UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOKRITOS"
MSC PROGRAMME IN DATA SCIENCE

Aspect Based Sentiment Analysis on hotel reviews in Greek

by

Dimitrios Tsintzouras

A thesis submitted in partial fulfillment
of the requirements for the MSc
in Data Science

Supervisor: Georgios Petasis
Researcher (C)

Co-supervisors: Evangelos Karkaletsis, Christos Tryfonopoulos
Researcher (A), Assistant Professor

Athens, July 2021

Aspect Based Sentiment Analysis on hotel reviews in Greek

Dimitrios Tsintzouras

MSc. Thesis, MSc. Programme in Data Science

University of the Peloponnese & NCSR "Demokritos", July 2021

UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOKRITOS"
MSC PROGRAMME IN DATA SCIENCE

Aspect Based Sentiment Analysis on hotel reviews in Greek

by

Dimitrios Tsintzouras

A thesis submitted in partial fulfillment
of the requirements for the MSc
in Data Science

Supervisor: Georgios Petasis
Researcher (C)

Co-supervisors: Evangelos Karkaletsis, Christos Tryfonopoulos
Researcher (A), Assistant Professor

Approved by the examination committee on July, 2021.

(Signature)	(Signature)	(Signature)
.....	.....	.....
Georgios Petasis Researcher (C)	Evangelos Karkaletsis Researcher (A)	Christos Tryfonopoulos Assistant Professor

Athens, July 2021

UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOKRITOS"
MSC PROGRAMME IN DATA SCIENCE

Declaration of Authorship

(1) I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

(2) I confirm that this thesis presented for the degree of Bachelor of Science in Informatics and Telecommunications, has

(i) been composed entirely by myself

(ii) been solely the result of my own work

(iii) not been submitted for any other degree or professional qualification

(3) I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or processional qualification except as specified.

(Signature)

.....

Dimitrios Tsintzouras

Athens, July 2021

Acknowledgments

The current thesis was written, under the supervision of Dr. Georgios Petasis, for the degree of MSc in Data Science in the two-year full-time program organized by the University of the Peloponnese and the NCSR "Demokritos." First and foremost, I want to express my thanks to my supervisor Dr. Georgio Petasi for his invaluable assistance and support during this project. His knowledge and experience in the subject have been extremely beneficial to our task. However, the most important was his understanding and the psychological support he offered me in order to complete this project. Thanks to my colleagues and friends, for their support and help in many stages during this program. I'd also like to express my gratitude to my family, particularly my parents and brother, for their unwavering support and belief in me.

To my family and my ex.

Περίληψη

Τα τελευταία χρόνια, ένας αυξανόμενος αριθμός επιχειρήσεων έχει χρησιμοποιήσει την Ανάλυση Συναισθήματος για τα προϊόντα και τις υπηρεσίες τους, προκειμένου να προσαρμοστούν στις μεταβαλλόμενες απαιτήσεις των καταναλωτών. Η αναγνώριση συναισθημάτων από τα κείμενα (Sentiment Analysis) είναι ζωτικής σημασίας για να γίνει αυτό το έργο πιο αυτοματοποιημένο και αποτελεσματικό. Η ανάλυση συναισθημάτων επικεντρώνεται στην κατηγοριοποίηση του συνολικού συναισθήματος ενός κειμένου, απο το οποίο μπορούν να εξαχθούν χρήσιμες πληροφορίες, όπως απόψεις και κριτικές που σχετίζονται με συγκεκριμένες πτυχές και χαρακτηειστικά ενός προϊόντος ή μιας υπηρεσίας (ABSA). Η ανάλυση συναισθημάτων βάσει συγκεκριμένων πτυχών και χαρακτηριστικών είναι μια πιο δύσκολη διαδικασία προσδιορισμού του συναισθήματος.

Ως αποτέλεσμα των πρόσφατων ανακαλύψεων στη μηχανική μάθηση, η ερευνητική κοινότητα ενδιαφέρεται όλο και περισσότερο για την ανάλυση συναισθημάτων σε βάθος και συγκεκριμένα την ανάλυση συναισθημάτων που εκφράζονται για συγκεκριμένα χαρακτηριστικά προϊόντων. Έχουν προταθεί διάφορες αρχιτεκτονικές που μπορούν να παράγουν αποτελέσματα υψηλής ακρίβειας. Οι περισσότερες από αυτές τις προσεγγίσεις εφαρμόζονται συνήθως σε σύνολα δεδομένων της αγγλικής γλώσσας και είναι σαφές ότι οι προσπάθειες εφαρμογής τους σε άλλες γλώσσες είναι περιορισμένες.

Ο στόχος αυτής της διπλωματικής εργασίας είναι να εξετάσει το θέμα της ανάλυσης συναισθημάτων ως προς συγκεκριμένα χαρακτηριστικά στην ελληνική γλώσσα. Χρησιμοποιώντας, ως αφετηρία, ένα μικρό σύνολο κειμένων με κριτικές ξενοδοχείων, σχολιάσαμε τα έγγραφα προκειμένου να προσδιορίσουμε τα χαρακτηριστικά για τα οποία εκφράζεται κάποια γνώμη απο τους χρήστες και να προσδιορίσουμε την πολι-

κότητα του συναισθήματος που εκφράζεται ως προς αυτά. Στη συνέχεια, μερικές από τις πιο σύγχονες μελέτες μηχανικής μάθησης οι οποίες επιτυγχάνουν και την μεγαλύτερη απόδοση για αυτό το ερευνητικό θέμα στην αγγλική γλώσσα διερευνήθηκαν και τροποποιήθηκαν ελαφρώς για να εφαρμοστούν στο ελληνικό σύνολο δεδομένων. Συγκεκριμένα, εφαρμόζονται αρκετές αρχιτεκτονικές, όπως Recurrent Neural Networks (RNNs) και το Bidirectional Encoder Representations from Transformers (BERT) πολύγλωσσο μοντέλο. Τέλος, προτείνουμε ένα μοντέλο, το οποίο αποτελεί μια επέκταση του μοντέλου με την καλύτερη απόδοση στο θέμα αυτό, το LCF-BERT, με την προσθήκη ενός λεξικού στην αρχιτεκτονική του, το οποίο και ονομάζουμε ως LCF-OP BERT. Τα αποτελέσματα που προέκυψαν, ειδικά για την κατηγορία του ουδέτερου συναισθήματος, η οποία είναι η κλάση με τις λιγότερες εμφανίσεις στο σύνολο δεδομένων μας, είναι ενθαρρυντικά.

Abstract

In recent years, a rising number of businesses have used the feedback mechanism of reviews for their products and services in order to adapt to changing consumer demands. Sentiment identification from texts (Sentiment Analysis) is critical for making this work more automated and efficient. Sentiment analysis focuses on categorizing a text's overall sentiment, which may leave out essential information such as distinct sentiments associated with different aspects of the text. Aspect-Based Sentiment Analysis (ABSA) is a more difficult process of determining the sentiment of certain targets of a text.

As a result of recent breakthroughs in deep learning, the research community has become more interested in ABSA, and various architectures that can produce state-of-the-art results have been suggested. Most of these approaches are usually applied on English language datasets and it is clear that efforts to apply them on other languages are limited.

The goal of this thesis is to examine the topic of aspect-based sentiment analysis in the Greek language. Using, as a starting point, a small dataset with hotel reviews in the Greek language, firstly we annotated the documents in order to specify the aspects and their corresponding polarity. Then, some of the state-of-the-art studies used for this task in English language were investigated and altered slightly in order to apply them in our Greek dataset. Specifically, several architectures are applied, such as Recurrent Neural Networks (RNNs) and the pretrained Bidirectional Encoder Representations from Transformers (BERT) multilingual model.

Finally we propose a model, in essense an extension of the high-scored state-of-the-art model, named LCF-BERT, with the insert of a lexicon in its architecture in

order to further improve its performance. The obtained results, especially for the neutral sentiment class, which is the class with the less instances in our dataset, are encouraging, underlying the robustness of the proposed approach.

Contents

List of Tables

List of Abbreviations

NLP	Natural Language Processing
ABSC	Aspect Based Sentiment Classification
ABSA	Aspect Based Sentiment Analysis
NN	Neural Network
ANN	Artificial Neural Network
DNN	Deep Neural Network
RELU	Rectified Linear Unit
FNN	Feed-forward Neural Network
SGD	Stochastic Gradient Descent
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
Bi-LSTM	Bidirectional Long Short-Term Memory
GRU	Gated Recurrent Unit
BERT	Bidirectional Encoder Representations from Transformers
LM	Language Model
MLM	Masked Language Model

LSE	Least Square Error

Chapter 1

Introduction

Because of the Web's massive, ever-increasing growth, a large number of applications have been transferred from the physical to the digital environment. Shopping, banking and service booking are only a few examples of markets where daily activities are carried out online.

The way people absorb knowledge has changed as a result of this transition. We typically look for information on a small number of resources in the physical world. In the digital world, on the other hand, the available tools and volume of knowledge are so vast that manually checking and comparing all available options is impossible.

Today, an increasing number of people are posting feedback and reviews on the internet in quantities much greater than our ability to read. Users who are deciding between choices, such as purchasing a specific type of product, watching a movie, or going to a restaurant, will benefit from online feedback [1]. Users express their opinions about a wide range of products and their "aspects" such as characteristics, features, attributes, or components, through reviews. Hundreds, if not thousands, of reviews have been written for any hotel.

Reviews are typically written in free text format, and users must carefully read them to identify the expressed opinions and determine the strengths and weaknesses of the available hotels in order to make the right choice. Many sites are working to improve how users' views are shown. Using natural language processing, machine learning, and deep learning techniques, sentiment analysis systems strive to analyze

user textual feedback and extract the items aspects that are being addressed, as well as their opinion polarities, in order to address the knowledge overload issue and assist users in decision-making tasks.

1.1 Sentiment Analysis

The computational analysis of people's views, sentiments, attitudes, and feelings toward target entities such as goods, organisations, individuals, subjects, and their attributes is known as sentiment analysis or opinion mining [2]. The most of approaches to this area of study [3][4][5] attempted to identify the overall sentiment of a sentence, paragraph, or text regardless of the entities (e.g., restaurants, hotels) and their aspects (e.g., food, service) presented in context. However, only considering aggregate sentiments (such as a restaurant's total star ratings) fails to take into account opinions about the various aspects on which an entity can be evaluated [6].

In this thesis, We're looking for a more granular approach to evaluating the emotions recorded in user-generated hotel reviews. Improving content analysis performance of the growing amount of user-generated content on the web is one of the most exciting applications that motivates us to conduct this research. Summarizing details for an entity by searching for similar phrases with a high frequency over high-rated feedback is one of the most common techniques. The loss of essential knowledge by summarization approaches is a drawback of these approaches. Instead, we can easily create representative models for aspects of the entity and related feelings using ABSA on these broad sets of user-generated feedback.

1.1.1 Sentiment Analysis Levels

Sentiment analysis has primarily been studied at three distinct levels: document level, sentence level, and aspect level based on the levels of granularity of previous studies [2][7][8].

1.1.1.1 Document Level

The aim of document-level sentiment analysis is to figure out whether an opinionated document that reflects on an item has a positive or negative overall sentiment.

A opinion analysis method, for instance, classifies the overall polarity of a consumer review of a particular product. Since most sentiment analysis results only have two (positive and negative) or three (positive, neutral and negative) outputs, this level of sentiment classification means that one text communicates feelings about a single item, such as user feedback of goods and services.

However, it is not applicable to documents in which views are shared on several items since it is normal to have several separate opinions in one document. Document-level sentiment analysis has been activelly reasearched, leading to a variety of approaches [9][8][10]. They mostly concentrate on how to automatically distinguish positive and negative messages, as well as on various strategies for improving accuracy. The lack of in-depth analysis is a significant drawback in document-level sentiment analysis due to the basic performance of the sentiment classification [2].

1.1.1.2 Sentence Level

Sentiment analysis at the sentence level entails deciding whether each sentence conveyed a neutral, positive, or negative viewpoint [11]. Since sentences are only brief texts, there is no inherent distinction between document-level and sentence-level sentiment analysis [2].

This level is connected to the classification of subjectivity [12], which is used to differentiate between subjective sentences that convey feelings or opinions and logical sentences that express facts. The subjectivity classification is crucial because it eliminates sentences that do not have certain beliefs. The emotion classification at the sentence level means that each sentence represents a single viewpoint from a single person. Wilson [13] on the other hand, argue that a single statement can contain various views as well as subjective and objective clauses.

As a result, it's critical to locate objective terms and assess sentiment weight. The word "neutral" typically refers to factual sentences or sentences that are devoid

of views. It's also worth noting that, as Liu [2] notes out, subjectivity is not synonymous with emotion, because objective expressions can also mean emotions, such as "I bought this phone a week ago, and now the battery only lasts three hours." Yet another subset of opinionated phrases includes factual sentences that also suggest beliefs. Since compound sentences can be comparative or contain clustered opinions about various facets of an object, sentence-level classification is not appropriate [14].

1.1.1.3 Aspect Level

Labeling thoughts at document level or sentence level is helpful in many situations, but they are not adequate to include the required information needed for , since they do not specify emotion targets or allocate opinions to such targets [2]. At the document level, a positive statement on the subject does not mean that the author has a strong view on all facets of the subject. Apart from the sentence-level classification of emotion, it is always an intermediate stage, because it is more useful to know what the characteristics or persons of the object of opinions are. As a result, an aspect level that conducts a finer-grained analysis is required. Aspect level is often referred to as an agent level or a function level in some studies [15][8][16].

The level of examination of sentimental aspects relies on opinions themselves, rather than on constructing texts, such as sections, clauses and statements. It is not enough simply to figure out the polarity of opinions; the identification of opinion targets is also important [16]. The aspect-level sentiment analysis can be broken down into two sub-tasks: the extraction aspect and the classification of sentiment aspect [2]. The task of extracting aspect can also be seen as a task of extracting facts, with the goal of extracting the aspects from which opinions are based. For example, in the phrase, "The screen of this Samsung S6 is great, but its battery life is too small." "Screen" and "battery life" are the facets of the object defined by "Samsung S6." The basic approach to extracting aspects is to locate frequent nouns or noun phrases that are described as aspects. The text containing facets is then graded as positive, negative or neutral [17][18].

However, a frequent omission in aspect-level sentiment analysis relates to the identification of aspects, the majority of studies are focused on assumptions of pre-

specified aspects through keywords [19][20]. Ding [21] suggested a lexicon-based approach to aspect analysis, but consider that aspects are understood prior to analysis. Liu [2] points out that the accuracy at aspect level sentiment is still poor since the current algorithms are still unable to handle complicated sentences well. The aspect level study of emotion is also more complex than both the text level and the sentence level classifications. Mate [22] suggested a system for the ranking of aspects, but aspects are predefined prior to classification and lacks of findings and analyses.

1.1.2 Challenges

Sentiment analysis is a challenging task for computers to solve. Identifying such objects, characteristics or structures is difficult for automated approaches, or even impossible, although it is easy for human beings. Below [7], we enumerate some difficult scenarios for computers:

- It may be difficult to realize why the opposite interpretation of a sentence is expected when dealing with ironies or sarcasm. Special punctuation marks, such as (!!!), may also be used to identify ironies, but this is not a general law or symbol for these types of phrases.

- Another difficult challenge is resolving pronouns. Despite the fact that certain methods and algorithms can help, sentiment analysis remains a difficult challenge. For example, if a sentence contains opinion terms, but the accompanying feature is a pronoun, it is difficult to determine which feature is conveyed by such emotion words.

- It should also be acknowledged that defining the strength of an opinion is a difficult challenge in this field. Opinions vary in their strengths. Some of them are extremely powerful.

1.2 Aspect Based Sentiment Analysis Task

The goal of this thesis is to determine whether a set of aspects items in a text has a positive, negative, or neutral attitude. We need to calculate the sentiment associated to each unique aspect given a set of aspect keywords for a certain entity. Each feature can be ascribed one of three emotions: positive, negative, or neutral.

We are provided a phrase $s = [w_1, w_2, ..., w_i, ..., w_j, ..., w_n]$ and an aspect target $t = [w_i, w_{i+1}, ..., w_{i+m+1}]$ in this aspect level sentiment categorization problem. A single word or a big phrase might be used as the aspect target. The goal is to categorize the sentiment polarity of the sentence's aspect target.

1.3 Motivation

The reasons for performing research on NLP, particularly aspect based polarity classification, are many. To begin with, NLP is a fascinating and fast evolving discipline. One of the initial objectives of this master's thesis is to offer an overview of the current state of the art in aspect-based sentiment analysis.

The goal of this thesis is to use a small dataset to investigate the aforementioned topic in the Greek language. The majority of current ABSA research is centered on the English language and employs specialized datasets. As a result, efforts to learn other languages are limited.

To that end, we'll present a problem formulation, explain and compare important works in the ABSA job, and determine which of the available techniques is the most successful and appropriate for further investigation. The implementation of the chosen techniques to aspect opinion polarity categorization, and their application in the Greek language is our main objective. We also want to research on adaptations of existing techniques and suggest new approaches of this task.

1.4 Thesis structure

The thesis is structured as follows. In Chapter 2, the related work is presented. A brief technical description of the models used in the current thesis is provided in

Chapter 3. The state-of-the art approaches on ABSA task are briefly described in Chapter 4. Chapter 5 presents the dataset and the methodology that was followed. In Chapter 6, the experimental results are presented and finally, Chapter 7 concludes this thesis and presents future extensions.

Chapter 2

Related Work

In recent years, a number of approaches for dealing with the ABSA task have been developed, including more "traditional" machine learning and deep learning methods. We'll go through the related work of aspect-level sentiment classification in this section, which includes both conventional machine learning and deep learning approaches.

2.1 Rule-Based Sentiment Analysis

The use of sentiment-bearing words and their compositions to assess phrasal units for polarity has been a primary focus of sentiment analysis research for a long time [23]. Early research found that just counting valence words, or using a bag-of-words technique, can lead to inaccurate conclusions [24]. As a result, valence shifter research emerged, which incorporated variations in valence and polarity of phrases dependent on context usage [24][25].

To detect sentiment, however, valence shifters were insufficient; it was also necessary to comprehend emotion flows between syntactic units. Researchers developed the concept of modeling sentiment composition, which was taught using heuristics and rules [26], hybrid systems [27], and syntactic dependencies [28][29], among other methods.

The heart of rule-based sentiment analysis systems are sentiment lexicons. Simply said, these lexicons are dictionaries that provide sentiment annotations for the

words, phrases, or synsets they contain [30]. Such dictionaries are SentiWordNet, Linguistic Inquiry and Word Count, Multi Perspective Question Answering Subjectivity Lexicon (MPQA), General Inquirer etc.

While lexicons are useful for storing the sentiment polarity of words or phrases, using them to infer sentence-level polarities has proven to be difficult. Furthermore, no one lexicon can account for contextual polarity or all of the nuances observed from semantic compositionality [31][32]. In addition, lexicon building has other obstacles, such as overcoming subjectivity in annotations [33].

Although sentiment lexicons remain an important part of sentiment analysis systems, statistical techniques are becoming more popular, especially in low-resource situations. These strategies avoid the problem of rule coverage and provide you more options for dealing with generalization.

2.2 ABSA with Traditional Machine Learning Methods

Machine-learning-based statistical approaches have piqued interest in this field, owing to their independence from hand-engineered rules. Despite greatest efforts, the rules could never be exhaustively enumerated, which hampered generalization capabilities. Machine learning has made it possible to learn generic representations.

For the ABSC challenge, conventional machine learning approaches [34][35] are primarily focused on feature engineering. This also means that a significant amount of time is spent gathering and analyzing data, developing features based on the dataset's characteristics, and obtaining sufficient language resources to create lexicons. Rule-based methods [21] and statistics-based methods are two common representative methods for this mission [36][37]. Machine Learning based approaches to sentiment analysis, both supervised and unsupervised, have used a variety of algorithms, including SVMs [38], Naive Bayes Classifiers [39], and nearest neighbor [40], as well as features such as bag-of-words (including weighted variants) [41] and lexicons [42]. The majority of these works have been given a thorough examination in [2].

However, as with most conventional machine learning approaches, manually designing features is very time consuming and inefficient. Furthermore, as the dataset shifts, the method's output suffers significantly. As a result, conventional machine learning approaches have limited generality and are difficult to apply to a wide range of datasets.

2.3 ABSA with Deep Learning-Based Methods

Latest research is increasingly combining with Neural Networks (NN) as they have a noteworthy ability to capture original features and project them into low-dimensional, continuous vectors without the need for feature engineering. Via multiple hidden layers, Neural Networks (NNs) are capable of fusing original features to create new representations [23].

Recursive NN (Rec-NN) has been used for syntactic analysis and sentence sentiment analysis [43]. Rec-NN was used to classify aspect sentiment by translating the opinion target into the tree root and propagating the targets' sentiment based on the context and syntax relations among them [44][45]. Rec-NN, on the other hand, requires dependency decoding, which is unlikely to work on nonstandard texts like news comments and tweets.

Convolution NNs are used to determine the sentiment of a clause, which was then used to infer the target's sentiment. The approach assumes that the objective and the opinion word are in the same clause. The authors of [46] propose TD-LSTM, an RNN-based framework that can extract background features from both sides. However, when the opinion word is far from the mark, TD-LSTM does not perform well because the captured feature is likely to be lost (Cho et al., 2014) [47] identified similar problems with LSTM-based models in machine translation. Since the size of memory is limitless and we just need to read from it, the attention mechanism, which has been used effectively in many areas (Bahdanau et al., 2014; Rush et al., 2015) [48][49], can be treated as a simpler version of NTM. The attention mechanism in ATAE-LSTM [50] is used to concatenate the representations of aspect and meaning. Aspects will participate in the computation of attention weights using ATAE-LSTM. Previous research has focused on specific aspects such as in-

dependent and auxiliary data. The authors of [51] proposed IAN, which generates meaning and aspect representations. IAN uses an attention mechanism to learn the features of context and a selected aspect in interactive manner, hence improving the aspect and context learning process. The interactive learning of meaning and aspect terms was first suggested by IAN. RAM [52] uses a multilayer architecture based on bidirectional LSTMs, with attention-based token aggregation and Gated Recurrent Units (GRU [53]) to learn sentence features in each layer. RAM discovered for the first time that different environments contributed to learning in varying degrees.

A noteworthy development is that pretrained models have increasingly become a study hotspot of the ABSC mission. The key characteristic of a pretraining model is to train a highly generic Language Model (LM) based on large corpus resources. The pretraining model can be used to dramatically increase the performance of a wide number of NLP tasks. Pretrained language models ELMo [54] and GPT [55], which are based on LSTM and transformer, respectively, are designed to enhance the efficiency of several NLP tasks. The BERT text pair classification model was modified by the authors of [56] to complete the ABSC task. BERT-SPC prepares the input sequence by affixing aspect to contexts and treating them as two segments. LCF-BERT [57] suggested a feature-level local context concentration mechanism focused on self-attention for aspect level sentiment classification and many other fine-grained natural language processing tasks. Although the pre-trained model is based on a broad universal corpus, BERT-ADA [58] show that it is simple to apply to many tasks and improve results. It isn't, however, task-specific. If the pre-trained BERT is fine-tuned on a task-related corpus for particular tasks, task efficiency can be enhanced even further. Although the bulk of deep network studies use automated feature learning, their substantial reliance on labeled data might be constraining. As a result, adding inductive biases via syntactic information or external knowledge in the form of lexicons has become more popular (Tay et al., 2018b) [59].

2.4 Sentiment Analysis in the Greek Language

Sentiment analysis on Greek texts is rather rare, as the vast bulk of sentiment analysis research has been done on datasets in English. However, the rise of social

media platforms such as Facebook and Twitter has made multilingual material more accessible. As a result, there has been a recent increase in multilingual works [60]. In general, the NLP community is increasingly advocating for research on languages other than English.

From a linguist's perspective, works like [61], [62] deal with sentiment analysis for Greek and create lexicons of annotated words that may subsequently be used to evaluate a text word by word. They employ Logistic regression (LR), Random Forests (RF), and SVMs for classification, which differs significantly from our deep learning methods. Similarly, the authors create a lexicon that associates words with attitudes in [63], which is then utilized as a parameter in an algorithm that scans tweets and hashtags and identifies the sentiment represented within. In [64] they chose to look at hotel reviews, and as a result, they built a prototype for predicting sentiment polarity in modern Greek hotel reviews. They trained a machine learning algorithm using unigram language modeling. In [65] they suggest four distinct deep learning network architectures, each of which receives the review texts, the review texts plus some information on the tag annotation, or the text annotation alone as training input.

Chapter 3

Technical Background

The aim of this chapter is to provide a high-level overview of the models and evaluation metrics used in this thesis. The aim of this quick overview [66] is to help the audience understand the models that will be discussed in the following sections, as well as the metrics that will be used to assess the models' results.

A sentiment classification model has two essential processes from a theoretical perspective. First, there's the language representation method, which involves converting text data into real-valued vectors that contain language information. Then, using these vector representations, a classification model may find features to differentiate between sentiment classes.

3.1 ANNs

Machine learning is a form of modeling that uses algorithms to identify hidden patterns in a set of data without having to specify which features are needed to address a task. This is achieved by including a learning function for the algorithm to refine and a large data set to search for features in.

ANNs, which were first suggested by McCulloch and Pitts [67] based on the arrangement of neurons in the brain, are one of the basic foundations of machine learning. When combined across various architectures, these constructs have proven to be applicable to a wide variety of tasks, but the basic concept remains the same. We'll now go through some popular ANN models, as well as the logic and dynamics

behind how they work to model data.

First, we start with the data representation. Let **X** be a p-dimensional dataset with n samples.

$$X = \begin{bmatrix} x_{11} & x_{12} & ... & x_{1p} \\ ... & ... & ... & ... \\ x_{n1} & x_{n2} & ... & x_{np} \end{bmatrix}$$

Each row is a vector x_i which has a corresponding label denoted y_i. The aim is to model this output vector as accurately as possible using the data. This is best expressed as a linear combination of the input data, which yields the well-known linear regression

$\hat{y} = VX^T$, where V is called a weight vector and consists of real value elements.

However, the relationship between data and its corresponding label is not always linear in many applications. The problem of category prediction is an example of this. We want to predict a class rather than an unbounded real number, as is the case with linear regression. We want to predict a set of probabilities that the result y_i belongs to one of the classes k = 1,..., K. In an ideal world, a perfect model will generate a vector in which the target class's likelihood is 1 and the rest are 0. In reality, however, we are pleased if the target class's odds are higher than the rest.

By increasing the dimension of the original weight vector, we can construct a model that outputs a vector of K values rather than just one. The softmax function can then be used to scale these outputs to probabilities between 0 and 1, reflecting the likelihood that the output belongs to that particular class. Softmax fuction ϕ is given by

$$\phi(z_j) = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_j)} \text{ j = 1, . . . , K where K is the number of classes.}$$

We can make predictions by assuming V is a dimensional matrix (K x p).The resulting model is: $\hat{y} = \phi(VX_i^T)$ where $\hat{y} : (1xK)$ is the estimated likelihood that the

output tag belongs to each category. We'll go over how to adjust the weights V to make these predictions more accurate in the next segment.

In a fully linear setting, we saw how to define the given task. We discovered that by introducing a non-linear function (such as softmax), we could model more complex behavior. However, the true power of ANNs is discovered when we combine nonlinear functions that all have weight parameters and each combination is known as hidden layer which is described by the equation

$$h = \sigma(VX^T)$$

where the values of matrix U are the weights and the function σ is the activation function of the corresponding hidden layer.

As previously mentioned, softmax is an excellent choice for the output node for classification tasks. In general, however, any differentiable function can be triggered. Since non-linear behavior is frequently desired in activation functions, common activation functions also include rectified linear unit (RELU)[68] and hyperbolic tangent (tanh).

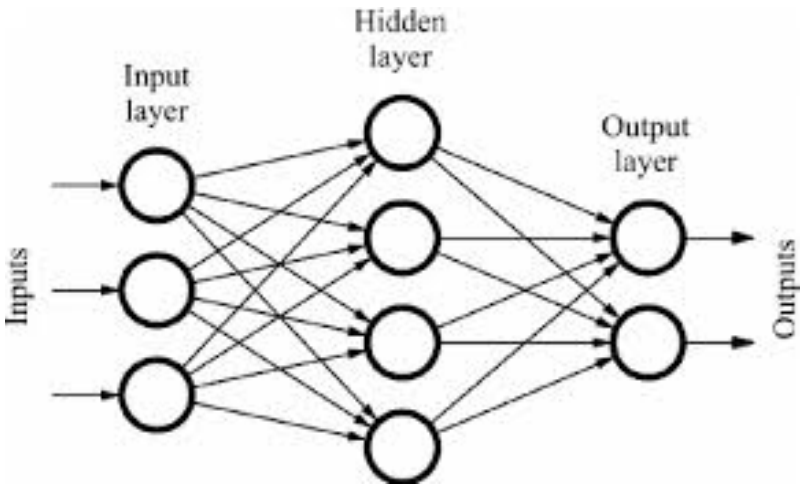

Figure 3.1: Neural Network

Figure 3.1 depicts a neural network. Since the input data rows are forwarded through the network to the output nodes, this form of neural network is known as a feed-forward neural network (FNN). We can potentially train the network to predict the likelihood of any input x_i corresponding to a category y_k, k=(1,...,5) if

we transfer the output layer through the softmax function. The following equation can be used to define this relationship.

$$\hat{y} = f(X;\omega) = \phi(\hat{V}h(X;\hat{V}))$$

It is obvious that $\hat{y}$ can be viewed as a function of the weights and the input data. As a result, we can adjust the weights to influence the predictions for any input x_i. In a classification framework, we can check the accuracy of these predictions by comparing them to our target vectors. If the weights in $\hat{U}$ and $\hat{V}$ are initialized at random, the predictions are likely to be incorrect. The weights will then need to be modified to yield more reliable results. This method, known as training or learning by gradient descent, is applicable to a wide range of network types.

3.2 Gradient

The basic concept behind training a neural network is to use an iterative gradient-based optimizer. To optimize, such an algorithm needs an objective function. That function is known as a cost function (also known as an error function or a loss function), and it is denoted by the letter J. Different cost functions are appropriate for various types of problems. The summed square error function, for example, is a reasonable option for regression models. In our case, the categorical cross-entropy function is used.

$$J(\omega) = -\sum_{i=1}^{n} \log(\hat{y}_i)$$

Given some parametrization ω and i=1,..,n training examples, y_i is the target class and $\hat{y}_i$ is the model's predicted probability of belonging to the target class. We may use J to calculate an update rule for our weights. We can measure the required derivatives to see if the cost will change if we modified the weights as long as the activation function in each layer is differentiable. Intuitively, we can descend towards the minimum of the cost function if we know the gradients of our optimization problem.

The scale of the step we take to descend in the opposite direction of the gradient is also important. The parameter that controls the phase size in a machine learning context is known as the learning rate, and it is pre-initialized. If it is too high, we risk surpassing the minimum. On the other hand, if it is too small, the algorithm can converge too slowly. Hyper-parameters are these types of parameters that are selected prior to training rather than as a result of training. The update rule for our weights is given by the equation

$$\omega^{t+1} = \omega^t - \eta \frac{\partial J}{\partial \omega^t}$$

where η is the learning rate and ω^t is the weights after time step t in the optimization process.

An epoch is completed when each training example has behaved on the model weights. Backpropagation is the process of updating the weight over several epochs. Gradient descent refers to the iterative method of weight updates.

Calculating the gradient for all weight updates is computationally infeasible when training sets are huge. As a result, the most common approach to model training is to update weights using a random subset of the training data. This method is known as stochastic gradient descent (SGD).

There is no guarantee of convergence when using SGD to train a neural network with a non-convex cost function. Convergence is affected by initial parameter values such as batch size, learning rate, and number of hidden layers. Choosing the right principles is a daunting task. Fortunately, as the field of deep learning has grown, variants of SGD have appeared. One such invention makes use of a dynamic learning rate to adjust our descent speed based on our previous momentum. This increases the likelihood of the algorithm's convergence.

One of the most prominent algorithms is called Adam (Adaptive moment estimation) and it learns based on the update rule below.

$$\omega^{t+1} = \omega^t - \frac{\eta}{\sqrt{\hat{v}_i} + \epsilon} \hat{m}_i$$

where $\hat{m}_t$ and $\hat{v}_t$ are bias corrected estimators of m_t and v_t given by

$$\hat{m}_t = \frac{m_t}{(1-\beta_1)^t}$$

$$\hat{v}_t = \frac{v_t}{(1-\beta_2)^t}$$

Where

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\frac{\partial J}{\partial \omega^t}$$

$$v_t = \beta_2 m_{t-1} + (1-\beta_2)(\frac{\partial J}{\partial \omega^t})^2$$

β_1 and β_2 are hyper-parameters (Ruder, 2016)[69].

3.3 Recurrent Neural Networks

Since we read words in order, language can be thought of as a sequence. As a result, modeling language using a FNN like the one shown above loses some valuable language knowledge about which words appear in which order in the series. Recurrent neural networks (RNNs), suggested by Elman (1990)[70] and others, take the sequential existence of the data into account when making output predictions.

RNNs achieve this by computing a collection of feedback weights in a hidden state vector at each time step in the series that transfer information from previous time steps. If we want to predict if a sequence belongs to a specific class, we can reformulate our model to integrate this new time dependence by making softmax probability predictions with the final recurrent hidden state vector.

$$\hat{y} = \phi(V h_p)$$

Where

$$h_p = \sigma(U x_p + W h_{p-1})$$

As a result, the model's parametrization ω is now made up of the weight-matrices U,

V, and W. In addition, our hidden layer is now reliant on previous states. Consider the following diagram to better understand this recurrence process.

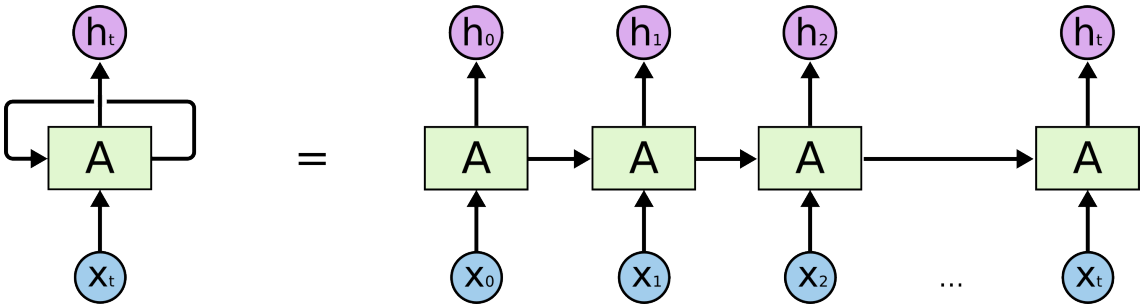

Figure 3.2: Recurrent neural network: The concept of unfolding through time [71].

A new hidden state vector is computed at each time level. Via gradient descent with an acceptable loss function, this vector can be trained to forward the most important information for solving the problem. To calculate the gradients of the weight matrices with respect to our loss function, however, we must unfold the network over time. If we have a classification problem, we can use the categorical cross-entropy function J to unroll our time dependence through the gradients.

Since we know that J is parametrized by U,V and W, we must compute the respective partial derivatives in order to compute the gradient updates. We can calculate this gradient by applying the chain rule to the various components of the weight update.

$$\frac{\partial h_p}{\partial h_p} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h_p} \frac{\partial h_p}{\partial h_1} \frac{\partial h_1}{\partial W}$$

However, because components of the gradient vector might grow or decay exponentially over lengthy sequences, RNNs with transition functions of this type will not converge optimally during training. The RNN model has a hard time learning long distance relationships in a sequence because of the exploding or vanishing gradients problem.

3.4 Long Short-Term Memory

To solve the problem of vanishing gradients, Hochreiter and Schmidhuber [72] [73], demonstrated how gating RNNs can learn long-term dependencies. We change

the cell structure since gradients vanish due to the continuous multiplication of partial derivatives. This is accomplished by adding elements to each recurrent layer, resulting in a memory for long sequences. The short-term memory capacities remain unchanged as compared to the simple RNN, resulting in the long short-term memory unit (LSTM).

From the basic RNN, LSTM networks incorporate two major innovations. To start, a hidden state vector and a local context vector are transferred to the next recurrent node at each time stage. Second, the LSTM network includes a collection of gating mechanisms that allow the model to determine which information to move forward in recurrence. This enables the LSTM model to learn long-term dependencies in the sequences more consistently.

Gates are a way of allowing information to move through if desired. They are built using a sigmoid layer and a pointwise multiplication operation. The sigmoid layer generates numbers ranging from 0 to 1, indicating how much of each section can pass through. A value of zero means "let nothing through," while a value of one means "let everything through." An LSTM contains three of these gates to secure and control the cell state.

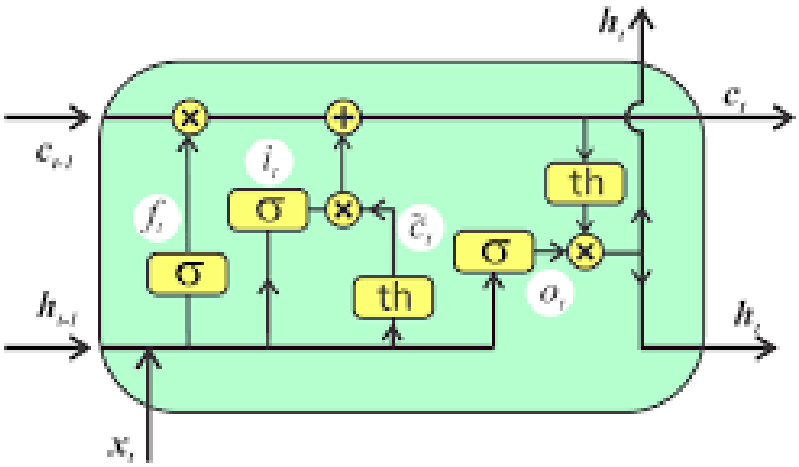

Figure 3.3: The LSTM architecture. [71]

The first step in the LSTM is to decide which information from the cell state will be discarded. A sigmoid layer known as the "forget gate" renders this option.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

The following step is to determine what new information the LSTM can store in the cell state. This is divided into two sections. To begin, a sigmoid layer known as the "input gate layer" determines which values will be changed. Following that, a tanh layer generates a vector of new candidate values, $\tilde{C}_t$, that could be applied to the state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_f \cdot [h_{t-1}, x_t] + b_C)$$

In the next step, these two are combined in order to produce an update to the state. Specifically, the old state is multiplied by f_t while disregarding the decisions made in the previous phase, and then it is added $i_t * \tilde{C}_t$.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Finally, the outcome is computed. This output is dependent on the cell state, but it has been filtered. A sigmoid layer is used to determine which parts of the cell state will participate in the output. The cell state is then passed through a tanh (to force the values to be between -1 and 1), and the result is multiplied by the sigmoid gate output.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

3.5 Bidirectionality

When modeling language, we must keep in mind that not all words are predicted by the word before them. Consider the sequences "river bank" and "bank account," where "bank" has a conceptual dependence not only on the previous term, but also on the word after. This is a problem that regular recurrent networks can encounter. However, Graves and Schmidhuber's(2005)[74] proposed bidirectionality can solve

this problem. This basically means reversing the sequence's course and feeding it to an independent network, which concatenates the resulting hidden states.

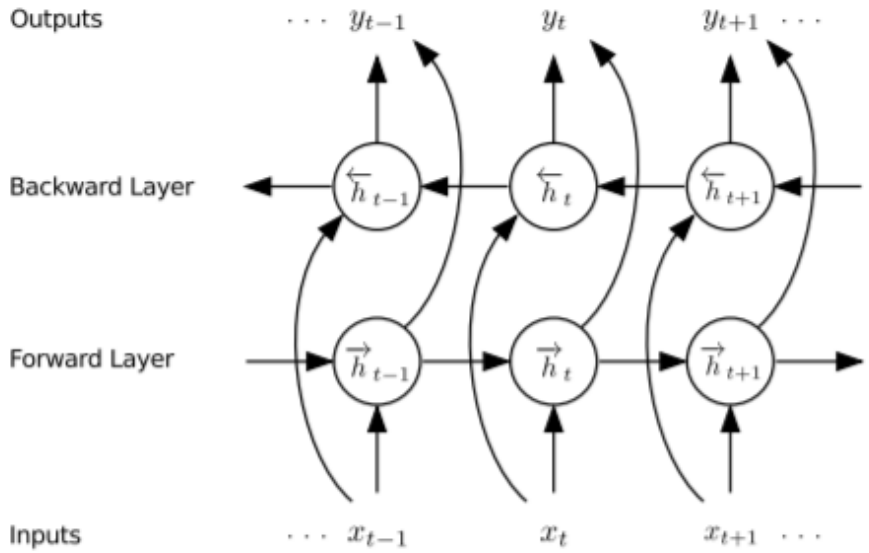

Figure 3.4: Bi-directional recurrent neural network [75]

The network with the reversed states is simply an independent copy of the original network. Commonly, the resulting hidden states are concatenated to form

$$h_t = (\overrightarrow{h_t}, \overleftarrow{h_t})$$

This final hidden state vector can then be used in modelling in the same way as for simple RNNs. We can now model bidirectional dependencies by doubling the number of weights. Constant scalar increases in complexity are normally not a concern in terms of computational capacity. Since it provides contextual knowledge to language models, it is often used to enhance model representation and predictions.

3.6 Vector Representation of Language

Machine and deep learning algorithms are unable to "read" data in the physical language that humans do. That is why we should represent the text in such a way that the model can interpret and process it. This is accomplished by representing the words in the corpus as vectors containing numbers, which are then used as input for the models. Word embeddings are a type of word representation technique used

in information retrieval and natural language processing.

The most common and straightforward approach for generating word embeddings is to interpret each sentence as a vector with a dimension equal to the total corpus vocabulary. If the word w appears in the sentence, the value at its vector location is set to 1, otherwise it is set to 0. If the same word appears in the text again, its value rises. Bag of Words (BoW) is a technique that can detect the frequency of words in a sentence. The number of times each word appears in the text is represented by the values that result for each word.

However, much of the time knowing the frequency of each word in a document is not as useful as knowing the importance of a word in the text. Although BoW weights all words solely on the number of times they appear, there is another approach that considers the sense in which the word occurs. Term frequency–inverse document frequency (Tf-Idf) is a technique that expresses the value of each word in a sentence in a series of sentences. Tf-Idf weights words not only by their frequency in a sentence, but also by their association with the rest of the dataset's sentences.

GloVe, as described in [76], captures the semantic relationships between words is a way to represent sentences and feed them into the model as input. A vector is used to represent each word in the corpus and each word is expressed by a GloVe-defined embedding of size 300. GloVe generates these vectors by counting the number of times each word appears. It minimizes the least square error (LSE) and attempts to differentiate the related terms using these statistics. GloVe differs from other word embeddings in that it not only considers simple co-occurrence probabilities, but also the ratio in which each word appears, as stated in [77]. The FastText embeddings [78], and the ConceptNet embeddings [79] are some other common embeddings models.

The choice of text corpus from which the model will learn is nearly as critical as the actual word representation model architecture. Since various types of language environments have completely different language features and vocabularies, this is extremely important. Training a word-embeddings model is clearly a computationally challenging job. Pre-trained word embeddings have been published because retraining these embeddings is challenging and computationally costly for all users.

These can be downloaded and used as inputs for NLP tasks in other machine learning models. However, since each NLP task uses different text data, the language worlds are slightly different. As a result, tuning the word-embeddings weights at the same time as training the model is standard practice. As a result, we can fine-tune the sense space of vectors that are appropriate for the issue. Also, certain words will still be missing from our word embeddings vocabulary. The most popular practice is to set these vectors of vocabulary words to random values. We can then train the word embeddings to sort these words into the appropriate part of meaning space based on the context in which they appear in the issue corpus.

3.7 Pre-training Language Model

The typical approach to solving a machine learning problem is to train a model from scratch using the task's training data. NLP is a broad field of study that includes a variety of tasks with limited sets of human-labeled training data. A significant amount of training data has been shown to enhance the performance of deep learning models, as shown by ImageNet[80] in the computer vision area. Deep learning NLP models can be used in the same way. A large amount of annotated text is used in the creation of a general-purpose language model, which is referred to as pre-training, and the language model's general purpose is to learn the contextual representation of words.

3.7.1 Language Model

Laws on how to use programming languages or formal languages may be specified. Natural languages vary from written languages in that they can have vague structures and words that can be used in ambiguous ways while still being understood by humans. Specifying and structuring a language using grammar can be challenging with words that alter their context.

When it comes to solving NLP problems, language models are crucial. They help with a form of language comprehension that helps it to predict the next or missing word in a text input by looking at the previous context. When training on text

data, a method of learning that includes understanding the word event and also the word prediction is used to be able to use meaning to predict words.

The language model learns meaning through techniques like word embedding, which uses vectors to represent words in a vector space. With a large amount of training data, the language model learns more word representations that are context-dependent, and enables similar words to have similar representations [81].

3.8 BERT

BERT [82], ULMfit [83], and ELMo [54] are examples of pre-training model architectures that have performed well in NLP tasks. The most recent pre-trained model, BERT [84], will be used exclusively in this thesis.

Pre-trained models are designed to learn a generalized language model by extracting features that can be used for other NLP tasks without requiring the model to be retrained. Transfer learning [85] is a type of training that enhances performance on other tasks that are connected to the pre-training task.

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained NLP model that considers both the left and right sides of a word's meaning [82]. Although the concept is straightforward, it achieves new state-of-the-art outcomes for a variety of NLP tasks, including sentiment analysis and question answering. BERT excels at tasks relevant to general language comprehension because it can fine-tune the BERT model to a particular task and because it can represent a word meaning from both left-to-right and right-to-left at the same time, allowing it to extract more context features of a sequence than ELMo [54].

3.8.1 Input Representation

The text input for the BERT model is first processed using a technique known as word-piece tokenization [86], which involves representing words as tokens(sub-word units) rather than whole words. As a result, we have a set of tokens, several of which represents a word. There are also two specialized tokens added to the set of tokens: a classifier token [CLS] at the start of the set and a separation token [SEP]

at the end of a phrase. If BERT is used to compare two sets of sentences, a [SEP] token will be used to distinguish them. This set of tokens is then transferred to the encoder layer after being processed through three separate embedding layers with the same dimensions that are later summed together:

- Token Embedding Layer: Each token in the input is mapped to a high-dimensional vector representation of the given token in this embedding.
- Segment Embedding Layer: This layer is used to isolate pairs of sentences, as one of the functions of BERT is to find relationships between them. This layer only has two representations: 0 for first-sentence tokens and 1 for second-sentence tokens.
- Position Embedding Layer: Since BERT employs Transformers, a position embedding layer is required to capture the tokens' sequential meaning. During the pre-training, BERT learned about the location embedding layer.

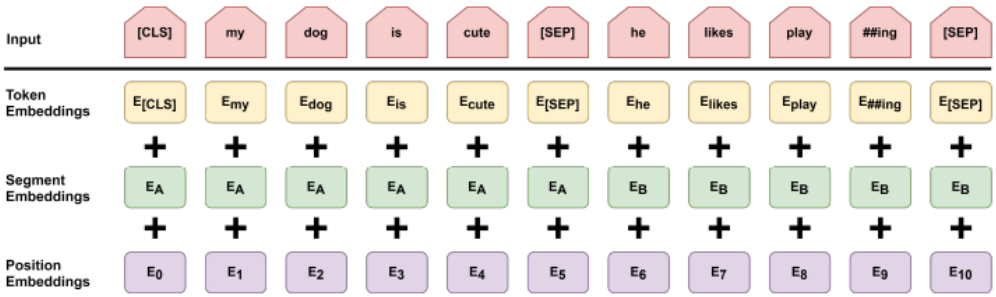

Figure 3.5: Tokenized input with embeddings [82]

3.8.2 Transformers

Previous studies used the sequence-to-sequence (seq2seq) method [87], as well as techniques like recurrent neural networks (RNN) [88] and long short-term memory (LSTM) [72], to define the state of the art in sequence modeling. Encoder-Decoder architecture is used in Seq2seq models, where the encoders map the input sequence into a high dimensional vector, which is then used as an input vector by the decoders, who convert the high dimensional vector to an output sequence.

Transformers' architecture is focused on attention-mechanics [89], which determine which parts are relevant in each computational step. The encoder not only converts the input to a high-dimensional space vector, but also adds the important keywords to the decoder's input. As a result, the decoder improves because it has more details, such as essential sequences and keywords that provide meaning to the sentence.

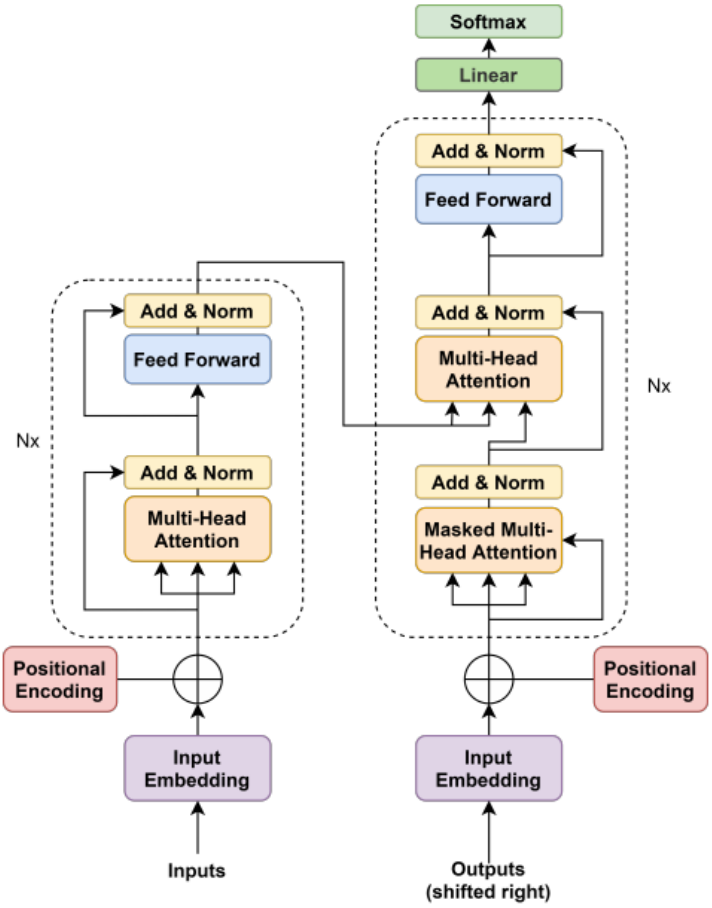

Figure 3.6: The Transformer architecture [89]

The transformers use self-attention layers rather than recurrent layers because self-attention layers have been shown to perform better in parallelization, owing to their ability to link all layer positions with a constant number of computational operations, while recurrent layers need a constant number of operations, making them faster than RNNs. The architecture of a Transformer is depicted in Figure 3.6, while the Transformer used in BERT only consists of the encoder. The Encoder is on the left, and the Decoder is on the right. Both the encoder and the decoder

are implemented using N_x modules, which are stackable.

Attention encoders include a multi-head attention layer and a feed forward neural network, while decoders include a masked multi-head attention layer, multi-head attention layer, and a feed forward layer. The multi-mead attention layer is focused on the attention function:

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

- Q are vector representation of the (query) one word in the sequence
- K are vector representation of all (key) words of (key,value) in the sequence
- V are vector representation of all (values) words of (key,value) in the sequence
- d_k are dimensions of the keys and query

The focus weights are determined by the effect of each word in the sequence (Q) on the other words in the sequence (K). After that, the weights apply the SoftMax function to all of the terms in the list (V). By repeatedly applying this attention-mechanism, the machine learns various representations of Q, K, and V. The Transformers, unlike the RNN, have no idea how the sequences are fed into the model, so they use the Encoder input-sequence, which includes the location from the Multi-Head Attention module. The Q, K, and V matrices for each location of the attention modules vary depending on whether the entire encoder input sequence or sections of the decoder input sequence are used.

3.8.3 BERT pre-training methods

The pre-training methods for BERT are designed to generate simplified features for bidirectional language representation. Furthermore, BERT demonstrates that by transferring the learned word representation and fine-tuning it to other tasks, this approach eliminates the need for feature-engineered task-specific architectures. Masked Language Model and Next Sentence Prediction are two pre-training tasks

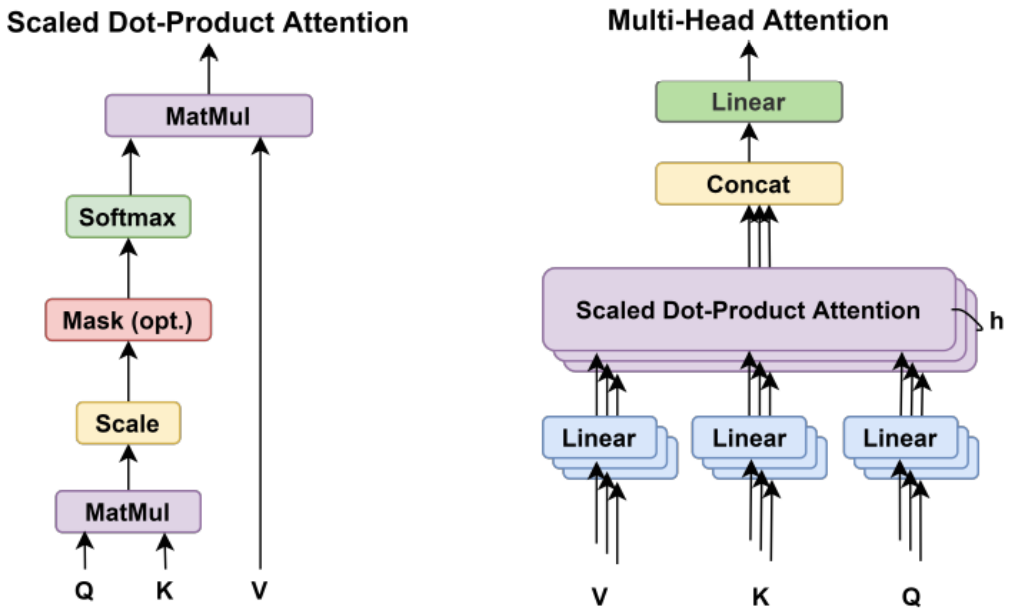

Figure 3.7: Scaled dot-product attention and multi-head attention [89]

that BERT contributes to in order to achieve a model that can be used in transfer learning.

3.8.3.1 Masked Language Model

The Masked Language Model (MLM) is a modified language model that uses masks to pre-train the language model BERT, which trains on both left and right sense. MLM's goal is to mask a random word in a sentence with a 15% chance of occurring; when the model masks a word, it replaces it with a token [MASK]. With the aid of transformers, the model then attempts to predict the masked word. But, unlike the traditional conditional language model, which teaches left-to-right or right-to-left word prediction and places the predicted word at the end or beginning of the text sequence, BERT hides a random word in the sequence. The other explanation for pre-training with a mask token is that the traditional conditional language model can only directly train left-to-right or right-to-left because the words can have the masked word, from left-to-right, unmasked in right-to-left, in a multilayered sense and thus know what the masked word should be.

3.8.3.2 Next Sentence Prediction

BERT, in addition to left and right context extraction using MLM, has a main goal that varies from previous work: next-sentence prediction. This improves BERT's semantic comprehension, which is used in fine-tuned tasks like question answering and natural language inference.

BERT has been pre-trained to predict whether or not there is a relationship between two text sentences in order to explain the relationship (IsNext or IsNotNext label) between Sentence A and Sentence B, separated by the token [SEP]. During preparation, sentence B is the half-time follow-up to sentence A and is used to predict the IsNext label. To predict the IsNotNext label, a random sentence is chosen for sentence B half of the time.

Due to the pre-training, BERT is designed as a general model that can be fine-tuned by adding an additional output layer on top of the Transformer [89] [87]. The additional layer does not need to be trained on a large number of parameters and therefore only requires limited training data for downstream tasks.

3.9 Evaluation metrics

We need a measure of model performance to know how well our model predicts. There are numerous methods for determining how effectively a categorization model predicts data. These figures will be examined further down.

The most intuitive metric, accuracy, is defined as the proportion of properly categorized samples. However, it is not the most helpful. In a case when class sizes are unequal, accuracy is dominated by the largest class, and the model may achieve a high accuracy score just by picking the larger class every time. Furthermore, we'd like to know how effectively our model distinguishes between each emotion class so we can figure out where it's going wrong. As a result, most model performance in a multi-class environment is assessed against other metrics such as Precision, Recall, and the F_β score.

When a model predicts that a data example belongs to a certain class, it is also forecasting that this data does not belong to the other k-1 classes. This means we

can speak about how many true and false positive (TP and FP) and true and false negative (TN and FN) predictions the model produced on the test data set for each class. This is referred to as the model's confusion matrix for this particular class.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_\beta = (1 + \beta^2) * \frac{Precision * Recall}{(\beta^2 * Precision) + Recall}$$

As a result, the Precision of a given class is the model's ability to distinguish between true and false positives. The model's recall measures how effectively it discovers all instances of a given class. F_β is therefore a harmonic mean of accuracy and recall, with Precision being weighted more than Recall in assessment if $\beta > 1$ and Recall being weighted higher than Precision if $\beta < 1$ [90]. This trade-off is situational and should be determined by the estimated cost of misclassification. However for our context these Precision and Recall metrics are balanced to create the $F_1 = 2 * \frac{Precision * Recall}{(\beta^2 * Precision) + Recall}$ score.

The model's ability to accurately categorize each class now includes a balanced metric for each class. We can now define the micro- Precision and Recall over all k classes for a multi-class data set.

$$micro - Precision = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k} TP_i + FP_i}$$

$$micro - Recall = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k} TP_i + FN_i}$$

$$micro - F_\beta = (1 + \beta^2) * \frac{micro - Precision * micro - Recall}{(\beta^2 * micro - Precision) + micro - Recall}$$

As a result, this score provides a method of balancing the model's performance across different classes during evaluation.

Chapter 4

Proposed Approaches

4.1 State-of-the-art ABSA Approaches

SemEval [1] is a series of international natural language processing (NLP) research workshops whose goal is to enhance the state-of-the-art in semantic analysis and to assist in the creation of high-quality annotated datasets in a variety of increasingly difficult natural language semantics challenges. Each year's workshop includes a set of shared objectives in which computational semantic analysis systems developed by various teams are presented and compared.

ABSA received a lot of interest as a study topic during the SemEval-2014 workshop, where it was first introduced as Task 4 and then reappeared at the SemEval-2015 and SemEval-2016 [91] workshops. The emphasis of this thesis is on ABSA and methods from the SemEval ABSA challenge[91] and specifically on the current state-of-the-art methods which are all organized in a repository in order to track the progress in this common NLP task.

The state-of-the-art models presented in SemEval workshops, and which are described in more detail in this chapter, are summarized in table 4.1 accompanied by their corresponding accuracy score on two datasets from different domains (a dataset of restaurant reviews and another with reviews about laptops)

[1] https://semeval.github.io/
[2] http://nlpprogress.com/english/sentiment_analysis.html

Subtask 2: Aspect term polarity results		
Model	Restaurant (acc)	Laptop (acc)
BERT-ADA (Rietzler, Alexander, et al., 2019) [58]	87.89	80.23
LCF-BERT (Zeng, Yang, et al., 2019) [57]	87.14	82.45
BERT-PT (Hu, Xu, et al., 2019) [92]	84.95	78.07
AOA (Huang, Binxuan, et al., 2018) [93]	81.20	74.50
TNet (Li, Xin, et al., 2018) [94]	80.79	76.01
RAM (Chen, Peng, et al., 2017) [52]	80.23	74.49
MemNet (Tang, Duyu, et al., 2016) [95]	80.95	72.21
IAN (Ma, Dehong, et al., 2017) [51]	78.60	72.10
ATAE-LSTM (Wang, Yequan, et al. 2016) [50]	77.20	68.70
TD-LSTM (Tang, Duyu, et al., 2016) [46]	75.63	68.13

Table 4.1: State-of-the-art ABSA Approaches performance[2]

4.1.1 TD-LSTM

The first model presented from Tang [46], called TD-LSTM, was a target-dependent extension on long short-term memory (LSTM) model. The LSTM model solves target-dependent sentiment categorization in a target-independent manner. That is, without taking into account the target words, the feature representation utilized for sentiment classification remains the same.

In this subsection, Tang made a minor adjustment to the aforementioned LSTM model and introduced a target-dependent LSTM (TD-LSTM) to account for the target information. The main idea was to describe the contexts surrounding the target string in both ways so that feature representations for sentiment classification can be employed in both directions as he believe that obtaining such target-dependent context information could help improve sentiment classification accuracy.

As shown in figure 4.1, in order to model the preceding and subsequent contexts,

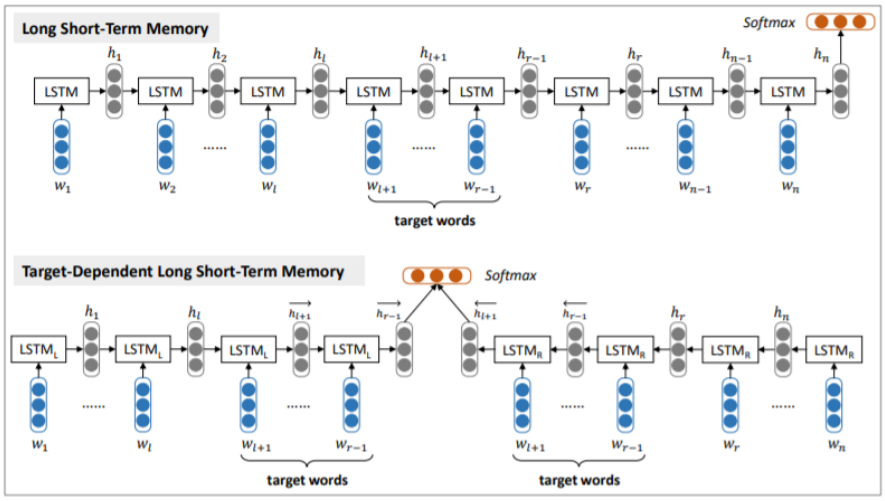

Figure 4.1: Architecture of TD-LSTM model [46]

they employ two LSTM neural networks, one on the left (LSTML) and one on the right (LSTMR). LSTML takes the preceding contexts plus the target string as input, while LSTMR takes the following contexts plus the target string as input. The last hidden vectors of LSTML and LSTMR are then concatenated and fed to a softmax layer to classify the sentiment polarity label.

4.1.2 ATAE-LSTM

TD-LSTM, on the other hand, can only examine the target, not the aspect information, which has been shown to be critical for aspect-level sentiment classification. Wang [50] introduced an attention strategy ATAE-LSTM (Attention-based LSTM with aspect embedding) in his research to force the model to pay attention to the most relevant element of a sentence in response to a certain attribute.

He suggested an aspect-to-sentence attention mechanism that focuses on the most important element of a sentence based on the aspect and proposed two methods for accounting aspect information during attention: one is to concatenate the aspect vector into the sentence hidden representations for computing attention weights, and the other is to append the aspect vector to the input word vectors. Figure 4.2 illustrates the approach of ATAE-LSTM.

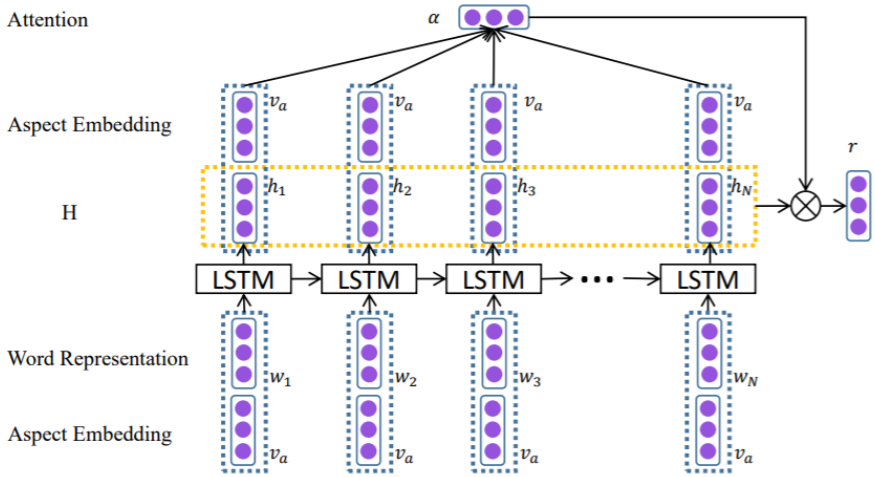

Figure 4.2: Architecture of ATAE-LSTM model [50]

u_a represents the embedding of aspect. The attention mechanism will produce an attention weight vector α and a weighted hidden representation r

$$M = tanh(\begin{bmatrix} W_h H \\ W_u u_\alpha \otimes e_N \end{bmatrix})$$

$$\alpha = softmax(\omega^T M)$$

$$r = H\alpha^T$$

and the final sentence representation is:

$$h^* = tanh(W_p r + W_x h_N)$$

4.1.3 IAN

Previous approaches, mainly focus on modeling the context based on a specific target and regard the aspect as an independent information. However, experiments shown that the aforementioned approaches were not so effective in polarity classification, and is absolute need to strengthen the effect of aspects in context representation. Jiang [36] into his work mention that 40% of errors are caused by not taking into account the target representation.

Ma and Wang [51], were the first who tried to model and create representations interactively for both context and aspect. Their model, IAN, is based on LSTM and learns simultaneously the features of context and target by applying an attention mechanism. This method tries to highlight the contribution of each token from context as well as from target which could be consisted from more than one words. IAN, makes use of the interactive information from context to supervise the modeling of target and vice versa and finally concatenate them to predict the sentiment polarity. The IAN model, as shown below , consists of two components which model the target and context interactively.

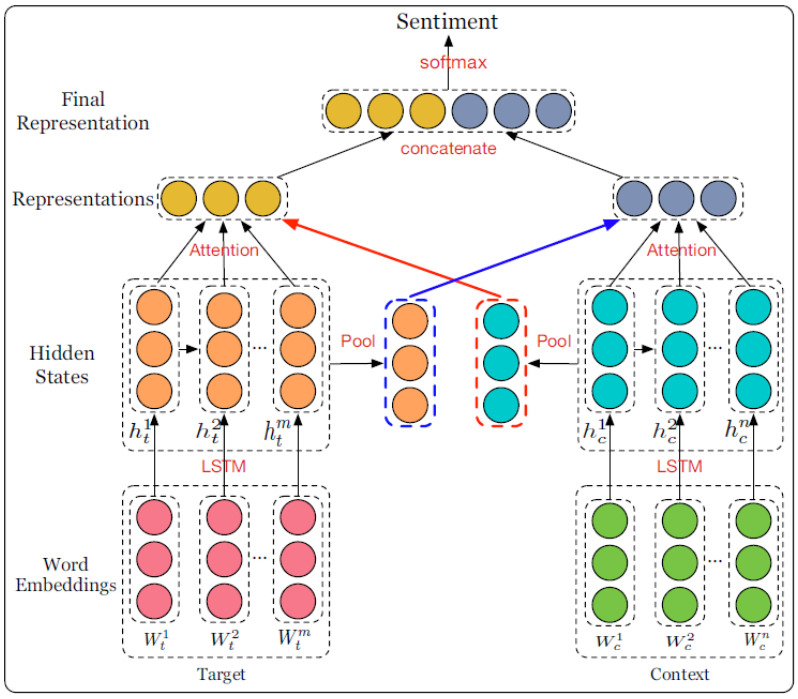

Figure 4.3: Architecture of IAN model [51]

With the help of two LSTM networks model learns the representations h_c and h_t, for context and target respectively, and uses them as initial representations by averaging the hidden states for the context, c_{avg}, and the target t_{avg} as below:

$$c_{avg} = \sum_{i=1}^{n} \frac{h_c^i}{n}$$

$$t_{avg} = \sum_{i=1}^{m} \frac{h_t^i}{m}$$

With these as input it introduces an attention mechanism to select the important parts of each representation by considering the influence from context to target and from target to context. For example, by considering the context word representations $[h_c^1,h_c^2,...,h_c^n]$ the attention vector α_i using target representation t_{avg} for context is:

$$\alpha_i = \frac{\exp(\gamma(h_c^i, t_{avg}))}{\sum_{j=1}^{n} \exp(\gamma(h_c^j, t_{avg}))}$$

Similarly the attention vector for the target based on the context representation c_{avg} is

$$\beta_i = \frac{\exp(\gamma(h_t^i, c_{avg}))}{\sum_{j=1}^{m} \exp(\gamma(h_t^j, c_{avg}))}$$

After that the target and context representations are calculated:

$$c_r = \sum_{i=1}^{n} \alpha_i h_c^i$$

$$t_r = \sum_{i=1}^{m} \beta i h_t^i$$

4.1.4 MemNet

Tang [95], inspired by the memory networks that were used with absolute success in questioning answering task, designed a deep memory network with reporting multiple computational layers in order to perceive the importance degree of each context word when gathering the sentiment polarity of as aspect.

Unlike, LSTMs which are designed to work in a sequential manner and are incapable to capture the important parts of a context, Tang tried to design a model that would work like a human brain. When a human is asked to find the sentiment regarding an aspect he will explicitly focus on parts of the contexts and gathers clues that are needed to build up the representation regarding this aspect in his mind. Having said that, Tang designed a deep memory network where each layer is a content and location based attention model that captures the important clues and

uses this knowledge to build continuous text representations in order to conclude to the final features. An overview of deep memory network approach is illustrated in the following figure.

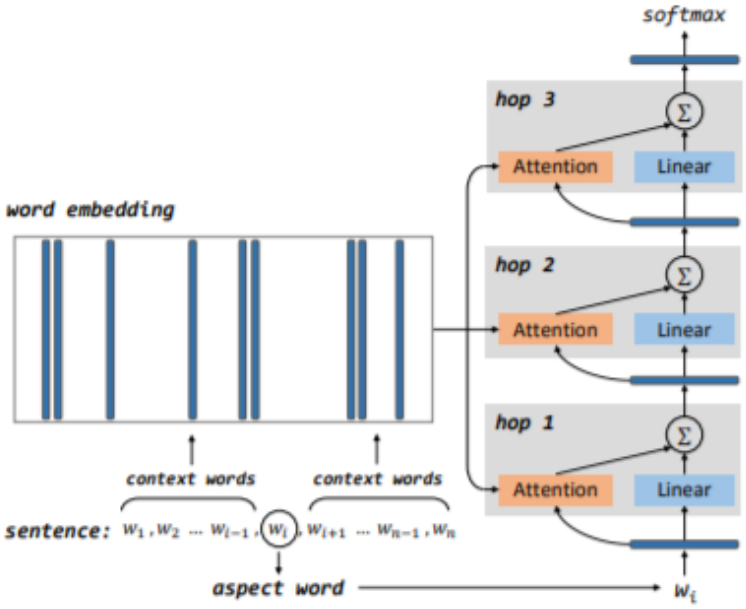

Figure 4.4: Architecture of MemNet model [95]

As shown in figure 3.4, the embedding vectors of context words are stacked and considered as the memory component. After that, the model includes multiple attention layers, called hops. The usage of multiple hops is helpful to learn complex text representations with multiple levels of abstraction.

At the first hop, the model regards the aspect embedding vector as input to obtain the relationship weight from context to target from memory. The sum of this attention layer and a linear transformation of aspect vector is the input of next attention layer. Each attention layer is a neural network based model that computes the weighted representation of each memory slice towards to the aspect as:

$$vec = \sum_{i=1}^{k} \alpha_i m_i$$

where the final importance scores $\alpha_1, \alpha_2, ..., \alpha_k$ are calculated from a feed forward neural network and m_i is a piece of memory which takes into account the location information between aspect and context word and is calculated by the context

embedding and the distance of the context word from the aspect in the sentence sequence.

4.1.5 RAM

Chen in his work [52] introduces a multilayer architecture that affiliates a combination of multiple attentions in order to exploit information from features based on their position in a sentence and their distance from the target.

This strategy was introduced in earlier models. However, the difference regarding the previously suggested models is that ATAE-LSTM adopts only one attention step. MemNet, although applying a multiple attention mechanism, uses the output of each attention step as input to the next attention step. As a result, the final output is a linear combination of the initial input representation. Instead, in RAM the outputs from each attention are incorporated with a GRU network.

The structure of RAM model consists of 3 main components as shown below:

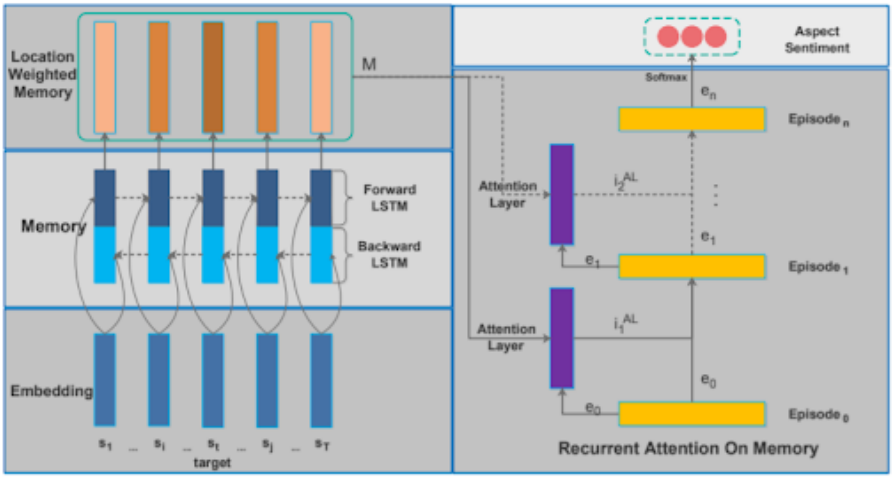

Figure 4.5: RAM model [52]

The first component is the Memory block which contains two Bidirectional LSTMs. The cell state of each hidden cell works as a memory for the whole model. The next block is the Location Weighted Memory. In order to gain the most relevant information from memory states regarding a specific target, the model gives bigger weights to context words or memory slices that are closer to the aspect, and the closeness of each word is calculated as the amount of tokens that separate it

from the target. However, this step highlights the closer opinion, they introduce an attention mechanism in order to capture information from opinion words that are not close to the target aspect. The third and most important component of the RAM model is the Recurrent Attention on Memory module. At this step, they apply multiple attentions on the memory block in order to export more accurate information from opinion words located in different positions from the target.

The outputs of attention steps are combined with Gated Recurrent Units (GRUs). The process of updating the state of each gate is described through the following formulas:

$$r = \sigma\left(W_{ri_t}^{AL} + U_r e_{t-1}\right)$$

$$z = \sigma\left(W_{zi_t}^{AL} + U_z e_{t-1}\right)$$

$$e_t^{\sim} = \tanh\left(W_{xi_t}^{AL} + W_g\left(r \odot e_{t-1}\right)\right)$$

$$e_t = (1_z) \odot e_{t-1} + z \odot e_t^{\sim}$$

where e_{t-1} is the state of the previous time step and ${i_t}^{AL}$ is the output of the current attention step. Each attention layer takes as input the memory slices and the previous state of the GRU network. The normalized attention score, g_j^t, of a memory slice m_j computed by the function,

$$\alpha_j^t = \frac{\exp\left(g_j^t\right)}{\sum_k \exp\left(g_k^t\right)} \quad where \quad g_j^t = W_t^{AL}(m_j, e_{t-1}[, u_T]) + b_t^{AL}$$

and the total output of a particular attention layer for the whole memory is :

$$i_t^{AL} = \sum_{j=1}^{T} \alpha_j^t m_j$$

The output of the final GRU is the input of a softmax layer which is used for sentiment classification.

4.1.6 TNet

TNet (Target-Specific Transformation Network) proposed by the authors of [94] comes to overwhelm the handicaps of attention mechanism and CNN in classification tasks. Previous works based on Recurrent Neural Networks (RNNs) with attention mechanisms try to compute the pairwise relations between the aspect and each context word. However, these approaches suffer from noise, which is introduced in the model when by mistake it gives a high attention weight to unrelated opinion modifier words, which leads to lower prediction accuracy.

In contrast with methods that already exist, TNet is bringing in a pioneering Target-Specific Transformation (TST) component which instead of computing the attention score between the target and the averaged context vector, first creates a new dissimilar target representation based on each separate context word and then computes the prominence of target words based on each context token.

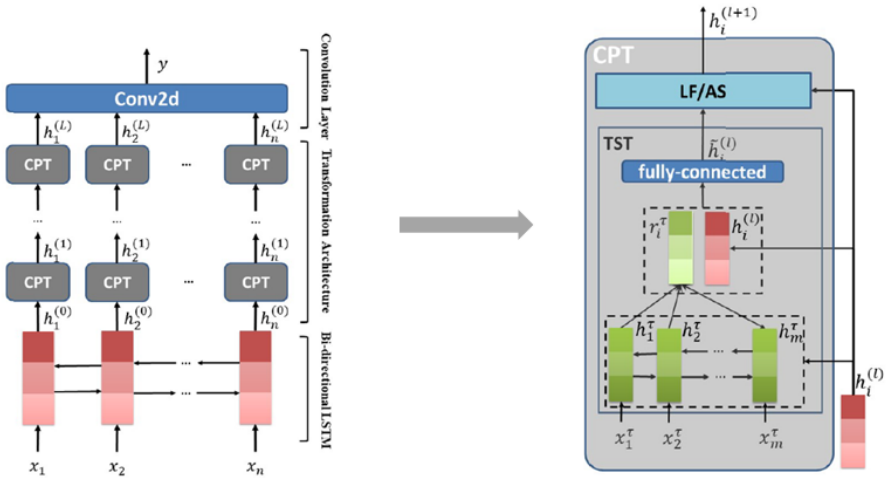

Figure 4.6: TNet model [94]

The architecture of their model includes three main components:

The first layer is a Bidirectional LSTM (Bi-LSTM) which generates the word representations of inputs.

The second, and most important component, consists of L Context-Preserving Transformation (CPT) layers. As shown in figure a CPT layer contains the "TST" block and a "LF/AS" block. As mentioned previously, TST component generates

the target word representations for the m possible tokens of target phrase with a Bi-LSTM:

$$h_j^T = [\overrightarrow{LSTM(x_j^T)}; \overleftarrow{LSTM(x_j^T)}]$$

After that, it finds out the relations among these representations and each context word representation from first layer:

$$r^T = \sum_{j=1}^{m} h_j^T * F(h_j^{(l)}, h_j^T)$$

where F measures the association between the j-th target word representation h_j^T and the i-th word level representation $h_j^{(l)}$:

$$F(h_j^{(l)}, h_j^T) = \frac{\exp(h_j^{(l)T} h_j^T)}{\sum_{k=1}^{m} \exp(h_j^{(l)T}, h_k^T)}$$

And finally generates the transformed representation for each word which gathers specific information from each target word:

$$\tilde{h}_i^{(l)} = g(W^T[h_i^{(l)} : r_i^T] + b^T$$

The transformation that takes place in TST is non-linear but is important to keep also the initial context information captured from the first layer. In order to achieve that they propose two procedures:

- Lossless Forwarding plan which promotes the initial representations as input to each next layer with the output of previous TST block:

$$h_i^{(l+1)} = h_i^{(l)} + \tilde{h}_i^{(l)}$$

however, with this approach we are not sure if the model will assign the weights to initial context features and transformed features correctly. So the second strategy

- Adaptive Scaling works like a gate which inspects how much information from

the transformed and initial features will pass through as follows:

$$h_i^{(l+1)} = t_i^{(l)} \odot \tilde{h}_i^{(l)} + (1 - t_i^{(l)}) \odot h_i^{(l)} \quad where \quad t_i^{(l)} = \sigma(W_{trans} h_i^{(l)} + b_{trans})$$

The third layer is a CNN layer accompanied with a closeness weight u_i. As a sentence may contain opinion words that not all refer to the target aspect, it is useful to find which one should be used as modifier for the classification task. So they suggest a score that takes higher value for opinion words that are closer to the target.

$$\hat{h}_i^{(l)} = h_i^{(l)} u_i \quad , i \in [1, n], l \in [1, L]$$

$$u_i = \begin{cases} 1 - \frac{(k+m-i)}{C} & , i < k + m \\ 1 - \frac{(i-k)}{C} & , k + m \leq i \leq n, \\ 0 & , i > n \end{cases}$$

where k is the index of the first target word, m is the length of the target and C is a pre-specified constant. The weighted input of CNN layer is:

$$c_i = ReLU(w_{conv}^{(T)} h_{i:i+s-1}^{(L)} + b_{conv})$$

and the final output which contains the most informative features is exported with max-pooling layer.

4.1.7 AOA

Attention-over-Attention (AOA) neural network proposed by the authors of [93] learns the representations for aspects and sentences simultaneously and explicitly captures the interaction between aspects and context sentence representations generated from LSTMs.

After the consideration that only a part of tokens in a sentence is important to gather the sentiment polarity about an aspect, they introduced an attention

mechanism that mutually creates attentions from aspect to context and vice versa in order to capture and focus on the tokens with the higher correlation to each other between aspect and text. The architecture of their model includes four components.

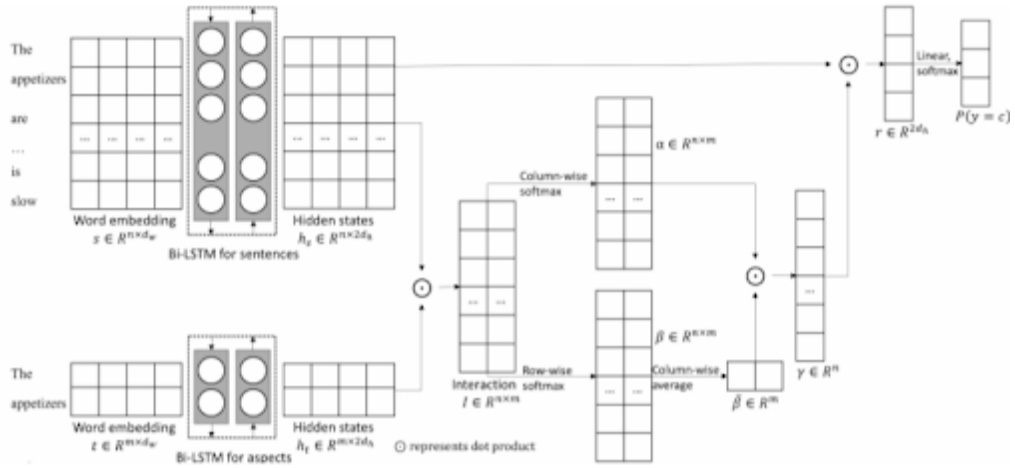

Figure 4.7: AOA model [93]

The first is word embeddings where the text sentence and aspect target are converted into its numerical representations. The second layers consists of two Bidirectional LSTMs (Bi-LSTMs) which are fed with the generated word vectors from embeddings in order to learn the hidden semantics of words in the sentence and aspect target. The third and main component of the model is the Attention-over-Attention module. In this step the attention weights are computed by an AOA module that includes the following steps:

- Compute a pair-wise interaction matrix between the two hidden states, where the value of each entry represents the correlation of a word pair between sentence and aspect.

- With column-wise softmax learn α, target-to-sentence attention.

$$\alpha_{ij} = \frac{\exp(I_{ij})}{\sum_i h_j^T \exp(I_{ij}}$$

- With row-wise softmax to get β, sentence-to-target attention.

$$\beta ij = \frac{\exp(I_{ij})}{\sum_i h_j^T \exp(I_{ij}}$$

- Calculate the column-wise average of β to get a target-level attention $\overline{\beta}$, which tells us the important parts in an aspect target.

$$\overline{\beta_j} = \frac{1}{n} \sum_i \beta_{ij}$$

- The final sentence-level attention γ is the weighted sum of each individual target-to-sentence attention α.

$$\gamma = \alpha \overline{\beta^T}$$

The last component is Final Prediction, where the output representation is a weighted sum of sentence hidden states using sentence attention from AOA module.

$$r = h_s^T \gamma$$

This final sentence representation is feed into a linear layer with a softmax function to output probabilities of sentiment classes. The sentiment class with the highest probability is the predicted label for the sentence, given the aspect target.

4.1.8 BERT

BERT-PT proposed by the authors of [92] suggests a novel post training technique for question answering (QA) task, called Review Reading Comprehension, based on the pre-trained BERT language model. Their approach can also be applied for aspect-based sentiment analysis tasks such as aspect extraction and aspect sentiment classification.

The challenge they try to tackle is that BERT on its own has specific neither domain knowledge, as it is trained on Wikipedia articles and cannot understand properly opinion words, nor task knowledge, as BERT is designed to be task-agnostic. In this paper they extend BERT with a task-specific layer and fine-tune it on specific end tasks.

They use the pre-trained weights from BERT as initializer and they gain knowl-

edge from unsupervised domain review datasets and supervised task-specific datasets before BERT fine-tuning in order to increase both the domain and task awareness. In order to fuse domain knowledge, they leverage the key objectives from BERT: masked language model(MLM) and next sentence prediction(NSP). For task knowledge they use the SQuAD dataset. The algorithm below describes a step of the post-training procedure:

$\nabla_\Theta \mathcal{L} \leftarrow 0$
$\{\mathcal{D}_{\text{DK},1}, \ldots, \mathcal{D}_{\text{DK},u}\} \leftarrow \text{Split}(\mathcal{D}_{\text{DK}}, u)$
$\{\mathcal{D}_{\text{MRC},1}, \ldots, \mathcal{D}_{\text{MRC},u}\} \leftarrow \text{Split}(\mathcal{D}_{\text{MRC}}, u)$
for $i \in \{1, \ldots, u\}$ **do**
  $\mathcal{L}_{\text{partial}} \leftarrow \frac{\mathcal{L}_{\text{DK}}(\mathcal{D}_{\text{DK},i}) + \mathcal{L}_{\text{MRC}}(\mathcal{D}_{\text{MRC},i})}{u}$
  $\nabla_\Theta \mathcal{L} \leftarrow \nabla_\Theta \mathcal{L} + \text{BackProp}(\mathcal{L}_{\text{partial}})$
end
$\Theta \leftarrow \text{ParameterUpdates}(\nabla_\Theta \mathcal{L})$

Figure 4.8: BERT-PT Post training Algorithm [92]

BERT-ADA presented by the authors of [58] is, in essence, an extension of the aforementioned model. In this work they try to explore how BERT behaves when is fine-tuned on aspect sentiment classification task by transferring knowledge across different domains. Their approach consists of two steps. In the first step they fine-tune the pre-trained weights of BERT on a domain dataset and in the second step they train the fine-tuned model on aspect sentiment classification task. The difference in this paper is that the try to find out how the performance of fine-tuning is related to the number of training steps and how this approach can be generalized by evaluating the performance across different domain from the one used for fine-tuning procedure, a method named Domain Adaptation. The framework of this method is concluded in the following pipeline:

$$D_{LM} \rightarrow D_{Train} \rightarrow D_{Test}$$

where D_{LM} represents the domain on which the first step of their method (model fine-tuning) is applied and D_{Train},D_{Test} stand for the training and testing domain respectively which take place in the second step.

4.1.9 LCF-BERT

Almost no prior research took into account the essential emotional information included in an aspect's local context. Traditional DNN-based techniques, for example, simply look at the correlations between global context and sentiment polarities before determining the sentiment polarity of a specific aspect based on global context information.

However the authors of [57] detected that an aspect's sentiment polarity is more important to context words next to it and introduced the LCF-BERT model, which uses self-attention to collect both local and global context information at the same time. To estimate sentiment polarity of a targeted element, LCF design models integrate local and global context data. The overall architecture of LCF-BERT design is illustrated in figure 4.9

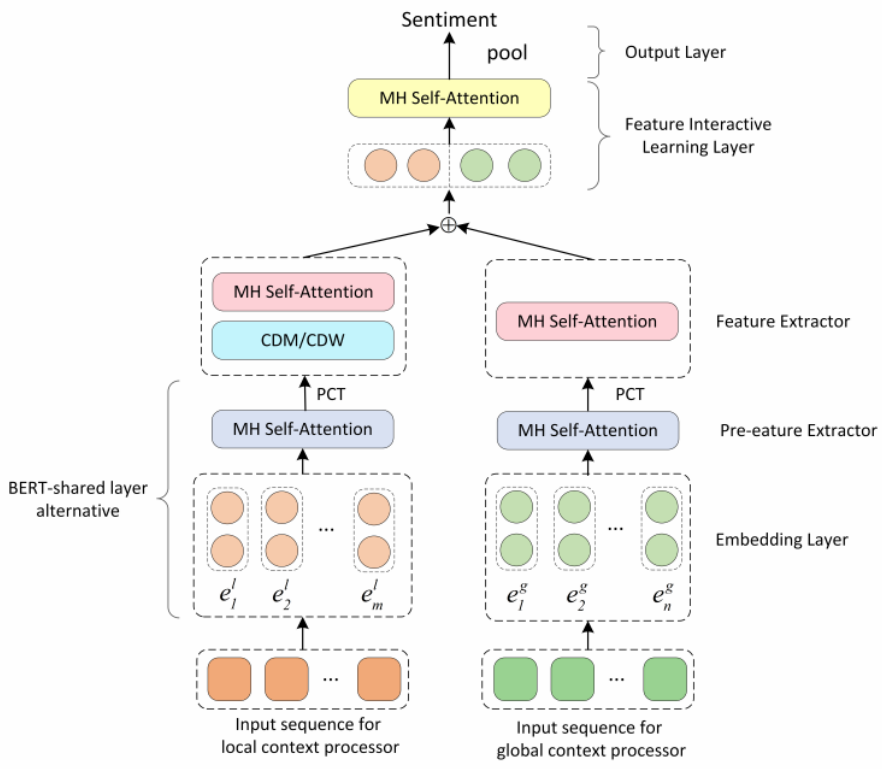

Figure 4.9: LCF-BERT model [57]

BERT-shared layer is alternative to substitute for GloVe word embedding layer and Pre-Feature Extractor layer.

The first layer of LCF-BERT is embedding layer which consists of two independent BERT-shared layers to model the input features of local context sequence, X^l, and global context, X^g, respectively.

$$O^l_{BERT} = BERT^l(X^l)$$

$$O^g_{BERT} = BERT^g(X^g)$$

where index l stands for local context and index g for global context.

A Feature Extractor (FE) layer is used in the LCF architecture to learn characteristics of the local and global contexts. It would unavoidably miss the properties of less-semantic-relative context words if it just considered the local context. LCF models use global context characteristics as a supplement to strengthen LCF design in order to completely preserve the features included in the global context and understand the association between global context and aspect. The local context feature extractor is differentiated from the global context feature extractor by the presence of a local context focus layer, whereas the global context feature extractor simply has an MHSA encoder.

For the local context focus layer they implement two architectures to focus on local contexts, Context Dynamic Mask (CDM) and Context Dynamic Weighted (CDW). The properties of semantic-relative contextual words are preserved in LCF design layers, whereas the features of less-semantic-relative contextual words are masked or weighed. As a result, the less semantic-relative surroundings can take part in the encoding process, and their passive impact is reduced.

The less-semantic-relative contextual words are considered based on the Semantic-Relative Distance (SRD) of all token-aspect pairs which is the count of the tokens between each contextual token towards specific aspect. So, the features on each position above the SRD threshold will be masked or weakened.

• With the CDM layer deployed, only the features of the less-semantic-relative context itself on the corresponding output position will be masked. All masked features will be set to zero vectors. CDM focuses on the local context by constructing

the mask vectors V_i^m.

$$V_i = \begin{cases} E & , SRD_i \leq \alpha \\ O & , SRD_i > \alpha \end{cases}$$

$$M = [V_0^m, V_1^m, ..., V_n^m]$$

$$O_{CDM}^l = O_{BERT}^l \cdot M$$

• With the CDW layer deployed, the less-semantic-relative context features will be weighted decay. CDW weights the features by constructing the weighted vector V_i^w.

$$V_i = \begin{cases} E & , SRD_i \leq \alpha \\ \frac{SRD_i - \alpha}{n} \cdot E & , SRD_i > \alpha \end{cases}$$

$$W = [V_0^w, V_1^w, ..., V_n^w]$$

$$O_{CDW}^l = O_{BERT}^l \cdot W$$

where α is the SRD threshold. M is the mask matrices for the representation of input sequences and n is the length of input sequence including aspect. $E \in R^{d_h}$ is the ones vector and $O \in R^{d_h}$ is the zeros vectors.

After the local context focus layer, another MHSA encoder is deployed to learn the masked/weighted context features.

$$O^l = MSHA(O_{CDM}^l) \ \ or \ \ O^l = MSHA(O_{CDW}^l)$$

and

$$O^g = MSHA(O_{BERT}^g)$$

After that, the Feature Interactive Leaning (FIL) layer is used to learn the characteristics of the global and local context in an interactive manner. FIL first concate-

nates the representations of O^l and O^g , then projects them into O^{lg}_{pool} and applies an MHSA encoding operation.

$$O^{lg} = [O^l; O^g]$$

$$O^{lg}_{dense} = W^{lg} \cdot O^{lg} + b^{lg}$$

$$O^{lg}_{FIL} = MSHA(O^{lg}_{dense})$$

The representation obtained by the feature interactive learning layer is pooled in the output layer by extracting the hidden states on the first token's corresponding location. Finally, a Softmax layer is used to predict the polarity of sentiment.

$$X^{lg}_{pool} = POOL(O^{lg}_{FIL})$$

$$Y = Softmax(X^{lg}_{pool})$$

4.2 Our Approach: LCF-OP BERT with lexicon

All previous approaches take into account the content, global or local, around the targeted aspect and the relationship between it and each word in order to predict the polarity regarding the aspect. However none of them consider the opinion words which may be exist in the content.

Because opinion words usually co-occur with aspect words, one of the most essential qualities is that they can give suggestive indications for aspect polarity classification. Our approach extends LCF BERT design as adopts the method of local context processor described in Section 4.9 and introduces a third branch which highlights opinion words which are in the aspect's local context. As in LCF BERT, the SRD threshold is used to limit the local context and the CDM feature extractor masks all the other tokens except from the opinion words.

Due to the fact that the training data lacks ground truth for opinion words, sentiment lexicon used to identify possible opinion terms. The overall architecture of our method, LCF-BERT design with lexicon, is illustrated in figure 4.10 where

the extension of the LCF BERT model is included in the red frame

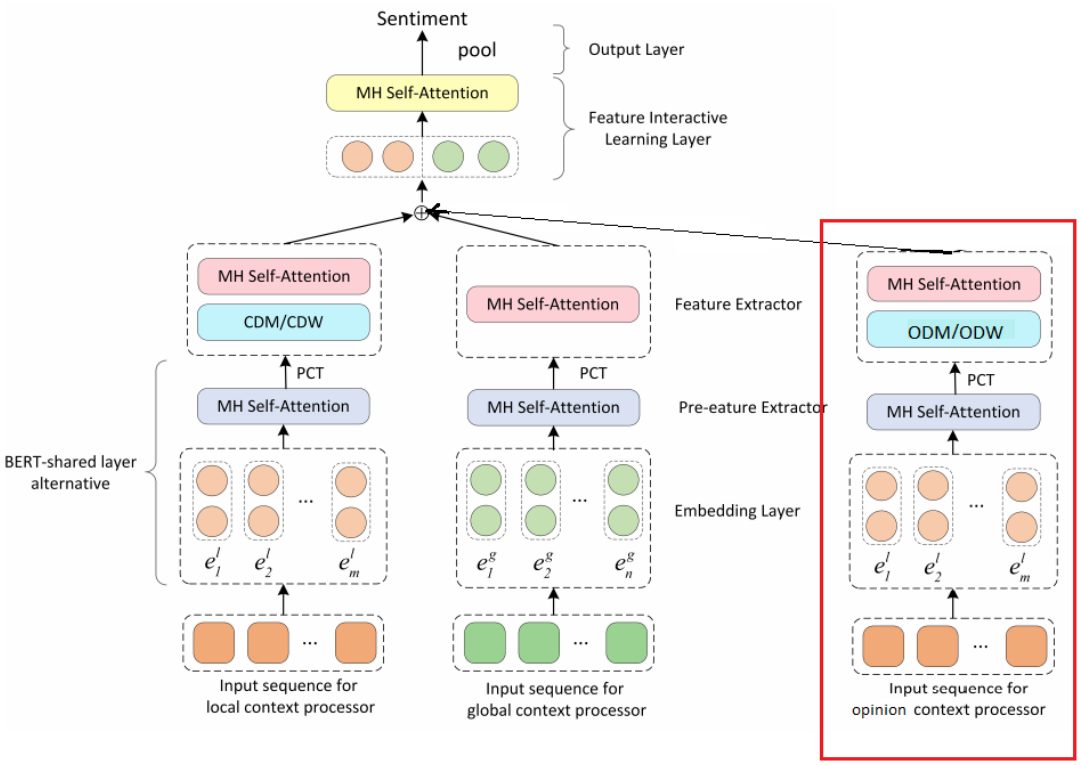

Figure 4.10: LCF-OP BERT model with lexicon

The main addition in LCF-BERT with lexicon is the opinion context generator which is similar to the local context generator. The diffirence here is that instead of the local context focus layer we used an local opinion focus layer with which the non opinion features of the local context will be masked.

$$V_i = \begin{cases} E & , SRD_i \leq \alpha \\ O & , SRD_i > \alpha \end{cases}$$

$$Op = [V_0^{op}, V_1^{op}, ..., V_n^{op}]$$

$$O_{OpDM}^{op} = O_{BERT}^{op} \cdot Op$$

and the Feature Interactive Leaning (FIL) layer concatenates the representations of O^l , O^g and O^{op}

Chapter 5

Methodoly and Experiments

The aim of this chapter is to explain the overall approach used in the experiments. It goes through the dataset, the data pre-processing that was done, and the creation of the train, development, and test sets that the models will use. Finally, the chapter discusses the models created, as well as the hyper-parameter tuning and evaluation metrics employed.

5.1 Dataset

The hotel reviews were gathered from the Greek version of Tripadvisor, one of the world's most popular travel sites[1]. There are 1,800 reviews in our data set (900 positive and 900 negative). The reviews that were translated into Greek were not considered because they contained grammatical and syntactic errors.

The dataset does not contain the annotation for aspect terms and their corresponding polarity. However the scope of this thesis is to study the sentiment related to each aspect included in the whole review. As a result the first step is to identify, manually, the aspects which a review may contains and then to annotate them with the corresponding sentiment polarity based on the opinion expressed in the review about each one of them.

[1] www.tripadvisor.com.gr

5.1.1 Data Annotation with CLARIN-EL

The process of labeling data available in various formats such as text, video, or images is known as data annotation. Labeled data sets are required for supervised machine learning so that the machine can easily and clearly understand the input patterns.

Data annotation task in this work consists of two subtasks. The first one is to identify all aspect terms present in each sentence given a set of review sentences. The aspects in hotel reviews could be nouns that describe features, properties and services of a hotel like 'room', 'breakfast', 'pool', 'staff', 'cleanliness' and so on. The second subtask is the sentiment annotation. The technique of assigning precise sentiment values to a given remark is known as sentiment annotation. A polarity label required to be assigned to each recognized aspect term. This process now considers three values: positive, negative, and neutral, but it can be expanded to a larger set.

The annotation tool we used for the aforementioned annotation subtasks is a new Web-based annotation tool called "CLARIN-EL Web-based Annotation Tool"[2]. Based on an existing annotation infrastructure provided by the "Ellogon" language engineering framework, this new tool uses cloud computing to migrate a large portion of Ellogon's features and functionalities to a Web environment [96].

The user has the ability to upload and manage the documents he/she wants to annotate. After the user has established or managed his or her collections, he or she can go to the "Annotation" page to annotate documents from the collections that are available. Once on the annotation page, the user must choose an annotation schema. In this work used a schema called 'aspect+polarity', as we wanted to annotate simultaneously the aspect and its corresponding polarity. An example of the annotation process is shown in the figure 4.1. In this review, there are four aspects (΄δωματια΄ and ΄φαγητο΄) highlighted in the text. In the tab on the right are shown the start and the end position of each aspect as well as the three possible polarity labels from which the user should choose the corresponding for each specific

[2]https://ellogon.org/clarin/welcome

annotated aspect by clicking on the desired component (in our example positive for the aspect ΄δωματια΄ and negative for the second aspect ΄φαγητο΄)

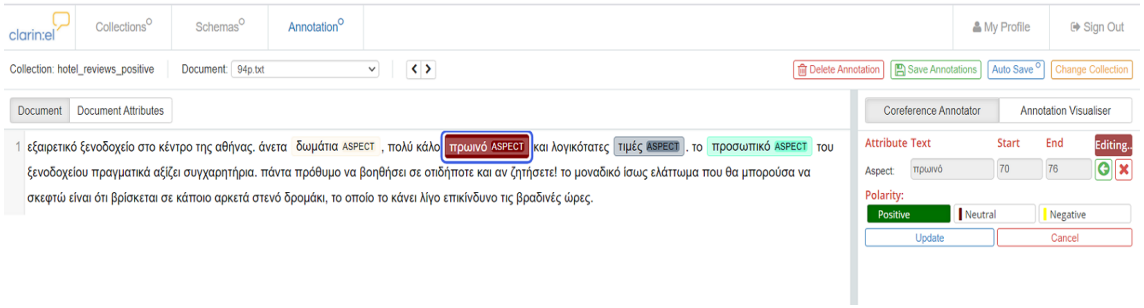

Figure 5.1: CLARIN-EL Annotation tool UI for annotating a review.

The whole dataset is exported as a json file where each review is an object with its corresponding elements. An example from the exported file for the aforementioned review document is illustrated in figure 5.2. Each highlighted aspect, is handled as a separated element.The values of this element contains the start and end position of the aspect and the corresponding polarity given for this aspect from the annotator.

```
{'id': 309,
 'name': '95p.txt',
 'text': 'εξαιρετικό ξενοδοχείο στο κέντρο της πόλης. φιλικό περιβάλλον, άψογη εξυπηρέτηση, ευρύχωρα και άνετα δωμάτια
          πλούσιο πρωινό. μοναδικό του ίσως μειονέκτημα, το δύσκολο πάρκινγκ λόγω των πεζόδρομων,
          αλλά το ξεχνάς μόλις περάσεις την πόρτα του ξενοδοχείου!',
 'encoding': 'UTF-8',
 'annotations': [
  {'spans': [{'segment': 'περιβάλλον', 'start': 51, 'end': 61}],
   'attributes': [{'name': 'aspect', 'value': '51 61'},
    {'name': 'opinion_polarity', 'value': 'positive'}]},
  {'spans': [{'segment': 'εξυπηρέτηση', 'start': 69, 'end': 80}],
   'attributes': [{'name': 'aspect', 'value': '69 80'},
    {'name': 'opinion_polarity', 'value': 'positive'}]},
  {'spans': [{'segment': 'δωμάτια', 'start': 101, 'end': 108}],
   'attributes': [{'name': 'aspect', 'value': '101 108'},
    {'name': 'opinion_polarity', 'value': 'positive'}]},
  {'spans': [{'segment': 'πρωινό', 'start': 118, 'end': 124}],
   'attributes': [{'name': 'aspect', 'value': '118 124'},
    {'name': 'opinion_polarity', 'value': 'positive'}]},
  {'spans': [{'segment': 'πάρκινγκ', 'start': 168, 'end': 176}],
   'attributes': [{'name': 'aspect', 'value': '168 176'},
    {'name': 'opinion_polarity', 'value': 'negative'}]}]}
```

Figure 5.2: Example of exported annotated file from CLARIN-EL

5.1.2 Data Preparation

The next step was to prepare the data in order to fed them with a specific form in our models. Each review document was separated into sentences and from them we kept only the sentences which contain at least one aspect. By doing this we succeed also to augment our dataset as from each review where extracted, finally, much

more annotated sentences. Furthermore, in case a sentence contained more than one aspect, we created a copy of instances from this sentence equal to the number of the aspect it includes. As a result, we had much more instances to use as inputs in order to train our models. An input example is in essence a list with 3 items.

- the whole sentence where the aspect was replaced by the text T in order to mention the position of the aspect in the context
- the aspect itself
- the given polarity for this aspect

For instance the review :
¨εξαιρετικό ξενοδοχείο στο κέντρο της πόλης. φιλικό περιβάλλον, άψογη εξυπηρέτηση, ευρύχωρα και άνετα δωμάτια. πλούσιο πρωινό. μοναδικό του ίσως μειονέκτημα, το δύσκολο πάρκινγκ λόγω των πεζόδρομων, αλλά το ξεχνάς μόλις περάσεις την πόρτα του ξενοδοχείου!'
gives us the following input instances:
[΄φιλικο T , αψογη εξυπηρετηση , ευρυχωρα και ανετα δωματια΄, ΄περιβαλλον΄, 1],
[΄φιλικο περιβαλλον , αψογη T , ευρυχωρα και ανετα δωματια΄, έξυπηρετηση΄, 1],
[΄φιλικο περιβαλλον , αψογη εξυπηρετηση , ευρυχωρα και ανετα T΄, ΄δωματια΄, 1],
[΄πλουσιο T΄, ΄πρωινο΄, 1],
[΄μοναδικο του ισως μειονεκτημα , το δυσκολο T λογω των πεζοδρομων , αλλα το ξεχνας μολις περασεις την πορτα του ξενοδοχειου !΄, ΄παρκινγκ΄, -1], The text inputs described above have been pre-processed before the training phase. Specifically the following steps have been followed:

- convert words to lower case
- remove the accent marks
- keep rare words. Because of the small size of the dataset, reviews with rare words were not eliminated in order not to further decrease the size of the dataset

- keep punctuation symbols, as symbols like (!) may are useful for the task of sentiment classification.

Based on the aforementioned pre-processing approach our dataset finally contains 8.688 instances, where each of them is a list of three items [text,aspect,polarity] where value 1 for polarity means positive, -1 negative and 0 neutral feeling regarding the aspect. The dataset, as summarized in figure 5.3 contains 5.069 aspects with

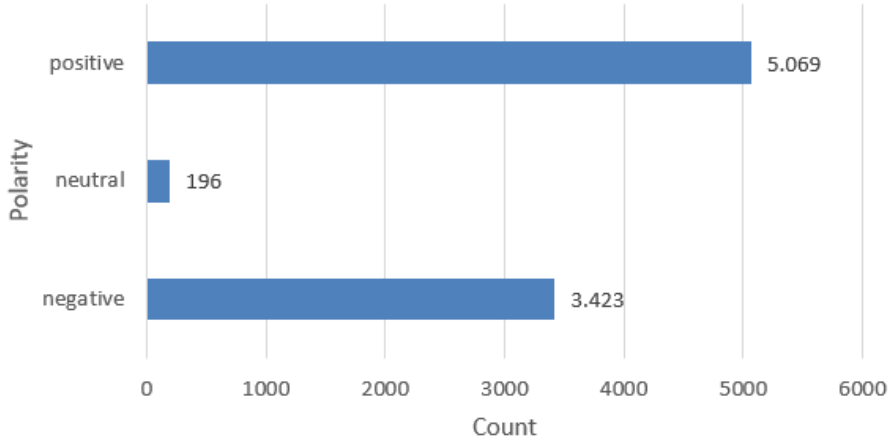

Figure 5.3: Instances per class

positive polarity, 3.423 annotated with negative opinion and only 196 examples of aspects with neutral polarity. As we can understand the dataset is imbalanced and this probably will affect our results. The most frequent annotated aspects are shown in figure 5.4 with their corresponding classification among the three classes.

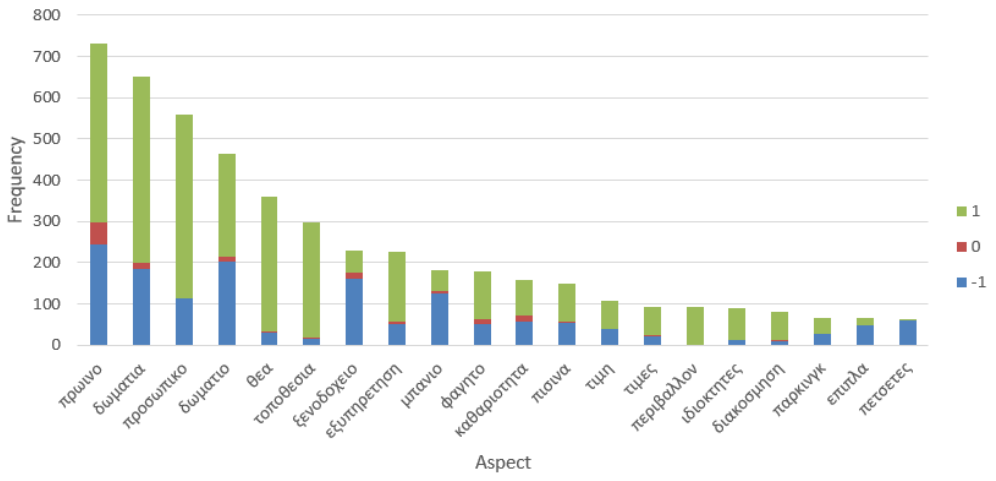

Figure 5.4: Top 20 annotated aspects and the polarity regarding them

From this dataset, a development set has been exported in order to be used for hyper-parameter tuning of the deep-learning models. Specifically, a test-train split of 20% was performed at the initial dataset, with the same distribution from the classes between the train and test dataset.

5.2 Models

Five alternative models for the aspect based sentiment polarity were employed in our experiments, notably one LSTM model and three models which use Transformers. The first three models are actually the implementation in Greek language of the models, LSTM, BERT-SPC and LCF-BERT which described in detail in chapter 4 and are used as baseline models, which is useful because it provides a benchmark against which we can evaluate our final model.

The last model is our contribution in this research field and is, in essence, an extension of LCF-BERT. With the usage of a lexicon to improve the state-of-the-art model's performance by highlighting the opinion words around the aspect as maybe opinion words are more important than other words in the content. The code for all experiments on is written in Python with the PyTorch[3] framework. The implementation of our models is based on corresponding models used in English language which are summed up in [57] and their codes are provided in a github repository[4].

5.2.1 LSTM baseline models

All the baselines that require pre-trained word embeddings use the Greek FastText [97] model[5] to obtain 300-dimensional word embeddings. The max sequence length of each input is selected to be equal to 80 where padding is applied if it is required. In addition, the best architecture for each model is selected with hyper-parameter tuning as described in the next section.

• For the baseline model, we select the TD LSTM which consists of two bidirectional

[3]https://pytorch.org/
[4]https://github.com/songyouwei/ABSA-PyTorch
[5]https://fasttext.cc

LSTM with two hidden layers used to represent the left and right context sequences with targeted aspects, respectively. Their outputs are concatenated and passed straight to a prediction layer, filtered through a softmax function to get a probability vector. The input of LSTM model are only the token indices of the text as they arise from the embedding matrix without highlighting the aspect. As a result the predicted polarity will be the same independently of the aspects contained in the text.

TD LSTM splits the input sequences into left and right context. For example, the sentence: ΄φιλικο περιβαλλον , αψογη εξυπηρετηση , ευρυχωρα και ανετα δωματια΄ has the following representation as input for the TD LSTM model:

$$left\ context + aspect$$

$$aspect + right\ context$$

For our instance, it takes as inputs the left context ΄φιλικο περιβαλλον , αψογη εξυπηρετηση΄ and the right context, έξυπηρετηση , ευρυχωρα και ανετα δωματια΄ , where both of them contain the aspect έξυπηρετηση΄ , with the following representations:
left_text_ind : [114 12 14 142 54 0 0 0 ... 0 0] and
right_text_ind : [73 359 23 1091 14 54 0 0 ... 0 0](*reversed*)

5.2.2 BERT models

The BERT based models were implemented with usage of the pre-trained uncased base GreekBERT[6] model in pytorch. Ilias Chalkidis and the Natural Language Processing Group of Athens University of Economics and Business have released a Greek version of BERT, called GreekBERT[7], which has been trained with texts obtained by Wikipedia, European Parliament Proceedings Parallel Corpus and the Greek part of OSCAR [98], a cleansed version of Common Crawl[8]. GreekBERT is a model similar to the English bert-base-uncased model and consists of 12 encoded

[6]https://huggingface.co/nlpaueb/bert-base-greek-uncased-v1
[7]https://github.com/nlpaueb/greek-bert
[8]https://commoncrawl.org/

layers, 768 hidden layers and 12 self-attention heads [99]. The learning rate has a big impact on BERT's fine-tuning process. Bert's performance can be maximized with only a small learning rate, as demonstrated in the original BERT study. We discovered that if the batch size was too high, the model's performance would suffer due to the instability of regularization across layers, therefore the ideal batch size of 16 was chosen. The LCF-BERT model's convergence rate will be slowed by large dropout, but experimental results demonstrate that it has no effect on LCF design. As a result, after careful study, the model's dropout is set to zero.

BERT follows different pre-processing and tokenizing procedures since it employs word-piece embeddings, it means that a word can be broken down into more than one sub-words, rather than GloVe embeddings. This type of tokenization is useful when dealing with terms that aren't in the vocabulary, and it may also help us better express difficult words. The sub-words are created during the training process and are dependent on the corpus used to train the model. Of course, we could use any other tokenization approach, but the best results will be obtained if we tokenize using the same tokenizer that the BERT model was trained on. For each of the BERTS models, the PyTorch-Pretrained-BERT package includes a tokenizer.

• For BERT SPC the input is based on Next Sentence Prediction task with [CLS] and [SEP] tokens as following:

$$"[CLS]" + sentence + "[SEP]" + aspect + "[SEP]"$$

In our example the input text is:
"[CLS]" + ¨φιλικο περιβαλλον , αψογη εξυπηρετηση , ευρυχωρα και ανετα δωματια' + "[SEP]" + ¨εξυπηρετηση' + "[SEP]"
and the corresponding representation is:

$$\text{text\_bert\_ind}: \begin{matrix}[101 & 2325 & 845 & 119 & 12043 & 2485 & 119 & 30033 & 344 & 4493 & 2218 \\ 102 & 2485 & 102]\end{matrix}$$

• Similarly, for the LCF BERT the input for the local context processor is:

$$"[CLS]" + sentence + "[SEP]"$$

and the input sequence for global context processor is the same as the input of BERT-SPC.

$$"[CLS]" + sentence + "[SEP]" + aspect + "[SEP]"$$

In addition to this, LCF BERT takes as input the masked weights (lcf_vec) created from the CDM/CDW layer in order to pay more attention in the local context of the aspect as described in Chapter 4. Figure 5.5 is visualization of CDW process of aspect ¨εξυπηρετηση' . The color's chroma shows how concentrated the context words are.

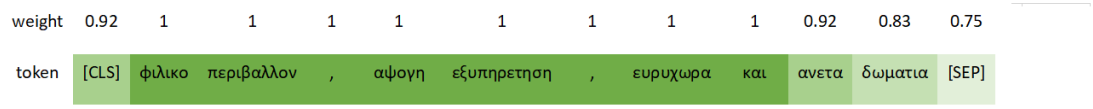

Figure 5.5: Dynamic weighted for local context features during MHSA encoding process.

• Finally, regarding the model we introduce in this thesis and described in Section 4.2, LCF-BERT with lexicon, it contains one more processor in contrast to original LCF-BERT, so we have one more vector as input. This vector (lcf_op_vec) is the input of opinion context processor and is the masked output of the opinion context focus layer where only the local opinion words are unmasked.

Figure 5.6 is visualization of ODM process of aspect ¨εξυπηρετηση' . In the white boxes, the features of the corresponding positions of local context words will be masked.

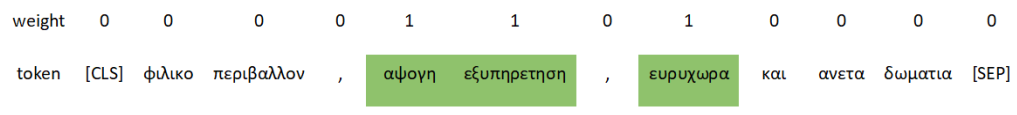

Figure 5.6: Dynamic mask for opinion local context features during MHSA encoding process.

A token is characterized as opinion word with the usage of polarity opinion lexicon in the greek language. A sample of the lexicon is shown below.

Opinion word	Polarity
αγαπημένο	1
αγενέστατο	-1
αισχρά	-1
ακριβό	-1
ανειδίκευτο	-1
ανεπαρκή	-1
ανέσεις	1
γευστικό	1
δελεαστικό	1
ελαττωματικό	-1
ελάχιστα	-1
ευγενική	1
εχθρικό	-1
ήσυχη	1
καλαίσθητο	1
καλές	1
κομψό	1
παλαιομοδίτικα	-1
ποικιλία	1
ποιότητα	1
χάλι	-1
...	...

Table 5.1: Sample of lexicon

5.3 Hyperparameter Optimization

The process of discovering the best combination of hyperparameters that enable the model to perform at its best is known as hyperparameter optimization. The only way to get the best performance out of models is to use the right mixture of hyperparameters. It's an optimization challenge to tune and discover the correct hyperparameters for a model. We aim to find optimal model parameters to minimize the loss function of our model.

In contrast to model parameters, hyperparameters are determined before to training. The weights in a neural network are model parameters learned during training, while the number of layers in a LSTM, the learning rate etc. are hyperpa-

rameters.

Hyperparameter tuning can be done in a manual or automated way. Some of the the basic hyperparameter optimization methods are Manual Search, Random Search, Grid Search, and Bayesian Optimization. In this thesis the hyperparameters for each model are selected with Bayesian Optimization [100]. Unlike random or grid search, Bayesian methods maintain records of previous evaluation results, which they use to build a probabilistic model that maps hyperparameters to a probability of a score on the objective function: $P(score|hyperparameters)$. Bayesian optimization methods are efficient because they pick the next hyperparameters in an informed manner. The fundamental concept is to take a bit longer time picking the next hyperparameters so that the objective function is called less often.

For all the models, we tuned the learning rate considering the range {0.2, 0.002, 0.0002, 0.00002}, the dropout rate in the range {0, 0.1, 0.2}, the batch size in {16, 32} and L2-regularization weight {0.1, 0.001, 0.0001, 0.00001}. For LCF models the SRD threshold is also a hyperparameter in the range {3,4,5}. The default optimizer is Adam, the embedding dimension is 300 and the max_seq_len is by default 80.

Hyper parameters	**TD-LSTM**	**BERT-SPC**	**LCF-BERT**	**LCF-OP BERT**
learning rate	$2*10^{-2}$	$2*10^{-5}$	$2*10^{-5}$	$2*10^{-5}$
L2-reg	$1*10^{-2}$	$1*10^{-5}$	$1*10^{-4}$	$1*10^{-4}$
batch size	16	16	16	16
dropout	0.3	0	0	0
training epochs	6	3	3	3
SRD	-	-	4	4
SRD_lex	-	-	-	2

Table 5.2: Hyperparameter settings for the models

The hyperparameters which finally selected are similar those that were used in the official implementations in the English language.

Chapter 6

Results and Discussion

The goal of this Section is to present the results from the experiments undertaken to test the developed methodologies described in Chapter 4 on the hotel dataset, and compare the performance among the proposed approaches on ABSA task.

The models were detailed in the preceding section, and the hyper-parameters were derived using the development set. The training and development sets were merged and the resulting dataset was used to train the models in order to assess the implemented models using the test set.

For each experiment we show the accuracy and loss plot for train and validation set. As mentioned in Chapter 3, the measures of accuracy, precision, recall and F_1 score used in order to evaluate the performance of the models. These metrics are summarized in the classification report for each class. Also, a confusion matrix with the actuals and predicted labels for each is provided in order to see where the model predicts incorrectly and understand the performance of each model among the three classes: Negative, Neutral, Positive.

6.1 Results

We begin presenting the results by presenting our baseline model. As described in Chapter 5 the baseline model is a LSTM-based model.

• The TD-LSTM model trained with the hyperparameters described in section 5.3. Accuracy and loss plots for train and test datasets are shown in Figures 6.1 and

6.2. As we can see from the figures above the validation accuracy is greater then

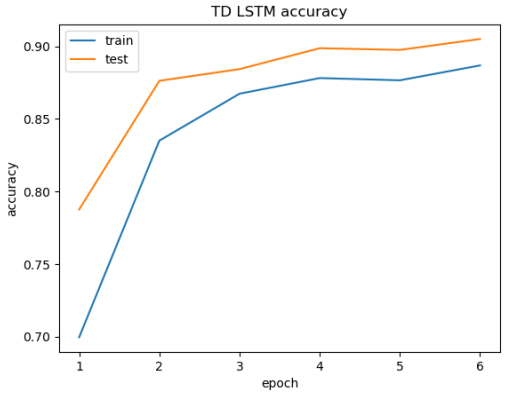

Figure 6.1: TD LSTM training accuracy per epoch

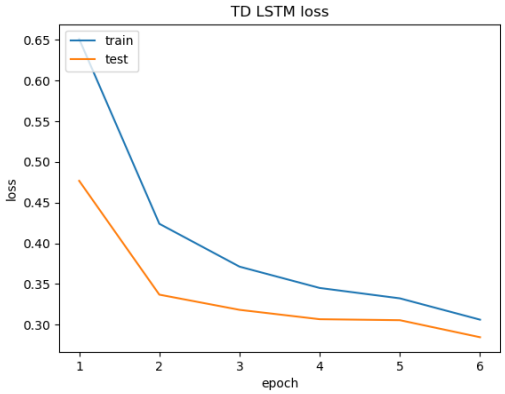

Figure 6.2: TD LSTM training accuracy per epoch

training accuracy from the first epoch. The training loss drops rapidly throughout the epochs, while the validation loss decreases more slowly. However this does not mean that we face overfitting as the train accuracy continue to increase.

In table 6.1 the classification report with the evaluation metrics regarding this experiment is shown, as well as the confusion matrix in table 6.2. The macro avg in classification report is the average of the corresponding metrics for the three classes.

Class	Precision	Recall	F-score	Support
negative	0.90	0.88	0.89	699
neutral	1.00	0.00	0.00	32
positive	0.91	0.95	0.93	1007
accuracy			0.91	1738
macro avg	0.94	0.61	0.61	1738
weighted avg	0.91	0.91	0.90	1738

Table 6.1: Classification Report for TD LSTM model

		Predicted		
		Negative	Neutral	Positive
Actual	Negative	616	0	83
	Neutral	19	0	13
	Positive	50	0	957

Table 6.2: Confusion matrix for TD LSTM model

As we can see from classification report and confusion matrix the model is suffering to predict the negative and especially the neutral class. The main reason for this performance is the fact that we have very small sample size for these two classes. Most of the false predicted values for neutral class are misclassified as positive.

• Now we represent the results for the BERT based approaches. The first model we evaluate is BERT-SPC. Accuracy and loss plots for train and test datasets are shown in Figures 6.3 and 6.4.

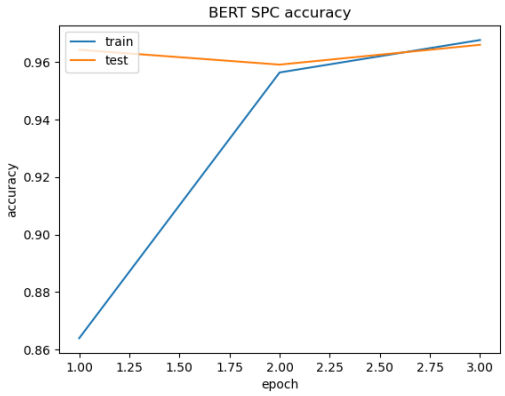

Figure 6.3: BERT SPC training accuracy per epoch

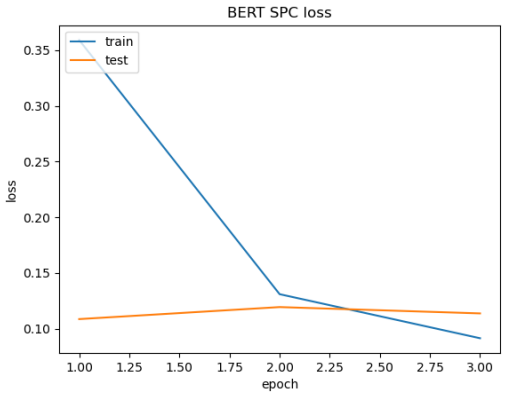

Figure 6.4: BERT SPC training accuracy per epoch

In table 6.3 the classification report with the evaluation metrics regarding this experiment is shown, as well as the confusion matrix in table 6.4. By using the BERT approach we see that we achieve remarkable accuracy score. However the most important result is that BERT model classifies correctly instances from the neutral class.

Also, false positives from positive class which were classified as negative, with

Class	Precision	Recall	F-score	Support
negative	0.95	0.97	0.96	699
neutral	0.86	0.56	0.68	32
positive	0.98	0.98	0.98	1007
accuracy			0.97	1738
macro avg	0.93	0.84	0.87	1738
weighted avg	0.97	0.97	0.97	1738

Table 6.3: Classification Report for BERT-SPC model

		Predicted		
		Negative	Neutral	Positive
Actual	Negative	675	3	21
	Neutral	12	18	2
	Positive	21	0	986

Table 6.4: Confusion matrix for BERT-SPC model

BERT-SPC are classified as neutral which is more normal as positive is closer to neutral class.

• The next BERT based model is the model with the greatest performance among the state-of-the-art approaches for ABSA task, named LCF-BERT.

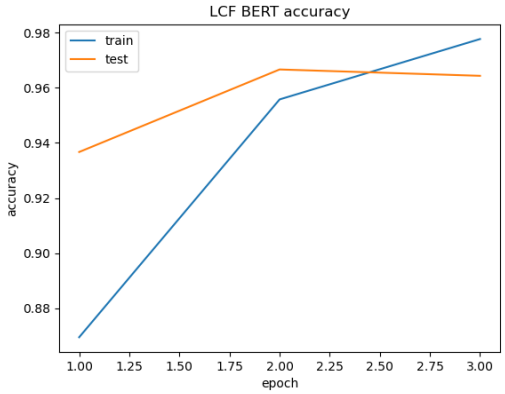

Figure 6.5: LCF-BERT training accuracy per epoch

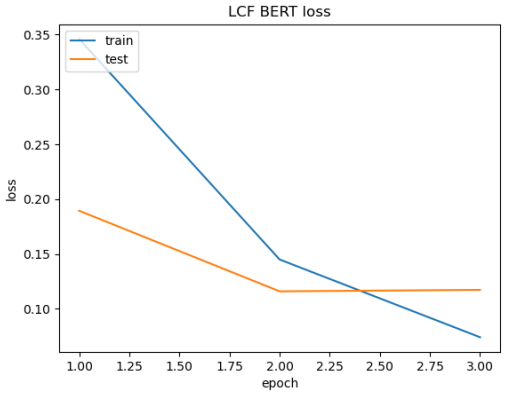

Figure 6.6: LCF-BERT training accuracy per epoch

In table 6.5 the classification report with the evaluation metrics regarding thi-

sexperiment is shown, as well as the confusion matrix in table 6.6. LCF-BERT improves more F-score and recall.

Class	Precision	Recall	F-score	Support
negative	0.98	0.94	0.96	699
neutral	0.79	0.69	0.73	32
positive	0.96	0.99	0.97	1007
accuracy			0.96	1738
macro avg	0.91	0.87	0.89	1738
weighted avg	0.96	0.96	0.96	1738

Table 6.5: Classification Report for LCF-BERT model

		Predicted		
		Negative	Neutral	Positive
Actual	Negative	658	5	36
	Neutral	4	22	6
	Positive	10	1	996

Table 6.6: Confusion matrix for LCF-BERT model

• The final BERT based model, LCF-OP BERT, is the model introduced in this thesis and an extension of LCF-OP BERT model.

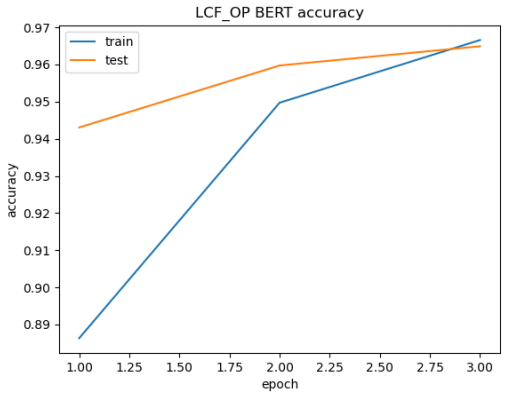

Figure 6.7: LCF-OP BERT training accuracy per epoch

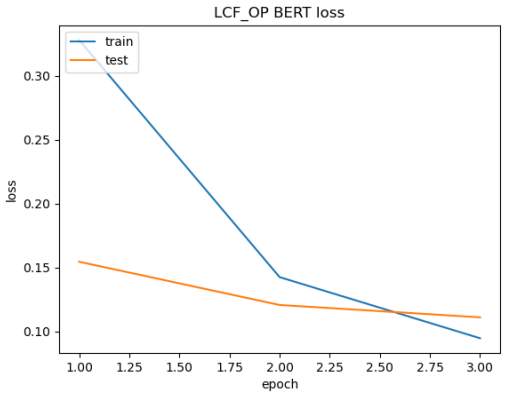

Figure 6.8: LCF-OP BERT training accuracy per epoch

LCF-OP BERT has similar results as LCF-BERT but achieves to improve little more the recall score.

Class	Precision	Recall	F-score	Support
negative	0.98	0.94	0.96	699
neutral	0.74	0.72	0.73	32
positive	0.9	0.96	0.98	1007
accuracy			0.96	1738
macro avg	0.89	0.88	0.89	1738
weighted avg	0.97	0.96	0.96	1738

Table 6.7: Classification Report for LCF-OP BERT model

		Predicted		
		Negative	Neutral	Positive
Actual	Negative	658	7	34
	Neutral	3	23	6
	Positive	10	1	996

Table 6.8: Confusion matrix for LCF-OP BERT model

6.2 Discussion

On our annotated dataset, we compare the performance of the LCF design to different baseline models. The results show that LCF design, may significantly enhance the state-of-the-art performance. Especially with the addition of a lexicon we achieve to increase a little the recall score as the LCF-OP BERT model predicts better especially for the neutral class with the least instances.

Model	**Accuracy**	**Recall**	**Precision**	**F-score**
TD-LSTM	90.51	61.05	93.60	60.64
BERT-SPC	96.61	83.58	92.92	87.23
LCF BERT	96.43	87.26	90.81	88.91
LCF-OP BERT	96.49	**88.31**	89.47	**88.86**

Table 6.9: Experimental results of performance of the models

Chapter 7

Conclusions and Future Work

In this chapter, we wrap up the thesis by summarizing the findings of the experiments from the models explained in Chapter 4 and presenting potential work lines that will be pursued to further expand the study provided here.

7.1 Conclusion

In this master thesis, the task of Aspect Based Sentiment Analysis (ABSA) with a small dataset in the Greek Language is studied. The subject is examined mainly in the English Language and is one of the challenging tasks in SemEval [1] research workshops. In these workshops ABSA task consists of the following subtasks: Subtask 1 - Aspect term extraction, Subtask 2 - Aspect term polarity, Subtask 3 - Aspect category detection and Subtask 4 - Aspect category polarity. In this thesis we focus only on Subtask 2 - Aspect term polarity for a given aspect target.

We presented some of the state-of-the-art approaches on this task and we tried to adopt them in the Greek Language. The initial dataset contained user reviews for hotels, and was annotated manually by us. We applied two types of models. The first was based on LSTM networks and were used as our baseline models. Then we applied two models with the higher performance in this task in the English Language. These models introduce BERT, a recent transfer learning model from

[1] https://semeval.github.io/

Google and the most accurate, LCF-BERT, gives also attention to the local context of the aspect instead of all the others approaches which are applied on the global context. Our contribution was to extent the LCF- BERT model with the usage of a lexicon in order to highlight and give more attention on local opinion words which may surround the aspect we want to predict the sentiment polarity expressed for in a sentence.

We evaluated the performance of the models with the metrics of accuracy, recall, precision and F-score. The results revealed that all the evaluation metrics increased impressively for the case of the Transformer while the accuracy was increased by 9%. Also the Local Context Focus approaches which give focus on the content closed to the aspect achieve better results and with the addition of the lexicon we improved the performance, especially in the neutral class with the with the fewest instances.

7.2 Future Work

All of the approaches mentioned produce excellent results and can be employed in the aspect based sentiment analysis task.

To test the ABSA framework's robustness, the first step in future development would be to apply it to other domain-dependent datasets. The dataset used in this thesis was small with small vocabulary and thus more easier to train the models fast.

Regarding BERT based models, they are in essence unsupervised models, which have been pre-trained on massive amounts of unlabeled data. A research direction will be the fine tuning of the Transformer along with the use of pretrained ebmeddings in order to evaluate the potential increase in the performance of the systems trained with new datasets. Furthermore, other researchers have built several versions of the BERT model, such as DistilBERT, RoBERTa, each with its own set of features and performance. The ABSA framework developed in this study can be applied to the various pre-trained models to assess performance differences.

For Local Context Focus models another way of finding the local context can be tried. The SRD threshold could be dynamically calculated, for example based

on the length of the sentence or the distance between each aspect. In addition as mentioned in Section 5.2, in our model, for the opinion local context mask we highlight the words that are opinion words as they are included in the lexicon. For further research, could be take into account the polarity of those opinion words, in order to investigate how this information affect the performance.

Finally, another research direction will be the creation of an end-to-end system, which will extract simultaneously the aspect may exist in a sentence and its corresponding sentiment polarity.

References

[1] Gayatree Ganu, Noemie Elhadad, and Amélie Marian. Beyond the stars: Improving rating predictions using review text content. In *WebDB*, 2009.

[2] Bing Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, May 2012.

[3] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Multi-facet rating of product reviews. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, page 461–472, Berlin, Heidelberg, 2009. Springer-Verlag.

[4] Bo Pang and Lillian Lee. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[5] 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-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[6] Bin Lu, Myle Ott, Claire Cardie, and Benjamin K. Tsou. Multi-aspect sentiment analysis with topic models. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*, ICDMW '11, page 81–88, USA, 2011. IEEE Computer Society.

[7] Zhenxin Qin. A framework and practical implementation for sentiment analysis and aspect exploration. 2016.

[8] Bob Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[9] Pedro Balage Filho and Thiago Pardo. NILC_USP: A hybrid system for sentiment analysis in Twitter messages. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 568–572, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics.

[10] Peter D. Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 417–424, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[11] Michael Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 841–847, Geneva, Switzerland, aug 23–aug 27 2004. COLING.

[12] Janyce M. Wiebe. Learning subjective adjectives from corpora. In *In AAAI*, pages 735–740, 2000.

[13] Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. Learning subjective language. *Comput. Linguist.*, 30(3):277–308, September 2004.

[14] Ramanathan Narayanan, Bing Liu, and Alok Choudhary. Sentiment analysis of conditional sentences. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 180–189, Singapore, August 2009. Association for Computational Linguistics.

[15] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel, editors, *KDD*, pages 168–177. ACM, 2004.

[16] Josef Steinberger, Tomáš Brychcín, and Michal Konkol. Aspect-level sentiment analysis in Czech. In *Proceedings of the 5th Workshop on Computational*

Approaches to Subjectivity, Sentiment and Social Media Analysis, pages 24–30, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[17] Sasha Blair-goldensohn, Tyler Neylon, Kerry Hannan, George A. Reis, Ryan Mcdonald, and Jeff Reynar. Building a sentiment summarizer for local service reviews. In *In NLP in the Information Explosion Era*, 2008.

[18] Chong Long, Jie Zhang, and Xiaoyan Zhu. A review selection approach for accurate feature rating estimation. In *Coling 2010: Posters*, pages 766–774, Beijing, China, August 2010. Coling 2010 Organizing Committee.

[19] Hongning Wang, Yue Lu, and ChengXiang Zhai. Latent aspect rating analysis without aspect keyword supervision. In Chid Apté, Joydeep Ghosh, and Padhraic Smyth, editors, *KDD*, pages 618–626. ACM, 2011.

[20] Gang Li, Rob Law, Huy Quan Vu, Jia Rong, and Xinyuan (Roy) Zhao. Identifying emerging hotel preferences using Emerging Pattern Mining technique. *Tourism Management*, 46(C):311–321, 2015.

[21] Xiaowen Ding, Bing Liu, and Philip S. Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of First ACM International Conference on Web Search and Data Mining (WSDM-2008)*, 2008.

[22] Chetan Mate. Product aspect ranking using sentiment analysis: A survey. 2016.

[23] Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, and Rada Mihalcea. Beneath the tip of the iceberg: Current challenges and new directions in sentiment analysis research. *CoRR*, abs/2005.00357, 2020.

[24] Livia Polanyi and Annie Zaenen. Contextual lexical valence shifters. In Yan Qu, James Shanahan, and Janyce Wiebe, editors, *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*. AAAI Press, 2004. AAAI technical report SS-04-07.

[25] Karo Moilanen and Stephen Pulman. Sentiment composition. In *Proceedings of Recent Advances in Natural Language Processing (RANLP 2007)*, pages 378–382, September 27-29 2007.

[26] Yejin Choi and Claire Cardie. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 793–801, Honolulu, Hawaii, October 2008. Association for Computational Linguistics.

[27] Vassiliki Rentoumi, Stefanos Petrakis, Manfred Klenner, George A. Vouros, and Vangelis Karkaletsis. United we stand: Improving sentiment analysis by joining machine learning and rule based methods. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2010, 17-23 May 2010, Valletta, Malta*. European Language Resources Association, 2010.

[28] Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. Dependency tree-based sentiment classification using CRFs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794, Los Angeles, California, June 2010. Association for Computational Linguistics.

[29] Clayton J. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Eytan Adar, Paul Resnick, Munmun De Choudhury, Bernie Hogan, and Alice H. Oh, editors, *ICWSM*. The AAAI Press, 2014.

[30] Erik Cambria, Yang Li, Frank Z. Xing, Soujanya Poria, and Kenneth Kwok. *SenticNet 6: Ensemble Application of Symbolic and Subsymbolic AI for Sentiment Analysis*, page 105–114. Association for Computing Machinery, New York, NY, USA, 2020.

[31] Orith Toledo-Ronen, Roy Bar-Haim, Alon Halfon, Charles Jochim, Amir Menczel, Ranit Aharonov, and Noam Slonim. Learning sentiment composition from sentiment lexicons. In *Proceedings of the 27th International Conference*

on Computational Linguistics, pages 2230–2241, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[32] Svetlana Kiritchenko and Saif M. Mohammad. Sentiment composition of words with opposing polarities. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1102–1108, San Diego, California, June 2016. Association for Computational Linguistics.

[33] Saif M. Mohammad. Challenges in sentiment analysis. In *A Practical Guide to Sentiment Analysis*. Springer, 2016.

[34] Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. NRC-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442, Dublin, Ireland, August 2014. Association for Computational Linguistics.

[35] Duy-Tin Vo and Yue Zhang. Target-dependent twitter sentiment classification with rich automatic features. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1347–1353. AAAI Press, 2015.

[36] Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 151–160, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[37] Wayne X. Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 56–65, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[38] Rodrigo Moraes, João Francisco Valiati, and Wilson P. Gavião Neto. Document-level sentiment classification: An empirical comparison between

svm and ann. *Expert Syst. Appl.*, 40(2):621–633, 2013.

[39] Songbo Tan, Xueqi Cheng, Yuefen Wang, and Hongbo Xu. Adapting naive bayes to domain adaptation for sentiment analysis. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, page 337–349, Berlin, Heidelberg, 2009. Springer-Verlag.

[40] Samaneh Moghaddam and Martin Ester. Opinion digger: An unsupervised opinion miner from unstructured product reviews. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, page 1825–1828, New York, NY, USA, 2010. Association for Computing Machinery.

[41] Justin Martineau and Tim Finin. Delta tfidf: An improved feature space for sentiment analysis. In Eytan Adar, Matthew Hurst, Tim Finin, Natalie S. Glance, Nicolas Nicolov, and Belle L. Tseng, editors, *ICWSM*. The AAAI Press, 2009.

[42] Yelena Mejova and P. Srinivasan. Exploring feature definition and selection for sentiment classifiers. In *ICWSM*, 2011.

[43] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[44] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[45] Hy Nguyen and Kiyoaki Shirai. A joint model of term extraction and polarity classification for aspect-based sentiment analysis. In *2018 10th International Conference on Knowledge and Systems Engineering (KSE)*, pages 323–328, 2018.

[46] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[47] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, October 2014. Association for Computational Linguistics.

[48] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[49] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[50] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas, November 2016. Association for Computational Linguistics.

[51] Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. Interactive attention networks for aspect-level sentiment classification. *CoRR*, abs/1709.00893, 2017.

[52] Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 452–461, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[53] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014. cite arxiv:1406.1078Comment: EMNLP 2014.

[54] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018. cite arxiv:1802.05365Comment: NAACL 2018. Originally posted to openreview 27 Oct 2017. v2 updated for NAACL camera ready.

[55] Alec Radford and Ilya Sutskever. Improving language understanding by generative pre-training. In *arxiv*, 2018.

[56] Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. Attentional encoder network for targeted sentiment classification. *CoRR*, abs/1902.09314, 2019.

[57] Biqing Zeng, Heng Yang, Ruyang Xu, Wu Zhou, and Xuli Han. Lcf: A local context focus mechanism for aspect-based sentiment classification. *Applied Sciences*, 9(16), 2019.

[58] Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4933–4941, Marseille, France, May 2020. European Language Resources Association.

[59] Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. Attentive gated lexicon reader with contrastive contextual co-attention for sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3443–3453, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[60] Kia Dashtipour, Soujanya Poria, Amir Hussain, Erik Cambria, Ahmad Y. A. Hawalah, Alexander F. Gelbukh, and Qiang Zhou. Multilingual sentiment analysis: State of the art and independent comparison of techniques. *Cogn. Comput.*, 8(4):757–771, 2016.

[61] Adam Tsakalidis, Symeon Papadopoulos, Rania Voskaki, Kyriaki Ioannidou, Christina Boididou, Alexandra I. Cristea, Maria Liakata, and Yiannis Kompatsiaris. Building and evaluating resources for sentiment analysis in the greek language. *Lang. Resour. Eval.*, 52(4):1021–1044, December 2018.

[62] Nikolaos Spatiotis, Iosif Mporas, Michael Paraskevas, and Isidoros Perikos. Sentiment analysis for the greek language. In *Proceedings of the 20th Pan-Hellenic Conference on Informatics*, PCI '16, New York, NY, USA, 2016. Association for Computing Machinery.

[63] Georgios Kalamatianos, Dimitrios Mallis, Symeon Symeonidis, and Avi Arampatzis. Sentiment analysis of greek tweets and hashtags using a sentiment lexicon. In *Proceedings of the 19th Panhellenic Conference on Informatics*, PCI '15, page 63–68, New York, NY, USA, 2015. Association for Computing Machinery.

[64] George Markopoulos, George Mikros, Anastasia Iliadi, and Michalis Liontos. Sentiment analysis of hotel reviews in greek: A comparison of unigram features. 9:373–383, 01 2015.

[65] Angela Braoudaki, Eleni Kanellou, Christos Kozanitis, and Panagiota Fatourou. Hybrid data driven and rule based sentiment analysis on greek text. *Procedia Computer Science*, 178:234–243, 2020.

[66] Hallén R. Joselson N. Emotion classification with natural language processing. 2019.

[67] Warren Mcculloch and Walter Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[68] Richard H. R. Hahnloser, Rahul Sarpeshkar, Misha A. Mahowald, Rodney J. Douglas, and H. Sebastian Seung. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405(6789):947–951, Jun 2000.

[69] Sebastian Ruder. An overview of gradient descent optimization algorithms., 2016. cite arxiv:1609.04747Comment: Added derivations of AdaMax and Nadam.

[70] Jeffrey L. Elman. Finding structure in time. *COGNITIVE SCIENCE*, 14(2):179–211, 1990.

[71] https://colah.github.io/posts/2015-08-understanding-lstms/.

[72] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[73] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 2000.

[74] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

[75] Alex Graves, Navdeep Jaitly, and Abdel rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *ASRU*, 2013.

[76] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[77] Sebastian Ruder. Neural transfer learning for natural language processing. 2019.

[78] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification, 2016. cite arxiv:1607.01759.

[79] Robert Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI Conference on Artificial Intelligence*, 2016.

[80] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE Computer Society, 2009.

[81] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[82] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. cite arxiv:1810.04805Comment: 13 pages.

[83] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *ACL*. Association for Computational Linguistics, 2018.

[84] Bihorac A. Hoang M. Aspect-based sentiment analysis using the pre-trained language model bert. 2019.

[85] S.J. Pan and Q. Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[86] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[87] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[88] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[89] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[90] C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, USA, 2nd edition, 1979.

[91] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL-Smadi Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 19–30, 2016.

[92] Hu Xu, Bing Liu, Lei Shu, and Philip Yu. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[93] Binxuan Huang, Yanglan Ou, and Kathleen M. Carley. Aspect level sentiment classification with attention-over-attention neural networks. *CoRR*, abs/1804.06536, 2018.

[94] Xin Li, Lidong Bing, Wai Lam, and Bei Shi. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[95] Duyu Tang, Bing Qin, and Ting Liu. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224, Austin, Texas, November 2016. Association for Computational Linguistics.

[96] Ioannis Manousos Katakis, Georgios Petasis, and Vangelis Karkaletsis. CLARIN-EL web-based annotation tool. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4505–4512, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[97] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information, 2016. cite arxiv:1607.04606Comment: Accepted to TACL. The two first authors contributed equally.

[98] Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. A monolingual approach to contextualized word embeddings for mid-resource languages. *CoRR*, abs/2006.06202, 2020.

[99] John Koutsikakis, Ilias Chalkidis, Prodromos Malakasiotis, and Ion Androutsopoulos. Greek-bert: The greeks visiting sesame street, 08 2020.

[100] Julien-Charles Lévesque, Christian Gagné, and Robert Sabourin. Bayesian hyperparameter optimization for ensemble learning. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, UAI'16, page 437–446, Arlington, Virginia, USA, 2016. AUAI Press.