained using a dataset of English reviews and made use of embeddings trained either solely on this language or, also, on a limited number of additional languages. However, the ultimate goal is to construct a system flexible enough so that it can adapt itself to any language without the need of specific resources or language-dependent pre-trained models.

As far as the proposed visualisation is concerned, there are different ways to handle the possible issues that may emerge which were discussed in Section 5.6. To start off, the visualisation itself could be enriched with a filtering feature that would restrict the amount of nodes displayed on screen. Depending on the number specified by the user, only the n nodes with the largest number amount of links will be presented. Additionally, to tackle the issue with the multiple occurrences of the same entities appearing in the network, nodes that seem to corresponding to the same entity could be clustered together to remove some noise from the visualisation. All entities consisted of identical tokens would be considered as just one entity represented by a single node on the graph. It is worth noting that this may result to the opposite problem, namely the incorrect clustering of similar entities together. This is particularly likely in the case of personal pronouns or other co-referring instances which could not be always successfully resolved to a noun entity. However, considering that a single sentence does not always convey enough information about the overall context from which it originates, this could be considered as the most accurate it can get based on the available resources.

Finally, since the results from the CRF-based models demonstrated that some syntactical and lexicographical features (e.g. PoS tags) are highly indicative for the prediction of the different model classes, they could also be employed for the LSTM-based approaches. These features could be added to the feature set to enrich and complement the different configurations of word embeddings and possibly lead to an increase in the performance.

Bibliography

[1] Apoorv Agarwal et al. “Sentiment analysis of Twitter data”. In: *Proceedings of the Workshop on Language in Social Media (LSM 2011)* (2011), pp. 30–38.

[2] Alan Akbik, Duncan Blythe, and Roland Vollgraf. “Contextual String Embeddings for Sequence Labeling”. In: *COLING 2018, 27th International Conference on Computational Linguistics* (2018), pp. 1638–1649.

[3] Luciano Barbosa and Junlan Feng. “Robust Sentiment Detection on Twitter from Biased and Noisy Data”. In: *Coling 2010* Poster Volume (2010), pp. 36–44.

[4] John Blitzer, Mark Dredze, and Fernando Pereira. “Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification”. In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* (2007), pp. 440–447.

[5] Leon Bottou. “Une approche theorique de l’ apprentissage connexionniste: Applications a la reconnaissance de la parole”. PhD thesis. Universite de Paris XI, 1991.

[6] Nitesh V. Chawla et al. “SMOTE: Synthetic Minority Over-sampling Technique”. In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[7] Kevin Clark and Christopher D. Manning. “Deep Reinforcement Learning for Mention-Ranking Coreference Models”. In: *CoRR* abs/1609.08667 (2016), pp. 1–6.

[8] Ronan Collobert et al. “Natural Language Processing (Almost) from Scratch”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2493–2537.

[9] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018), pp. 1–14.

[10] Jacob Eisenstein. *Lecture notes for the Courses CS 4650 and CS 7650 (”Natural Language”).* School of Interactive Computing, Georgia Tech. Nov. 2018.

[11] Andrea Esuli and Fabrizio Sebastiani. "SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining". In: *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC'06* (2006), pp. 417–422.

[12] Alec Go, Richa Bhayani, and Lei Huang. *Twitter Sentiment Classification using Distant Supervision.* 2009. URL: `http://www.stanford.edu/~alecmgo/papers/TwitterDistantSupervision09.pdf`.

[13] Yoav Goldberg. "A Primer on Neural Network Models for Natural Language Processing". In: *CoRR* abs/1510.00726 (2015), pp. 1–75.

[14] Zhiheng Huang, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF Models for Sequence Tagging". In: *CoRR* abs/1508.01991 (2015), pp. 1–10.

[15] Ozan Irsoy and Claire Cardie. "Opinion Mining with Deep Recurrent Neural Networks". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (2014), pp. 720–728.

[16] Frederick Jelinek. "Markov Source Modeling of Text Generation". In: *The Impact of Processing Techniques on Communications.* Ed. by J. K. Skwirzynski. Dordrecht: Springer Netherlands, 1985, pp. 569–591.

[17] Yoon Kim et al. "Character-Aware Neural Language Models". In: *CoRR* abs/1508.06615 (2015), pp. 1–9.

[18] Julian Kupiec. "Robust part-of-speech tagging using a hidden Markov model". In: *Computer Speech & Language* 6 (1992), pp. 225–242.

[19] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". In: *Proceedings of the Eighteenth International Conference on Machine Learning* (2001), pp. 282–289.

[20] Xin Li et al. "A Unified Model for Opinion Target Extraction and Target Sentiment Prediction". In: *CoRR* abs/1811.05082 (2018), pp. 1–9.

[21] Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. "Maximum Entropy Markov Models for Information Extraction and Segmentation". In: *Proceedings of the Seventeenth International Conference on Machine Learning* (2000), pp. 591–598.

[22] Gregoire Mesnil et al. "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding." In: *INTERSPEECH* (2013), pp. 3771–3775.

[23] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings* (2013), pp. 1–12.

[24] Margaret Mitchell et al. "Open Domain Targeted Sentiment". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (2013), pp. 1643–1654.

[25] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. “Thumbs up? Sentiment Classification using Machine Learning Techniques”. In: *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing* 10 (2002), pp. 79–86.

[26] Jeffrey Pennington, Richard Socher, and Christopher Manning. “Glove: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014), pp. 1532–1543.

[27] Matthew E. Peters et al. “Deep contextualized word representations”. In: *CoRR* abs/1802.05365 (2018), pp. 1–15.

[28] Maria Pontiki, Dimitrios Galanis, and John Pavlopoulos. “SemEval-2014 Task 4: Aspect Based Sentiment Analysis”. In: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)* (2014), pp. 27–35.

[29] Maria Pontiki et al. “SemEval-2015 Task 12: Aspect Based Sentiment Analysis”. In: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)* (2015), pp. 486–495.

[30] Maria Pontiki et al. “SemEval-2016 Task 5: Aspect Based Sentiment Analysis”. In: *Proceedings of SemEval-2016* (2016), pp. 19–30.

[31] Alec Radford et al. “Improving language understanding with unsupervised learning”. In: *OpenAI Technical Report* (2018), pp. 1–12.

[32] Ellen Riloff and Janyce Wiebe. “Learning Extraction Patterns for Subjective Expressions”. In: *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing* (2003), pp. 105–112.

[33] Ellen Riloff, Janyce Wiebe, and Theresa Wilson. “Learning Subjective Nouns Using Extraction Pattern Bootstrapping”. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4* (2003), pp. 25–32.

[34] Ilya Sutskever, James Martens, and Geoffrey Hinton. “Generating Text with Recurrent Neural Networks”. In: *Proceedings of the 28th International Conference on International Conference on Machine Learning* (2011), pp. 1017–1024.

[35] Maite Taboada et al. “Lexicon-based Methods for Sentiment Analysis”. In: *Comput. Linguist.* 37.2 (2011), pp. 267–307.

[36] Peter D. Turney. “Thumbs Up or Thumbs Down?: Semantic Orientation Applied to Unsupervised Classification of Reviews”. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (2002), pp. 417–424.

[37] Ashish Vaswani et al. “Attention Is All You Need”. In: *CoRR* abs/1706.03762 (2017), pp. 1–15.

[38] Janyce Wiebe and Ellen Riloff. “Creating Subjective and Objective Sentence Classifiers from Unannotated Texts”. In: *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing* (2005), pp. 486–497.

[39] Bishan Yang and Claire Cardie. “Joint Inference for Fine-grained Opinion Extraction”. In: (2013), pp. 1640–1649.

[40] Kaisheng Yao et al. “Spoken language understanding using long short-term memory neural networks”. In: *2014 IEEE Spoken Language Technology Workshop (SLT)* (2014), pp. 189–194.