Arab American University – Palestine
Faculty of Graduate Studies

On Developing a Machine Learning Context-sensitive Sentence Level Sentiment Analyzer

By

Shatha Mahmoud Rabaia

Main Supervisor

Dr. Mohammed A. M. Maree

Co- Supervisor

Dr. Mujahed Eleyat

This thesis was submitted in partial fulfillment of the requirements for the Master`s degree in Computer Science

April / 2022

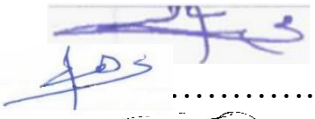
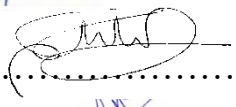
Thesis Approval

On Developing a Machine Learning Context-sensitive Sentence Level Sentiment Analyzer

By

Shatha Rabaia

This thesis was defended successfully on 28/5/2022. and approved by:

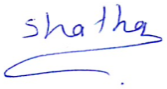
Committee members	Signature
1. Supervisor Name: Dr. Mohammed A. M. Maree	
2. Co- Supervisor Name: Dr. Mujahed Eleyat	……..
3. Internal Examiner Name: Dr. Ahmad Hasasneh	
4. External Examiner Name:Dr. Ramesh Kumar Ayyasamy	 ……..

Declaration

This is to declare that the thesis entitled “On Developing a Machine Learning Context-sensitive Sentence Level Sentiment Analyzer” under the supervision of Dr. Mohammed A. M. Maree and Dr. Mujahed Eleyat is my own work and does not contain any unacknowledged work or material previously published or written by another person, except where due reference is made in the text of the document.

Name: Shatha Rabaia

Date: April 20, 2022

Signature: 

Acknowledgment

Foremost, I would like to express my sincere gratitude to my supervisor Dr. Mohammed Maree for the continuous support of my master thesis and research, for his patience, enthusiasm, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

My sincere thanks also go to my co-supervisor Dr. Mujahed Eleyat for the useful comments, remarks and engagement.

I would like to thank my husband Dr. Osama Khader and my daughters Reem, and Yara. Their help, support, patience, and understanding made this endeavor possible.

Huge thanks, love and appreciation go to my family: My father Mr. Mahmoud Rabayah, my beloved mother Mrs. Hiam Rabayah, my brothers Wael, Tareq, Dr. Mohammed, and Ahmad and to all of my sisters: Fida', Samah, Remah, and Muna for their belief in me and my work and for supporting me spiritually throughout my life.

A very special gratitude goes out to all of my friends who have supported me over the last few years: Enas Alawneh, Khetam Abuhadia, Sumoud Jaradat, Hanean Qtait, Lama Alqasrawi, and Zeina Sadeddin.

Thanks for all your encouragement

Abstract

On Developing a Machine Learning Context-sensitive Sentence Level Sentiment Analyzer

By: Shatha Rabaia

Main Supervisor: Dr. Mohammed A. M. Maree

Co- Supervisor: Dr. Mujahed Eleyat

Sentiment Analysis (SA) has become one of the most reliable tools for assisting organizations better understand the perception of their users about the products and services that they offer. In particular, with the ever-increasing user-generated sentiment reviews on the Web, SA tools have become significantly indispensable. The exploitation of such tools in real-world application domains does not only serve organizations, but also individuals who are interested in learning more about the various perceptions about the products and/or services that are being used by other users or customers. Over the past few years, there has been a growing number of SA techniques which can be characterized by a number of strengths and weaknesses; demonstrated by their accuracy rates in terms of the results that they produce. Lexicon-based and machine learning approaches have been among the most frequent techniques that are employed for this purpose in particular. To address limitations of these techniques, newer neural network models have been proposed in an attempt to automate the feature learning process and enrich the learned features with word contextual embeddings to identify their semantic orientations. However, the high accuracy rates of such models entail sentences should fall under the same domain of the training data used beforehand.

In this thesis, we experimentally evaluate six sentiment classification methods, namely using Support Vector Machines (SVM), Naive Bayes (NB), Logistic Regression,

Random Forests, Feedforward Neural Network, and Convolutional Neural Networks. In addition to experimentally evaluating the first four conventional methods, we also measure the impact of incorporating lexical semantic knowledge captured by WordNet on expanding original words in sentences using a variety of NLP pipelines. The two latter methods are based on neural networks with automated feature processing and ability to enrich learned features with word contextual embeddings. Besides measuring their quality, we experimentally investigate the impact of exploiting GloVe word embeddings on enriching feature vectors extracted from sentiment sentences. In the conducted experiments, we have used four real-world datasets that comprise1,600,000 tweets, 50,000 movie reviews, 10,662 sentences, and 300 generic movie reviews. With regard to the first five methods, results indicate that coupling lemmatization and knowledge-based n-gram features proved to produce higher accuracy rates. With this coupling, the accuracy of the SVM classifier has improved to 90.43%, while it was 86.83%, 90.11%, 86.20%, 88.01%, respectively using the four other classifiers. For neural networks-based methods (FNN and CNN), findings indicate that using larger dimensions of GloVe word embeddings increases the sentiment classification accuracy. In particular, results demonstrate that the achieved accuracy of the CNN using a larger feature map, a smaller filter size, as well as ReLU activation function in the convolutional layer was 90.57% when applied on the IMDB dataset, while it was 82.02% and 78.14% using Twitter's and the sentiment sentences datasets, respectively.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

SA: Sentiment Analysis.

NLP: Natural Language Processing.

ANNs: Artificial Neural Network.

FNN: Feedforward Neural Network

CNN: Convolutional Neural Network.

RNN: Recurrent Neural Networks.

LSTM: Long-Short Term Memory.

DBGRU: Deep Bidirectional Gated Recurrent Unit.

MLP: Multi-Layer Perceptron.

GloVe: Global Vector.

Word2Vec: Word to Vector.

SSWE: Semantic Specific Word Embedding.

POS: Part-Of-Speech.

ADJ: Adjective.

ADV: Adverb.

VV: Verb.

RL: Reinforcement Learning

SVM: Support Vector Machine.

NB: Naïve Bayes.

LR: Logistic Regression.

RF: Random Forest.

FF: Feature Frequency.

TF-IDF: Term Frequency-Inverse Document Frequency

STS-Gold: Stanford Twitter Sentiment Gold Standard.

RelU: Rectified Linear Unit

TP: true positive.

TN: true negative.

FP: False Positive.

FN: False Negative.

Chapter One

Introduction

1.1 Introduction and Problem Statement

Actually, internet has changed how individuals express their opinions, views. Blog posts, product review sites, social media and other platforms are the main ways people express their thoughts and opinions. People today use social network sites such as Facebook, Instagram, Twitter, etc., to share their emotions, opinions and ideas. Individuals can influence and inform others through this interactive content. The social media generates vast amounts of sentiment-rich data, such as tweets, comments, reviews, etc.. Further, social media offers businesses an opportunity to reach potential customers by creating a platform for advertising. In order to make their decision, people generally rely heavily on the content generated by users of the Internet. Before buying a product or using any service, a consumer often look for its reviews online. The same can be said for organizations which improve their products and services by taking the opinions of their customers into account. Since users generated content is too large and varied to be processed manually, there is a need to automate it. As a result various sentiment analysis techniques are used. Sentiment Analysis (SA) has recently become one of the most active research areas in the field of Natural Language Processing (NLP) [1, 2]. In general, SA is a classification problem that aims at identifying, extracting, and analyzing the sentiment orientation of sentences. Several formal definitions to SA can be found in the literature, among which the definition by Alam and Yao is widely quoted where it is defined as "*a computational process which identifies and categorizes an opinion in a piece of text that expresses the positive, negative, or neutral attitude of a*

writer towards a particular product, event or personality" [3]. Recently, SA has been considered as one of the most reliable resources for assisting organizations better understand the perception of their users about the products and services that they offer. It has also become one of the active research fields in knowledge representation and NLP. More importantly, with the ever-growing user-generated sentiment reviews on the Web, the significance of SA has become more crucial [1, 4, 5]. As such, exploiting SA techniques in real-world application domains has several benefits for both organizations and even individuals whom are interested in learning more about the various perceptions about different products and/or services that are being used by other users or customers [6, 7].

In general, there are two main SA approaches that can be characterized by a number of strengths and weaknesses as they vary in their accuracy in terms of the results that they produce. These are i) lexicon-based approaches [8, 9], and ii) machine learning approaches [6, 10-12]. For more details on each of these approaches, we refer the reader to recent surveys [4, 5, 13, 14] that have highlighted the challenges associated to the techniques that are employed within each approach. As far as lexicon-based approaches are concerned, a lexicon containing a set of words associated with their sentiment polarity is exploited to tag words in sentences with their respective term polarities. Examples of such lexica are: SentiWordNet, SenticNet and HowNet lexicons [1]. Under the umbrella of lexicon-based approaches, there are two main groups of methods that are utilized for SA purposes. These are the i) Corpus-based and ii) Dictionary-based methods. The corpus-based method aims to create a dictionary for a specific field, where it is formed through a group of seed opinion terms. This dictionary is expanded by employing semantic and statistical techniques in order to search for new words

related to the field. Synonymy and antonymy relationships, which can be obtained using WordNet thesaurus, are examples of using semantic techniques, while latent sentiment analysis is an example of a statistical technique. On the other hand, dictionary-based techniques are based on using a manually constructed dictionaries such as a SentiWordNet and are expanded through the use of synonymy, hypernymy, meronymy and antonymy relations that are obtained from WordNet. In this approach, a sentence polarity depends on the polarity of its micro phrases which can be found by calculating the polarity score of its terms which can be obtained from lexical resources like SenticNet or SentiWordNet lexica. However, depending on the lexicon alone is not sufficient and produces inaccurate results because a word's prior-polarity doesn't reflect its contextual polarity in a sentence. In addition, in order to obtain better results when using the lexicon approach, it must be combined with various semantic relations and taxonomies which requires defining such elements manually in most scenarios.

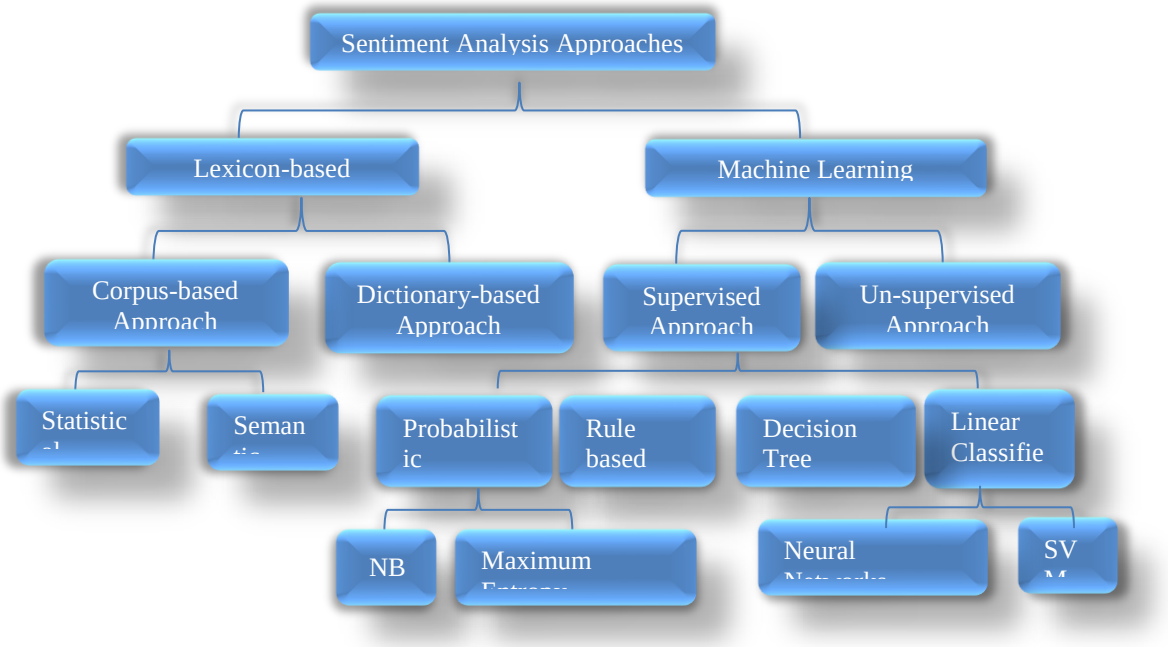


Figure 1:Sentiment Analysis Approaches [29].

On the other hand, machine learning based approaches rely on using training samples to predict the polarity of opinions. There are two main classes of machine learning methods that are employed for SA purposes. These are: i) Supervised and ii) Un-supervised learning methods. Unsupervised learning relies on using clustering techniques, wherein the analysis is performed on unlabeled dataset. This method is used when it is not possible to find a labeled dataset to be used for training the classifier. On the other hand, supervised machine learning is based on approaches that rely on using training samples to predict the polarity of opinions where a group of data is classified into several labeled classes, like positive, neutral, negative, that represent feature prediction vectors. Conventional methods using Support Vector Machine (SVM), Maximum Entropy (ME) and Naïve Bayes (NB), which are often combined with complex feature extraction process, are utilized for this purpose. Among the main limitations of these approaches are the incompleteness of the training data, lack of semantic information about the processed text, domain-dependance and huge computational cost when processing verbose sentiment sentences. In general, sentiment analysis using machine learning involves four main phases: data collection and preparation, feature extraction, training, and testing the sentiment classifier. With the recent advances in deep learning techniques, specifically in the NLP domain, new models have been proposed in an attempt to address these challenges [15, 16]. Among the main goals in this context is to automate the feature learning process; requiring less training data, and introducing semantic dimensions through the utilization of word embedding techniques. For more details on the main advantages that deep learning models introduce, we refer the reader to a recent survey proposed by the authors of [17]. An important aspect that we would like to highlight in the context of utilizing deep learning is the ability to couple discrete representations of text using One-hot vectors

and distributional representations of words using Global Vectors (GloVe) or Word2Vec. However, in view of the diversity of neural networks and the multiple factors and configuration settings of the neural network architecture, defining the appropriate parameters to obtain the best results presents a challenge to researchers in this field. Starting from this position, we aim to experimentally investigate the impact of exploiting GloVe word embeddings on enriching feature vectors extracted from sentiment sentences, and subsequently its impact on the quality of the utilized sentiment classifiers, namely FNN and CNN.

Before we proceed with the discussion, it is important to point out that there are SA techniques that can be performed at different levels including: document level, such as the work proposed in [18], sentence level [19-22], and aspect level or word level [23, 24]. In addition, the sentiment orientation of sentences can be classified according to a three-class (positive, negative or neutral reviews) or five-class (very positive, positive, neutral, negative, very negative) polarity scale. At document level, the whole document is treated as one unit which may lead to inaccurate results because contrary attitudes related to the same topic can be found. Sentiment analysis at the word level is also imprecise because the polarity of the word varies depending on the context where it appears. Therefore, relying on the prior polarity of a word does not reflect the polarity of its context. Therefore, we focus our research activities on analyzing the sentiment at sentence level, wherein sentiment sentences consist of a collection of factors used to represent users' opinions. In this line of research, researchers have considered different factors to represent a given opinion. For instance, Kim and Hovy in [25] used four factors to represent opinions, which are topic, opinion holder, claim about the topic, and sentiment. Other researchers like Visha and Sonawane [21] defined an opinion as a

combination of five factors, which are the object, object feature, opinion, polarity of the opinion, and opinion holder. An object in sentence represents an entity which may be a person, company, product, or subject. A feature is an attribute of an object while an opinion is a comment about a feature that is used to evaluate object polarity. Polarity is an orientation of a comment on an object feature. On the other hand, the entity or person or institution that expressed the opinion is the opinion holder. For example, the sentence *"The story of the film was beautiful, interesting"* expresses a positive sentiment on the film. The sentence has the following factors:

- Opinion holder: author.

- Object: film.

- Feature: story.

- Opinion: beautiful, interesting.

- Polarity: positive.

Researchers face several challenges when performing sentiment analysis. For a comprehensive review on the various challenges faced during SA, we refer the reader to the review article proposed in [1]. In this article, the author groups challenges into several categories. The following are among the most important challenges associated with SA as reported in [1]:

- **Negation**: In linguistics, the presence of negation in the text reflects the polarity of the words, sentence, or the context in which they are mentioned. This is why researchers must process the negation to obtain the correct polarity of the text.

For example, if the text contains “not bad,” the use of bigram tokens will show that the polarity of the text is negative as the prior-polarity of the words (not, bad) is negative polarity while the contextual-polarity of the text is Positive. It is important to point to the fact that the utilization of various n-gram tokens in our proposed NLP-based pipelines for SA has led to a considerable variation in the quality of the employed sentiment classifiers as we demonstrate in the Experimental Setup and Evaluation Chapter.

- **Incorporation of Contextual Knowledge**: Knowing the different facts, contexts and events is one of the important elements in obtaining good results for analyzing and classifying sentiments. For example, the phrase "*The Good Son*" when classified without using global knowledge leads to classifying it as a positive sentence because it contains the word good, while it is an objective sentence as it is the name of a movie. Therefore, the need for global knowledge is one of the challenges facing this field. To address this challenge in the context of our work, we propose embedding sentiment sentences with word embedding techniques that aim at contextualizing the mentions of words in sentences and accordingly assist in better understanding the overall contexts of sentences. We experimentally demonstrate that the exploitation of Word2Vec and GloVe word embeddings has significantly improved the quality of the utilized sentiment classifiers.
- **Spammed Fake Detection**: This challenge is due to the wide spread of websites that contain user opinions about different products and services, and because of the great influence of these opinions on the decisions of consumers wishing to buy these products. There is an increased possibility of these sites being used to display fraudulent opinions, which includes self-promotion or defamation of

other people's products. So, in product review apps, these fraudulent opinions must be detected. This is in order to obtain valid and reliable results when analyzing the feelings of opinions and presenting the results of the product review to the users of the application, so detection of fraudulent opinions is important to maintain the users' confidence in the product review applications.

- **Domain Dependency**: There are many terms and expressions with different polarities depending on the domain in which the texts are mentioned. For example, the word "unpredictable" is a negative description when it comes to a car's steering capabilities. However, it is considered a positive characteristic when describing serial events. As we pointed out in the previous point, such domain dependencies of terms can significantly skew the results and lead to degrading the quality of the exploited sentiment analyzer. This is one of the reasons that called for incorporating external semantic knowledge resources, such as WordNet and other word embeddings, in an attempt to better identify the context of the terms in a given sentence and further embed the sentence with relevant embeddings to improve the quality of the sentiment classifier.
- **Presence of Bipolar Words**: Bipolar words are words that change polarity according to their context. For example, in "*cold pizza*" and "*cold cola*" the same word "*cold*" has different polarity in each sentence; negative polarity in the first sentence and positive polarity in second one. So, the polarity of the word "*cold*" changes depending on the context where it appears. As we experimentally demonstrate in Chapter four, contextualizing words and incorporating this feature as part of the sentiment analysis pipeline can improve the accuracy rates of the utilized sentiment classifiers.

- **Construction of a Huge Lexicon**: To get good results in SA, high order n-grams must be used as we have discussed earlier in this section. However, its usage leads to the formation of very huge dictionaries. For example, if the dictionary size for unigram |D=1000|, the dictionary size for bigrams $|D^2$=1000000, and $|D^3$=1000000000 for trigrams. So, using n-grams with high order is very difficult [1].

1.2 Research Questions and Methodology

In this research project, we study current sentiment analysis techniques and evaluate their effectiveness using publicly available datasets on the one hand, and exploit knowledge encoded in generic knowledge repositories and semantic resources to investigate the impact of expansion and enrichment for sentences with different types of semantic and taxonomy relations on the other hand. We will compare our proposed SA techniques using various NLP pipeline configurations against existing models and attempt to identify the best combination of parameters that can be used for improving the accuracy of the sentiment analysis process. Accordingly, in our research work, we attempt to address the following questions:

a) Will knowledge captured by existing knowledge resources assist in better identifying the semantic and sentiment orientation of sentiment reviews? And what is the impact of utilizing generic semantic repositories on current sentence-level sentiment analysis techniques?
b) Can semantic and taxonomic relations be incorporated for term-based expansion? And what are the different priorities that need to be assigned for expansion candidates (NN, ADJ, ADV, VV or ADJ, NN, ADV, VV)?

c) What is the impact of utilizing different dimensions of GloVe pre-trained embeddings, and Wor2Vec embeddings on the accuracy of sentiment classification using Feedforward neural networks and convolutional neural networks?

d) What is the impact of using different types of activation functions (in the hidden layers of feedforward neural networks and convolutional neural networks) on the accuracy of sentiment classification?

e) What is the best combination in terms of the number of hidden layers in the neural network, the activation function used, the size of the feature maps, and the size of the filter window that should be used to obtain the best accuracy rates?

Our research hypothesis in this context is that expanding words in sentiment sentences with semantic and taxonomic relations will improve the quality of current sentiment prediction techniques. We argue that using different priorities for word type to do sentence expansion will impact the quality of current sentiment prediction techniques. We also argue that varying the configurations of the NLP pipeline will have a significant impact on the accuracy rate of the exploited sentiment analyzer. To support our argument during this research project, we will study the main features that characterize SA techniques and evaluate their effectiveness using large-scale real-world datasets that comprise sentiment reviews. We will focus on the feature engineering process and explore the impact of using semantic and taxonomic relationships to enrich words in sentences and compare this with current models. Moreover, we study the impact of using different priorities for word types (NN, ADJ, ADV, VV or ADJ, NN, ADV, VV) to perform sentence-level sentiment identification.

In attempt of address the abovementioned research questions, this research project follows the Design Science Research Methodology (DSRM) proposed by (Peffers et al., 2007). In this context, the main steps that we will carry out as part of the DSRM implementation are as follows:

- Problem Identification and Motivation: Our goal here is to review relevant literature and explore the main aspects that characterize the features of existing SA approaches and techniques.
- Delineate the Objectives of a Solution: in compliance with this step and as we have pointed out in this section, we have summarized the main objective of our research proposal and highlighted the main research theme and questions that we plan to address.
- Design and Development of the Solution: To implement this step, we have collected a number of publicly-available sentiment datasets that vary in their features, such as whether they comprise short or verbose sentences. As we demonstrate in the Chapter four, we have developed the proposed solution using a number of NLP pipelines and configurations to each of the utilized models.
- Evaluation: We have conducted several experiments using the collected datasets to evaluate the quality of the exploited sentiment analyzers. We have reported the variations in the accuracy rates considering the hyperparameters, as well as NLP pipeline configurations that we prepare for each run.

The main contributions of our proposition are summarized as follows:

- Explore existing SA techniques and evaluate their effectiveness, as well as efficiency using three publicly-available datasets.

- Exploit knowledge encoded in generic knowledge repositories and semantic resources to investigate the impact of enriching sentences with different types of semantic and taxonomic relations, and compare that with existing models. Our focus will be on using synonymous words along with their semantically-related entities to carry out sentence enrichment. We will explore other types of relations, such as holonomy and meronymy and study their impact on the quality of the enrichment process.

1.3 Thesis Organization

The rest of this thesis is organized as follows. In Chapter 2, we introduce the related works and discuss the main features that characterize existing SA techniques. Next, in Chapter 3, we provide the theoretical details and highlight the main NLP tasks that are utilized among the SA pipeline. In Chapter 4, we introduce the experimental evaluation and validation steps that we carried out in order to evaluate a variety of SA pipelines. We also compare our proposed SA model with state-of-the-art techniques and discuss the findings accordingly. In Chapter 5, we present the future works and highlight the future directions of our proposed work.

Chapter Two

Literature Review

2.1 Background

Sentiment Analysis can be defined as the "*practice of applying natural language processing and text analysis techniques to identify and extract subjective information from text*" [1]. It has also been defined as the "*process that automates mining of attitudes, opinions, views and emotions from text, speech, tweets and database sources through Natural Language Processing (NLP)*" [21]. The main aim of SA is to automate the extraction and classification of users' opinions about a given topic of interest. To achieve this goal, researchers exploit various NLP pipelines; coupling text analysis, in addition to using extrinsic resources that can assist in better identifying the semantic orientation of sentiment sentences. Over the past few years, several attempts have been made to develop sentiment analysis techniques that aim at better understanding users' opinions and assist organizations enhance their products and services [26]. Among these techniques are: (i) Lexicon-based [27, 28] and (ii) machine learning approaches [29-33]. For instance, in Reference [18], the authors compared two machine learning approaches, SVM and Artificial Neural Networks (ANNs). Both techniques were utilized to classify the polarity of texts at document level. In their work, the authors reported that the SVM has demonstrated to be less effective than ANNs, especially when using unbalanced data contexts. Nevertheless, experiments showed some limitations when using these two approaches, specifically in terms of the high computational cost required for processing and analyzing large-size documents containing huge amounts of textual content. Therefore, as reported in Reference [19], working on document level is expensive and

inaccurate in some scenarios. This is mainly because the whole document is treated as a single unit and the entire content of the document is classified as positive or negative towards a specific issue. In addition, a document may contain many contradictory opinions on the same topic. This problem also appears when working at the paragraph level, as a paragraph may contain two sentences where each of them has a different sentiment polarity. Therefore, other researchers have focused on sentiment analysis at the sentence level [34, 35]. In Reference [35], the researchers analyzed a dataset of financial reports at sentence level using multiple sentence embedding models. For the purpose of learning about risk statement representations, researchers used models based on SiameseCBow and fastText. However, as reported by Basha and Rajput in [36], among the challenges faced when using a lexicon with predefined term polarities is that they don't reflect any contextual sensitive polarities at the sentence level. To address this issue, Meškelė and Frasincar proposed a hybrid model called ALDONA [22], that combines lexicon based (knowledge-based) and machine learning (neural networks) approaches. The results showed that this approach provides more accurate results than state-of-the-art models and deals with sentences which have a complex structure. In the same line of research, we can also find some researchers who analyzed the sentiment polarity at a word level [23]. In their work, Chen et al., utilized the Reinforcement Learning (RL) method and combined it with a Long-Short Term Memory (LSTM) model to extract word polarities. Their approach succeeded in correctly determining the polarity of the word (positive, negative) in most cases but it also failed in some scenarios.

2.2 Existing Systems

When exploring the related works, we can find out that most of the sentence-level SA approaches consist of two main phases: (i) the first phase is concerned with the feature selection process, and (ii) the second phase is carried out for learning and classification purposes [3]. The main factor that has a major impact on the quality of the produced results is the feature selection. This is demonstrated by the work proposed by Haddi, et. al., [37]. The authors used three different features selection and weighting methods, namely (Feature Frequency (FF), Term Frequency-Inverse Document Frequency (TF-IDF), and Feature Presence (FP)). The Chi-square method was utilized to extract and filter the most relevant features, and the SVM classifier was used to evaluate the system's performance. Results confirmed that the SVM classifier's performance varied significantly depending on input features. In a similar work proposed by Alam et. al. [3], the authors developed a methodology for comparing the performance of different machine learning classifiers (SVM, NB, and MaxE). The authors used different NLP techniques (stemming, stopwords removal and emoticons removal) to extract significant features. They also used sets of n-gram token types (Uni-grams, Bi-grams, Uni-grams and Bi-grams, Uni-gram with part of speech tags), in addition to using Word2vec technique for word vectorization, in an attempt to increase the classification accuracy. Researchers used a dataset that contains 359 documents to apply experiments. The results confirmed that the NLP techniques had a significant impact on the quality of the employed sentiment analyzers. Results confirmed that the NB classifier has outperformed the SVM and MaxE classifiers, however, it is not clear which combination of features should be used in what scenarios i.e., sentiment classification tasks. In a recent work proposed in Reference [38], the authors used a dataset of social

media posts obtained from Twitter to conduct a comparative study between several machine learning SA techniques (Decision Tree, Neural Network, SVM, and W-Bayes Network) at sentence level. The authors used both the RapidMiner software package and Python programing language to make this comparison. In the proposed methodology, researchers used a series of NLP techniques, including normalization, removing stopwords, and tokenization. For feature weighting, the authors used Word2vec and TF-IDF methods. The results confirmed significant variations in the accuracy of the utilized SAs after using the proposed NLP techniques. In [39], the authors proposed a methodology to apply SA on three well known datasets, namely Obama-McCain Debate (OMD), Health Care Reform (HCR) and Stanford Twitter Sentiment Gold Standard (STS-Gold). Researchers compared several classification techniques using a variety of NLP techniques, including removing stop words, stimming, and n-gram tokenization. To test and evaluate their proposed model, they used four popular supervised machine learning classifiers namely: SVM, NB, KNN, and Decision Trees. Results confirmed that n-grams captured by extrinsic resources has proved to improve the quality of the sentiment analyzers.

Traditional machine learning techniques cannot learn features on their own. Thus, the feature engineering process must be used to extract the features manually. Due to this, researchers have recently focused on analyzing sentiment using Deep Learning techniques to learn features automatically. In reference[4], researchers compared a variety of popular machine learning techniques for sentiment analysis, namely SVM, NB, ME and Artificial Neural Network method. They discussed these methods in detail and provided an approximate comparison between them, as well as presented a set of challenges that researchers in the field of sentiment analysis are facing. Specifically,

they demonstrated that the NB method and neural networks are characterized by high accuracy, whereas the SVM method is low in accuracy and the ME method is medium. Due to the great development being made in the field of deep learning and neural networks, they believe that one of the biggest challenges in this field is to study the various neural networks in depth and determine which features are most effective in the field of sentiment analysis. In addition to eliminating the need for process engineering advantage, neural networks have shown good results in sentiment analysis[4]. Due to this, many researchers are focusing their efforts on the use of different neural networks, such as feed-forward neural networks (FNN), convolutional neural networks (CNN), and recurrent neural networks (RNN). Deep Learning approaches generally follow two main phases: (i) the first phase focuses on the word embedding (feature vectorization), while (ii) the second phase is used for learning and classification purposes. Therefore, mathematical representations of texts are one of the most important research areas which researchers focus on. To illustrate the importance of pre-trained word vectors to extract tweets' sentiment, researchers in [40] proposed a deep convolutional neural network to analyze tweets' sentiment. The proposed network has a convolutional layer followed by two fully connected layers with dropout. A sigmoid activation function is used for the first layer, and a tangent activation function is used for the second layer and the softmax activation layer is applied as an output layer. The dropout layers were used with a dropout rate of 0.7 and 0.5 respectively. The researchers generated word vectors using three pre-trained word embedding models that are word2vec, global vectors for word representation (GloVe) and the semantic specific word embedding (SSWE). The authors have used three sets from the SemEval Task 10 challenge for training and testing and used a set of parameters to determine their effect on sentiment analysis accuracy. These parameters are: filter window size, number of hidden units, feature

maps size, patch size and activation function in the convolutional layer. Experiments showed that an improved performance is obtained when using hyperbolic activated tangent units in the convolutional layer, 500 hidden units in the first hidden layer, and 300 hidden unit in the second hidden layer, as well as increasing the size of the feature maps to 300. Further, GloVe word embedding outperformed all other word embedding methods. A traditional feedforward neural network extracts only the current time information and discards the useful information transmitted in the spatial and time arrangement of the data. To address this problem, researchers used the recurrent neural networks (RNN). On the other hand, RNN faces problems such as gradient explosion and gradient vanishing and they, as a result, developed LSTM and GRU network. LSTM can remember long-term information, and its sequential structure is more sophisticated and intelligent than RNN, while GRU is characterized by its high efficiency. Besides, the researchers in [28] suggest a model that combines both LSTM AND GRU to extract sentiment polarity. They used pre-trained word embedding models to create word vectors and trained and tested their model using both IMDB and Review_polarity datasets. Three basic stages make up the model: embedding words layer, neural network layer and output layer. During the first layer, text is converted into vectors in space using the Glove model. The neural network layer consisting of 64 LSTM units and 64 GRU units. The results showed that the proposed model performed better than the RNN model in terms of accuracy.

In our work, we propose utilizing a wide set of hybrid features in an attempt to investigate the best combination of features that lead to producing the most accurate sentiment analysis results. Firstly, we selected some of the most important features and NLP processes (such as n-gram, stemming, lemmatization, and removing stopwords).

And then we tested different combinations of these processes to determine which one produced the best results and with which classifier .We exploit a number of techniques, including SVM, NB, LR, RF, FNN and CNN. Our main goal in this context is to explore the impact of utilizing a variety of NLP pipeline phases on the quality of each of the employed techniques. We argue that the quality of a sentiment analyzer can be further improved depending the combination of NLP-based processes involved.

Table 1:Classification of the studied sentiment analysis approches.

Reference No.	Approach	Category	Goal of the system	Implementation techniques	Main Characteristics
[18]	Machine Learning Approach	Document level	Authors compared between two machine learning approaches SVM and Artificial Neural Networks (ANNs).	Artificial Neural Network (ANN) Support Vector Machine (SVM)	-limitations: high computational cost required for processing and analyzing large-size documents containing huge amounts of textual content.
[19]	Machine Learning Approach	Document level	Authors study the impact of using set of pre-processing techniques on classification accuracy for financial news sentiment. Researchers used two approaches for sentiment analysis namely: event-based and reader-based approaches.	SVM, and Naive Bayes classifier.	-The reader-based approach proved accurate in predicting financial news sentiment. Limitations: -working on document level is expensive and inaccurate in some scenarios.
[22]	Hybrid Approach	Sentence level	Authors proposed a combined model by combining Lexicalized Domain Ontology with machine learning (neural networks) techniques to classify sentiment sentences, called ALDONA.	Deep Bidirectional Gated Recurrent Unit (DBGRU).	-Proposed approach provides more accurate results than state-of-the-art models and deals with sentences which have a complex structure. Limitations: -Very complex model.
[23]	Neural Networks	Word level / Sentence level	Authors proposed model combines Reinforcement learning (RL) with LSTM classifier called Word-level Sentiment LSTM (WS-LSTM) to extract words polarity in sentence. And next use them to find sentence polarity.	RL with LSTM.	-Proposed model achieved good results at the sentence level. Limitations: -Although the model achieved good results at the sentence level, it is not accurate at the word level. -Applying RL to classifying sentences is a complex and hard task.
[37]	Machine Learning Approach	Document level	Researchers focused on explaining the importance of feature extraction phase on classification accuracy. So they examined the impact of using set of NLP techniques to extract features, and 3	SVM.	-Proposed model achieved good results at the document level.

			types of feature vectorization on classification accuracy. Researchers used Chi-square method to extract and filter the most relevant features		
[3]	Machine Learning Approach	Document level	Authors studied the impact of using set of NLP techniques with n-grams for futures extraction. In addition to using Word2vec for features vectorization on classification accuracy.	SVM, NB, and MaxE.	-Proposed model achieved good results at the document level Whereas accuracy improved after applying proposed NLP processes. Limitation: -Researchers used only 359 document in experiments.
[38]	Machine Learning Approach	Sentence level	Authors examined the impact of using set of NLP techniques. In addition to using Word2vec and TF-IDF For features vectorization on classification accuracy.	Decision Tree, Neural Network, SVM, and W-Bayes Network	- Proposed model achieved good results at the sentence level.
[39]	Machine Learning Approach	Sentence level	Authors studied the impact of using set of NLP techniques with n-grams for futures on classification accuracy.	SVM, NB, KNN, and Decision Trees	-Preprocessing in sentiment analysis was extensively discussed by the authors.
[4]	Machine Learning Approach	Sentence level	Researchers discussed four popular methods for sentiment classification in detail. And provided an approximate comparison between them, as well as presented a set of challenges that researchers in the field of sentiment analysis are facing	SVM, NB, ME and Artificial Neural Network	Limitation: -Researchers do only approximate comparison between machine learning classifiers.
[40]	Neural Network (Deep Learning)	Sentence level	Researchers examined the impact of using Word2Vec, GloVe, and SSWE approaches for features representation on CNN sentiment classification accuracy.	Deep Convolutional Neural Network	- Proposed model achieved good results at the sentence level. -Researchers claimed that the proposed a convolutional network with multiple filters that had never been

					used before.
[28]	Neural Network (Deep Learning)	Sentence level	Researchers suggest a model that combines both LSTM AND GRU to extract sentiment polarity. In addition to using GloVe pre-trained word embedding to create word vectors.	LSTM AND GRU	-The proposed approach proved accurate in predicting sentiment. -In the proposed approach, long-term information is learned and RNN's defects are compensated. - proposed approach utilizes computing resources more efficiently.

2.3 Summary

In this chapter, we presented a literature review of sentiment analysis, its approaches, and highlighted the main features that characterize each of the discussed methods. Among the approaches we have discussed are: lexicon-based approaches and machine learning approaches. In addition, we conducted a comprehensive analysis of these approaches and outlined their strengths and weaknesses according to a set of evaluating criteria.

Chapter Three

Proposed Methodology and Theoretical Framework

The process of identifying the sentiment orientation of a given sentence passes through several phases, starting from data collection and preparation phase, to feature extraction, training the sentiment classifier, and finally testing its quality. In the next sections, we introduce the details of each of these phases.

3.1 Data Acquisition and Cleaning

Sources of sentiment sentences can vary on the Web. Twitter and other social media websites are among the most commonly referred to sources [26, 39]. For experimental evaluation purposes, we use the well-known IMDB movie reviews dataset, which is publicly available at Kaggle[1]. We also used the Sentiment140[2] dataset, which contains 1,600,000 tweets collected from twitter API. In addition, we collected the 10,662 LightSide dataset, and 300 other generic movie reviews from Twitter. They all contain sentiment sentences, half of which are positive and the other half are negative. A set of examples of dataset content is presented in the Table 2. The first task after the data acquisition is to clean the raw data as it normally contains some special characters, including hashtags, consecutive white spaces, URLs, and unnecessary symbols. In addition, there is a set of emoticons that we cleaned using a pre-defined set of icon representations. After this step, we proceed to the second phase of data processing, that is tokenization and feature extraction.

[1] https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews/
[2] https://www.kaggle.com/kazanova/sentiment140

Table 2:A set of examples of dataset content

Dataset	Positive	Negative
IMDB	If you like original gut wrenching laughter you will like this movie. If you are young or old then you will love this movie, hell even my mom liked it. Great Camp!!!	Besides being boring, the scenes were oppressive and dark. The movie tried to portray some kind of moral, but fell flat with its message. What were the redeeming qualities?? On top of that, I don't think it could make librarians look any more unglamorous than it did.
Sentiment140	happy to spend time with the family.	is upset that he can't update his Facebook by texting it... and might cry as a result School today also. Blah!
Sentiment sentences	offers much to enjoy . . . and a lot to mull over in terms of love , loyalty and the nature of staying friends	a very stylish but ultimately extremely silly tale . . . a slick piece of nonsense but nothing more .

3.2 Tokenization and Feature Extraction

At this stage, a set of features is extracted from textual information encoded in sentiment sentences. As we discussed in the literature review section, there are several SA models that employ different feature extraction techniques. For instance, considering the n-gram tokenization process, choosing a specific size for n-gram tokens affects the overall's accuracy of the sentiment analyzer, especially when dealing with sentences that contain negations, such as "not good" or "not very good" as reported in [26, 32]. In this context, we can see that using unigram features to process a sentence with "not good", will separate the word "good" from the negation word "not". Similarly, using the same bigram features in the second scenario, the negation word "not" and the word "good" will be separated into two different tokens, these are: "not very" and "very good". In terms of computational cost, if the dictionary size when using unigrams is |D|, it will become $|D|^2$ when using bigrams, and $|D|^3$ for trigrams. Therefore, using n-grams with *n* greater than 3 is very time consuming. In an earlier study in 2004, Pang et. al.

have showed that unigram features are more significant than bigrams when performing emotional classification for movie reviews [41]. On the other hand, other studies showed that coupling bigrams and trigrams based on extrinsic semantic resources provided better results than unigrams alone [26, 42]. As reported by Maree and Eleyat [26], incorporating high-order n-grams such as trigrams that can be acquired based on external knowledge resources captures a sentence contextual information more precisely than other unigram or bigram-based tokenizers. Accordingly, and in light of these conclusions, it is crucial to experimentally evaluate the utilization of n-gram tokenization as part of the large-scale SA process. In the next sections, we provide details concerning each of the proposed NLP pipeline phase.

3.2.1 Stopwords Removal

Raw text of sentiment sentences usually contains a lot of words that are frequently repeated. Such words have no significant contribution in determining the sentiment orientation of sentences. Moreover, they may have a negative impact on determining the polarity of sentences as reported in [3, 19]. These words are referred to as stopwords. Removing such words is a crucial step in the data pre-processing phase. Some researchers divided stopwords into two types, these are: 1) general stopwords and 2) domain-specific stopwords [43]. General stopwords such as (the, a, on, of, ...) are considered as stopwords regardless of the domain of the dataset. On the other hand, domain-specific stopwords are those words that appear to be of little significance in deriving the meaning of a given sentence in a particular domain [44]. Such words can be identified through the utilization of the TF-IDF model in the same manner as described in [26, 44]. For example, as far as the movie reviews dataset is concerned, words such as, movie, actor, and film appear very frequently in the sentiment sentences. These

words are specific stopwords to this field and can be regarded as stopwords when they appear redundantly in sentences. To remove stopwords from a given dataset, researchers use a list of pre-defined words that is updated frequently. This process is error-prone and requires efforts and time to prepare, therefore, the automatic identification and removal of stopwords in this context remains more superior [42].

3.2.2 Word Stemming and Lemmatization Processes

Stemming is one of the common morphological analysis processes that is mainly concerned with the removal of derivational affixes in the hope of achieving a common base form for a given word [26, 45]. The goal of this process is to reduce inflectional forms and achieve a common base form for words in sentiment sentences. As reported by Haddi et al., the importance of this step lies in reducing text dimensions for sentiment classifiers [37]. In this context, the number of dimensions representing different words in the text will be reduced. As such, this allows representing a word, in addition to its inflectional forms, as one word [45]. For instances, the words: *product*, *producer*, *production* and *produce* will be reduced to *produc* [19]. This reduction in word dimensions helps also to correctly determine the weights of the words and their importance in the text. In our work, we have exploited and evaluated two of the most common stemming techniques that are employed in the context of sentiment analysis. These are: Porter [46] and Snowball stemmers[47]. Porter stemmer is one of the oldest and most popular stemming algorithms that is still utilized as part of various sentiment analysis pipelines [6, 11]. It is a rule-based stemmer that consists of five phases of word reductions that are applied sequentially. Each rule group is applied to the longest suffix. One of main limitations of this stemmer is that it may return the same form for two words with different sentiment polarities. For instance, the word "*Dependability*" which

has a positive polarity and the word "*Dependent*" which has a negative polarity are both stemmed to "*Depend*". Similarly, Snowball stemmer, which is was developed based on Porter's stemmer, shares many of the limitations that are inherent in Porter stemmer. However, this technique has demonstrated to produce more promising stemming results against those produced by Porter. In addition, as reported by Rao in [7], the performance of Snowball is higher than Porter and it can be applied on many languages other than English.

Similar to stemming, lemmatization is another common morphological analysis process that has widely utilized for sentiment analysis purposes. It is mainly used to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma*. One of the main advantages of lemmatization is that a returned lemma can be further processed to obtain synonyms, as well as other semantically-related terms from extrinsic semantic resources such as WordNet and Yago. For example, words, such as: *educational*, *educator*, *educators*, and *education* are lemmatized into *educate*. Unlike stemmers which will stem these words into: *educat*, the lemma *educate* can be submitted to WordNet, for example, to find synonyms, such as *train*, *school* and *cultivate* which can in return assist in deriving the context of the word in a given sentence [11, 48]. While both operations aim to reduce the inflectional forms of words, stimming relies on a list of common prefixes and suffixes, lemmatization relies on morphological analysis. which means lemmatization is more accurate. For example, leafs and leaves lemmatized to leaf while leafs stemmed to leaf and leaves stemmed to leav.

3.3 Features Selection and Sentiment Classification Techniques

Several research works have focused on feature selection from text, namely for sentiment analysis purposes [26, 49-56]. The main features that have captured the focus of these works are term frequency, inverse document frequency, n-gram based tokens, and POS-tags of words extracted from the text of sentiment sentences. The aim of the feature selection process in this context is to reduced feature vectors to comprise relevant features in an attempt to improve the computation speed and increase the accuracy of the sentiment classification methods. However, little attention has been given to the latent semantic aspects of words encoded in sentiment sentences [26, 57, 58]. In addition, the composition of the best features that significantly contribute to producing the most accurate sentiment classes has not been given sufficient attention. Accordingly, in the context of our research work, we attempt to investigate the impact on combining multiple features, including semantically-related features, on the quality of a number of machine learning based sentiment classification methods. After features selection, the features are converted into vectors. The mathematical representations of the texts contribute greatly to the accuracy of the neural network results. Two main approaches can be used to represent features, namely discrete representations using One-hot feature vector approach and distributional representations using dense vectors produced by Global Vectors (GloVe)[3] or Word2Vec [4, 59]. During the first part of our experiments. In particular, we employ the well-known *TF-IDF* feature weighting technique with the combination of n-gram tokens and semantic features extracted from WordNet. As such, a sentiment sentence is assigned a weight based on Eq. 1. The unit we use to represent each sentence is D. We decided to use the variable D to denote

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

sentences in the dataset to maintain consistency with the general terms used to define the *TF-IDF* equation.

$$\mathrm{TF-IDF(t)} = (1 + \log_{10} \mathrm{TF(t)}) * \log_{10}(\mathrm{N/DF(t)}) \quad (1)$$

Where,

- N: is the number of sentiment sentences in the corpus.
- DF: represents the number of sentences that contain term *t*.
- TF: is the number of occurrences of term *t* in sentence D.

It is important to point out here that the *TF-IDF* feature weighting technique is applied on composite features extracted from pre-processed text of sentiment sentences.

There are many drawbacks to using discrete representations, including the huge feature vector size. This wastes space and makes algorithms more complex, resulting in the curse of dimensionality. In addition to the inability to show connection between words. To overcome these limitations, the researchers proposed an approach that uses dense vectors to represent features. Using the distributed approach, words are represented as n-dimensional dense vector. Where similar words are represented by similar vectors. in second part of our experiments, we employ the GloVe and Word2Vec approaches for features vector representation. Relying on distributional representations. Researchers used different approaches to represent words, where in reference [60] they used random initialization for the word vectors. And then allowed the model to learn the most accurate representation of the words. However, according to the author, this approach was ineffective. Recently, researchers utilized unsupervised learning in order to learn word representations from large text corpora [61]. This approach provides researchers

with pre-trained vectors that can be used to perform various NLP tasks. Word2Vec and GloVe pre-trained embeddings are the most efficient and effective ways to convert words into corresponding dense vectors [4]. In our experiments, we employed both of them.

A wide range of machine learning (supervised, semi-supervised and un-supervised) techniques have been developed and utilized for sentiment analysis purposes [13, 29]. The input to any of the utilized methods is a set of composite features generated based on the techniques discussed in the previous section. As such, the quality of the utilized sentiment classifier will be highly dependent on the quality of the selected features, especially when dealing with highly skewed datasets. Traditionally, there are various types of classifiers that have been commonly used for sentiment analysis. These are: Support Vector Machine (SVM), Naïve Bayes (NB), Random Forest (RF) and Logistic Regression (LR). It is important to point out that these classifiers are considered among the most commonly used and robust classification methods that have been successfully applied for sentiment analysis as reported in Reference [19]. They have demonstrated highly accurate classification of sentiment sentences in various application domains [18]. In addition, they tend to be less affected by noisy terms, especially when compared with ANN-based sentiment classifiers when the data imbalance increases. Furthermore, the time required to train an ANN is usually much higher than the that required for training these types of classifiers.

3.3.1 Support Vector Machines (SVM)

One of the most common and successful classifiers is the SVM classifier [3, 20, 21, 37]. It is a "*linear learning method that finds an optimal hyperplane to separate two classes*

as a supervised classification approach" [18]. Several researchers confirmed that SVM is accurate at analyzing sentiments [18]. Also, it has a strong theoretical basis, where it aims to create a hypothesis that leads to obtaining the minimum generalization error value. In this method, nonlinear inputs are represented by vectors in space, which form two groups of classes, as shown in Fig. 2.

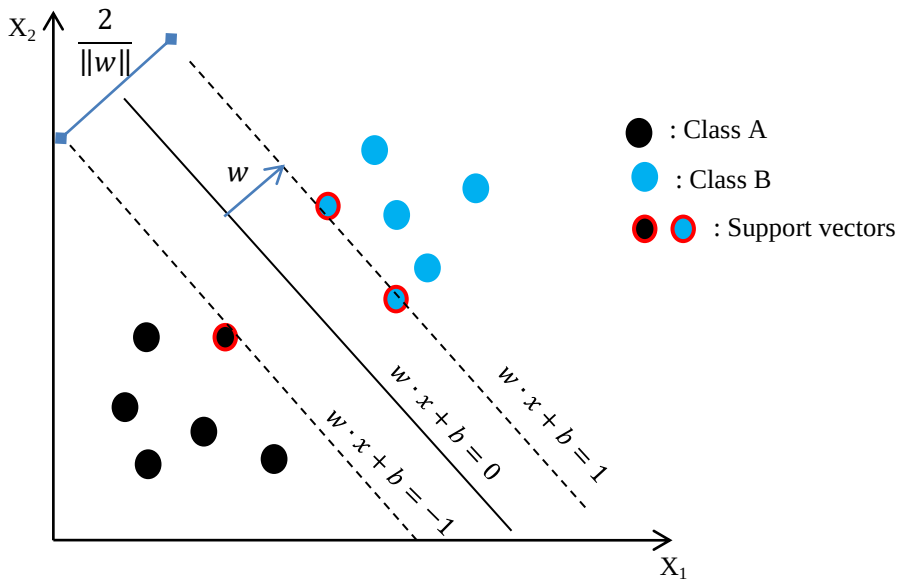


Figure 2: SVM Hyperplanes

This method separates these two sets of nonlinear data by finding the best surface separating them. With the aim of maximizing the margin between the two classes, as maximizing the margin reduces indecisive decision [21].

The hyperplane which separates the two classes (positive and negative classes) calculated as follow:

$$a\vec{x} + b = 0 \tag{2}$$

Where $\vec{x}$ is n-dimensional input vector and $\vec{y}$ is output value (positive, negative).

($\overrightarrow{x_{in}}$, $\overrightarrow{y_i}$) together form a training dataset and a_i is a Lagrangian multiplayer. And b is an intercept of the hyperplane equation

$$\vec{x} = (x_{i1}, x_{i2}, x_{i3} \dots, x_{in}) \quad (3)$$

$$\vec{y} = (y_1, y_2, y_3 \dots, y_i) \quad (4)$$

Using eq.2 hyperplane is defined using weight vector $\vec{w}$.

$$\vec{w} = (w_1, w_2, w_3 \dots, w_n) \quad (5)$$

Prediction classes determined at testing phase by following equations:

$$\text{Positive if} \quad \vec{w} \cdot \vec{x} + b \geq 0 \quad (6)$$

$$\text{Negative if} \quad \vec{w} \cdot \vec{x} + b < 0 \quad (7)$$

Increasing classification accuracy is achieved by maximizing the margins to obtain the optimal hyperplane. As shown in Fig.2, it is easy to separate the dataset, but what if the dataset is more complex, as shown in Fig.3.

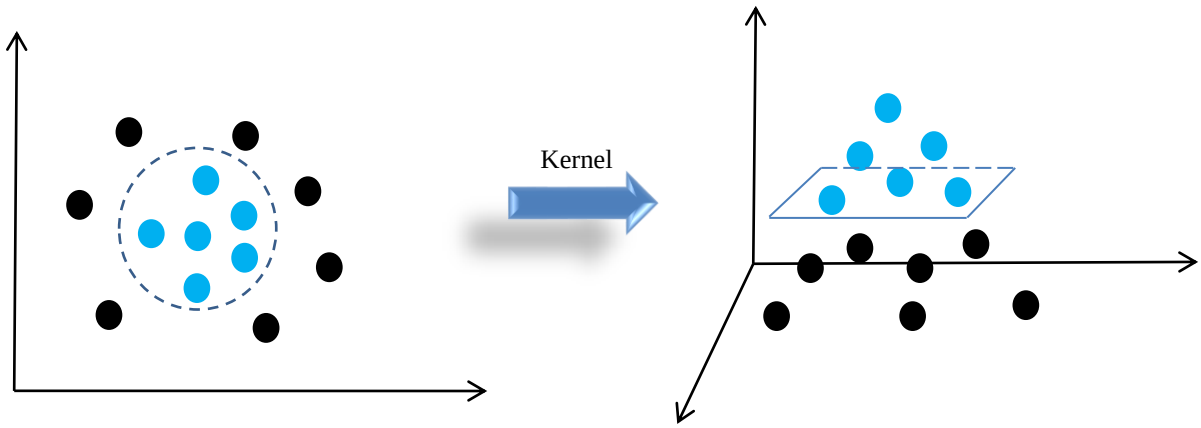


Figure 3:SVM kernel.

The data in Fig. 3 are nonlinear. So the two sets of data cannot be separated linearly. SVM uses mathematical functions called kernels in order to solve nonlinear data problem. Using kernels, nonlinear data is transformed to linear data using higher dimensions. SVM algorithm use different types of kernel functions like, linear, radial basis function (RBF), and sigmoid. A linear kernel is used when the data being

processed is linear. It is the most simple and fastest type of kernel. This kind of kernel is used for classifying texts. And when a large number of features are present. RBF is a Gaussian kernel function and it is one of the most popular and commonly used kernel functions in SVM. RBF kernels are generally used when processing data that is nonlinear and when data are unknown. For the sigmoid kernel, it is widely used in neural networks. In order to obtain good classification accuracy, as well as to reduce classification errors, the appropriate kernel must be chosen so that the data is converted in a correct manner. In addition to tuning some important factors, including: regularization parameter, and gamma parameter. The regularization parameter is a balance between misclassifications and margin width. When its value is high, the SVM approach chooses a smaller margin width , so the algorithm is more stringent with the misclassifications. Which means higher classification accuracy. And When regularization parameter value is small, the SVM approach chooses a Larger margin width, so the algorithm is more stringent with the misclassifications. But it must be emphasized that an increase in the value of the regularization parameter does not necessarily mean an increase in the accuracy of the classification, as more increasing in the parameter value may lead to over-fitting. gamma parameter used with Gaussian RBF kernel, it is reflect the number of samples which will be used to be support vectors. gamma parameter used with Gaussian RBF kernel, it is reflect the number of samples which will be used to be support vectors. where larger gamma value means only nearest samples will used as support vectors. and smaller gamma value means far samples will also use as support vectors.

3.3.2 Naïve Bayes (NB)

NB is a probabilistic method that depends on the theories of probability and statistics in the classification process [20]. Bayes theory is based on the principle of independent occurrence of features [18, 34]. As it does not take into account the location of terms in a sentence, each feature is considered independently. Compared with SVM, Maximum Entropy, Decision Tree, and Random Forest, this method is characterized by its simplicity, effectiveness and short time spent in data training which results in low computational cost [3, 20]. In terms of performance in some studies, SVM outperformed NB [18], but it showed good results in other applications [3], however, it is very sensitive to the selection of features [3]. Therefore, focusing on the features selection process is very important to improve performance in addition to analyzing large dataset [20]. Bayes theory is represented by the following equation:

$$P(class|features) = \frac{p(class)*p(features|class)}{p(features)} \quad (8)$$

Where $p(class)$it is the conditional probability of a class in corpus, $p(features|class)$ is the conditional probability of a feature related to a class and $p(features)$ is the Likelihood of a feature.

3.3.3 Logistic Regression (LR)

Logistic Regression is a supervised learning algorithm. It is one of the linear logarithmic classifiers that classifies inputs into two classes. It's a regression model that uses sigmoid function to obtains polarity of sentences. The function is mathematically expressed as shown in Eq. 9. and it output probability between 0 and 1. As shown in Fig. 4, this model is represented by the following equations:

$$h\left(x^{(i)},\theta\right)=\frac{1}{1+e^{-\theta^{T}x^{(i)}}} \quad (9)$$

Where *x* denotes features, θdenotes parametrs and i denotes data points. T denotes threshold and its value equals 0.5as the result of the dot product between Theta *transpose (θ)and x equal to zero.* Positive and negative classes are determined by following equations:

$$\text{Positive} \quad \text{if} \quad \theta^{T}x^{(i)} \geq 0 \quad (10)$$

$$\text{Negative} \quad \text{if} \quad \theta^{T}x^{(i)} < 0 \quad (11)$$

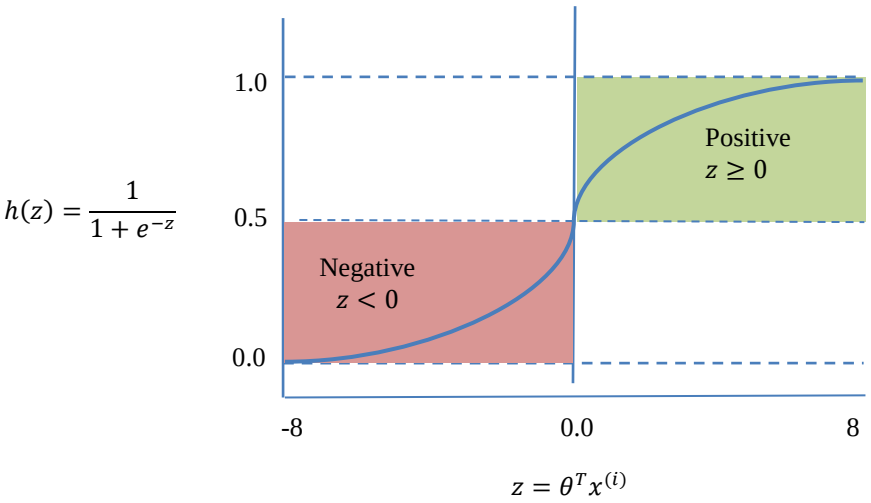


Figure 4: Logistic Regression.

3.3.4 Random Forest (RF)

Random forest is an extension of the decision tree model, where polarity is determined by building several trees instead of creating one tree. This model combines between two main concepts, namely Bagging and random subspaces[12, 20]. In the Bagging step, training dataset is randomly separated into smaller groups of data called bootstrap samples. These bootstrap samples will later be used to construct classification trees where each sample is used to construct a decision tree that produces its own prediction, as shown in Fig. 5. The majority voting mechanism is then used to aggregate their

individual results and generate final output. In the random subspaces step, two parameters are defined, namely the number of trees to be built and the number of random variables used to split the nodes, then these parameters values are adjusted with the aim of improving the results and reducing the error value. This approach is characterized by its ability to classify large amounts of data accurately [12]. Moreover, in contrast to decision trees, it is more flexible and accurate, with a reduced risk of overfitting.

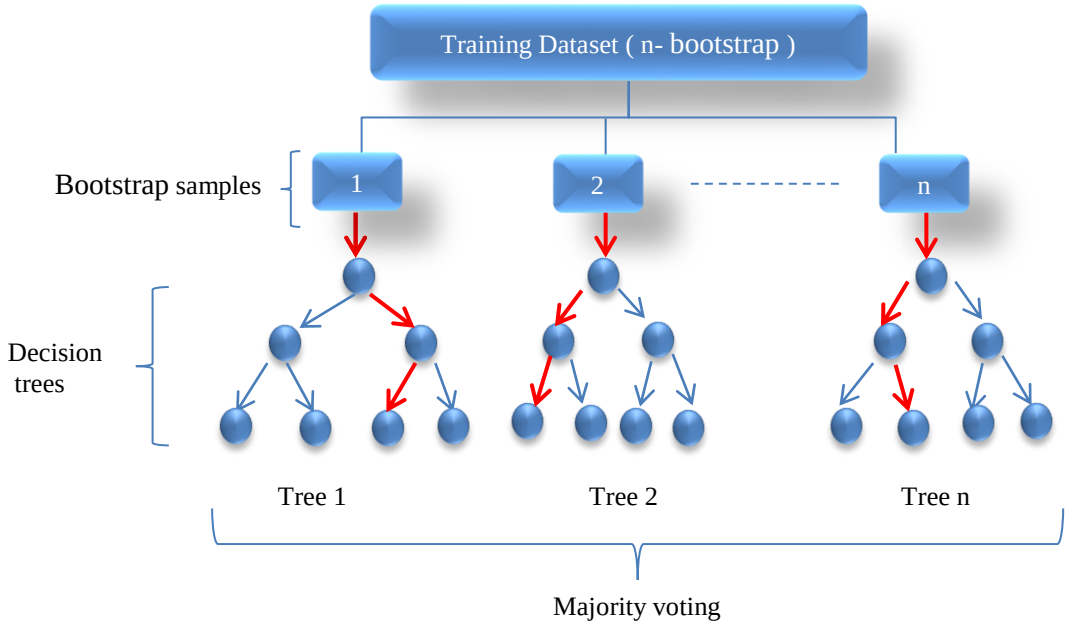
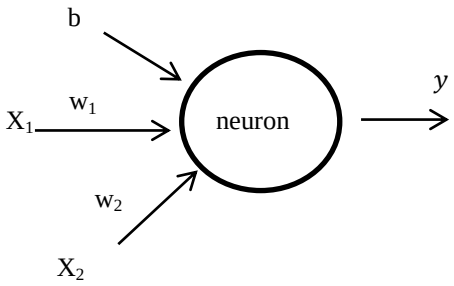


Figure 5: Random Forest

3.3.5 Artificial Neural Networks

An artificial neural network can be defined as "*a mathematical model for the simulation of a network of biological neurons (e.g., human nervous system). It simulates different aspects related to the behavior and capacity of the human brain*" [62]. Neural networks consist of basic units of computation each called a node or a neuron. A neural network consists of a set of layers, each layer containing a set of nodes. Terminal nodes receive data (x) and each entry has a weight (w) that is determined according to the importance

of the entry compared to other inputs. Internal node implements an activation function f like (sigmoid, ReLU, tanh, ...) on the weighted sum of its inputs.



$$y = f(x_1 . w_1 + x_2 . w_2 + \mathrm{b}) \quad (12)$$

Figure 6: A single neuron

Eq. 12 illustrates how value Y is computed as the output of the neuron. Activation Function f is a non-linear function which introduces non-linearity into neuron output. The reason why this is important is that most real-world data is non-linear, and we need neurons to learn non-linear representations. Many activation functions are used in neural networks, and perhaps the most commonly used are sigmoid, hyperbolic tangent, and Rectified Linear Unit Relu activation functions. A sigmoid activation function adjusts input value into a 1 to 0 range as shown in Fig. 7. While the tangent function adjusts the input values into the range [1,-1]. ReLU activation function replaces negative values with zero. Parameter b stands for the Bias and it is a constant value that allows shifting the activation function in order to better match the prediction with the data

The three major types of neural networks are feedforward neural networks (FNNs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs). In our experiments, we focused on feedforward neural networks, and convolutional neural networks.

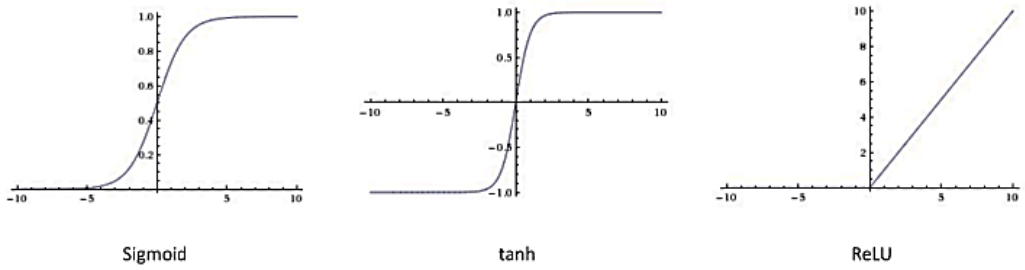


Figure 7: Diffrent Activation Function.

1- Feedforward Neural Network (FNN)

The feedforward neural network is the simplest type of artificial neural network. It consists of several neurons arranged in layers connected by connections with weights attached to them. As we can see in Fig. 8. There are three types of layers in a feedforward neural network: input layer, hidden layer and output layer. In the input layer, input nodes receive input data and pass it on to the next layer without performing any arithmetic operations. Hidden layers are located behind input layers, so there is no direct connection to the data source. These layers form a link between the input layer and the output layer and they usually perform calculations on the data and passes it to the output layer. A neural network may or may not have hidden layers. FFNs with hidden layers are called Multi-Layer Perceptron (MLP), while a network without any hidden layers is known as a Single Layer Perceptron. In a feedforward neural network containing hidden layers, data passes through them in one direction (forward). The output nodes in the output layer perform arithmetic operations on the data they receive from the network and then pass output to the outside world.

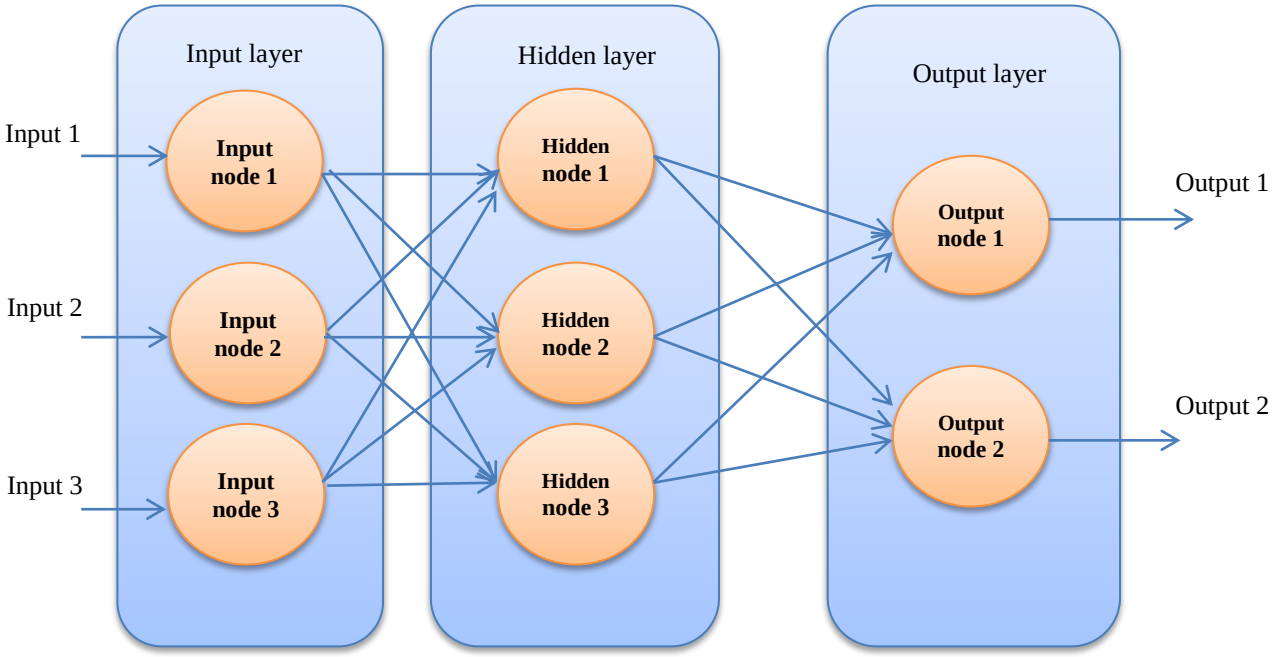


Figure 8:An example of Feedforwared Neural Network

2- Convolutional Neural Network (CNN)

The Convolutional Neural Network is likely one of the most widely used neural networks for sentiment analysis. This approach automates feature generation; therefore, we don't need to deal with feature engineering. And it considers both word order and context when analyzing a sentence. So, it is flexible and accurate. As we can see in Fig. 9. CNN consists of several layers namely convolution layer, pooling layer and fully connected Multi-Layer Perceptron (MLP). Layer one in this network is a convolutional layer and its output is called a feature map. Moreover, this layer uses a kernel which acts as a sliding window over the data where each piece of data in the convolutional layer is represented as one unit in the feature map; therefore, it acts as a feature extractor. By applying the max feature or averaging adjacent features on the feature map, the pooling layer reduces them to a single unit. The outputs from this layer are passed into a feedforward neural network.

The general principle of neural networks is learning from errors, so the principle of neural networks can be summarized as receiving data, making predictions, comparing the predictions with the real values and adjusting the weights to predict with greater accuracy next time. These steps lead to the neural network being trained. The next and final step is the testing model.

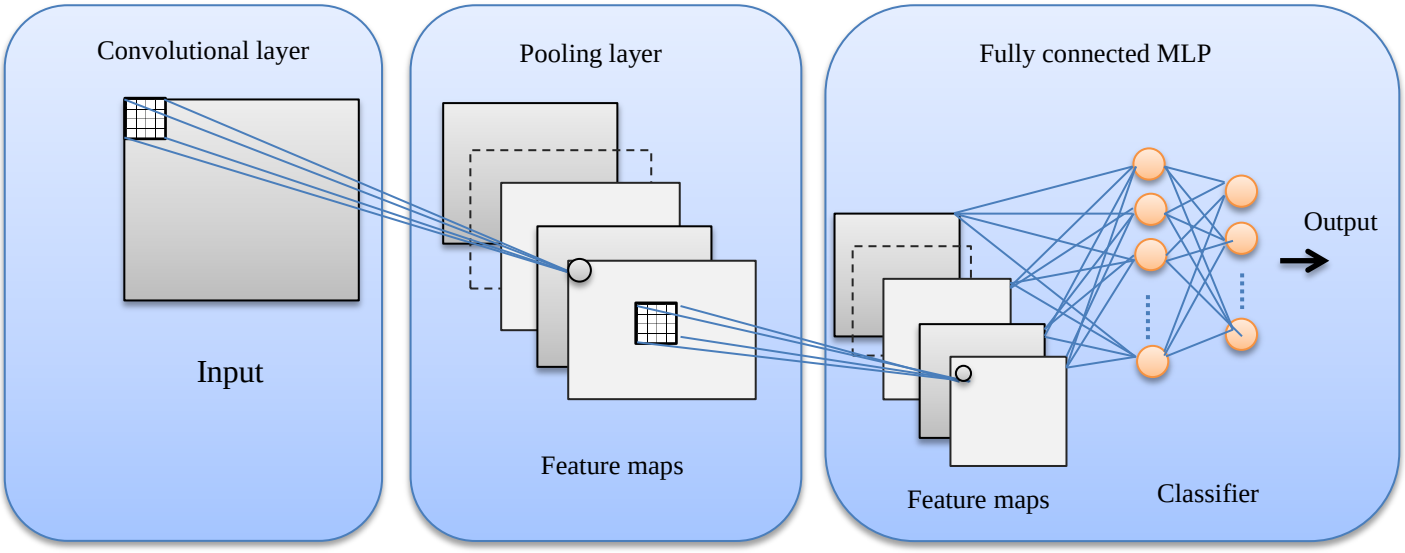


Figure 9: Convolutional Neural Network

There are two fundamental steps to training a convolutional neural network: Forward Propagation and backward propagation. In Forward Propagation, data is received, then mathematical processes are applied to received data, and eventually output is generated. As for Backward Propagation, we have the error calculation and parameter update.

-Forward Propagation phase:

First, we have the convolution phase which involves sliding the filter over the input matrix and multiplying each pixel of the matrix by the parallel pixel of the filter as shown in Fig. 10. To introduce non-linearity to the network, the activation function is applied to each convolved value. For example, If the ReLU activation function is used, each negative value will be converted to zero, while positive values will be saved.

Pooling is then performed, for example, if max pooling is used, then the maximum value will be taken for each filter, as shown in Fig. 11. A fully connected multilayer perceptron is then used to predict output.

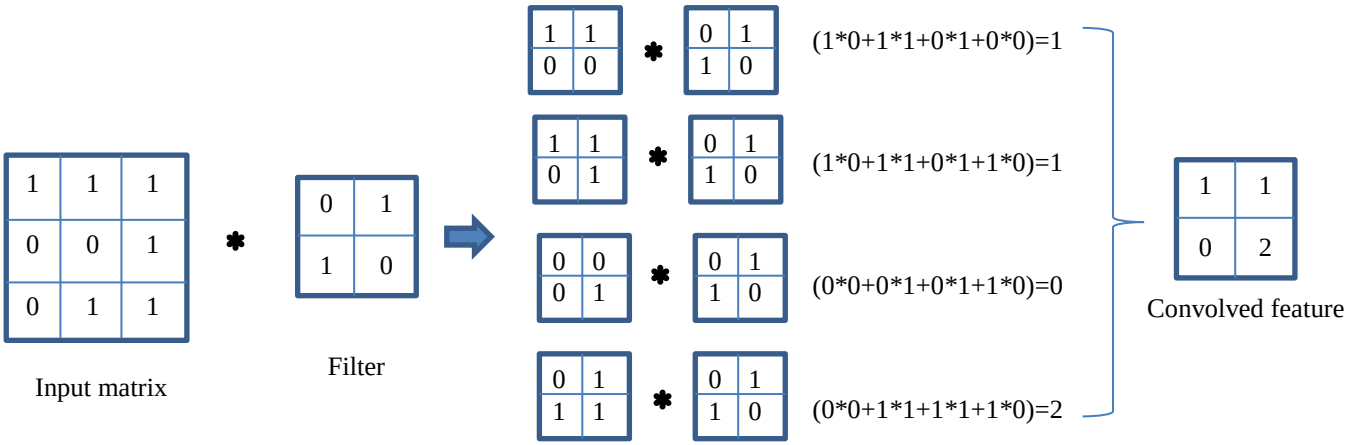


Figure 10: Convolution process.

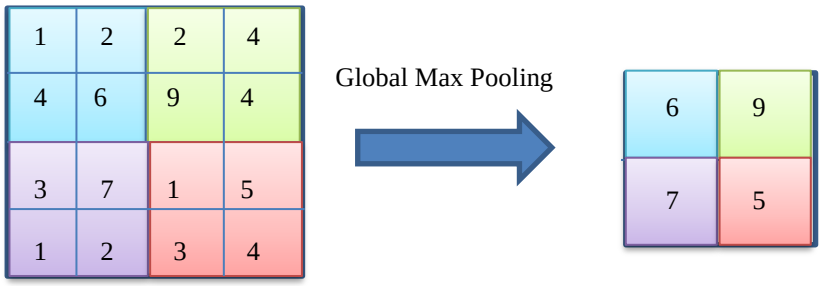


Figure 11:Global max pooling.

-Backward Propagation phase:

In forward propagation, network parameters such as weights and bias are randomly initialized. Model tries to update network parameters in order to improve accuracy during backward propagation phase based on using Gradient Descent Algorithm as shown in equation below:

New parameter = old parameter - (learning rate * gradient of parameter)

The learning rate usually specifies how much the old variable value has changed, while the gradient specifies how it changed (increase or decrease).

3.4 The Proposed Sentiment Analysis Pipeline

For sentiment analysis, we have proposed three models in this section. Also we explained the details of proposed pipeline using diagrams and pseudo code algorithms as depicted by the following figures.

3.4.1 First Proposed Sentiment Analysis Pipeline

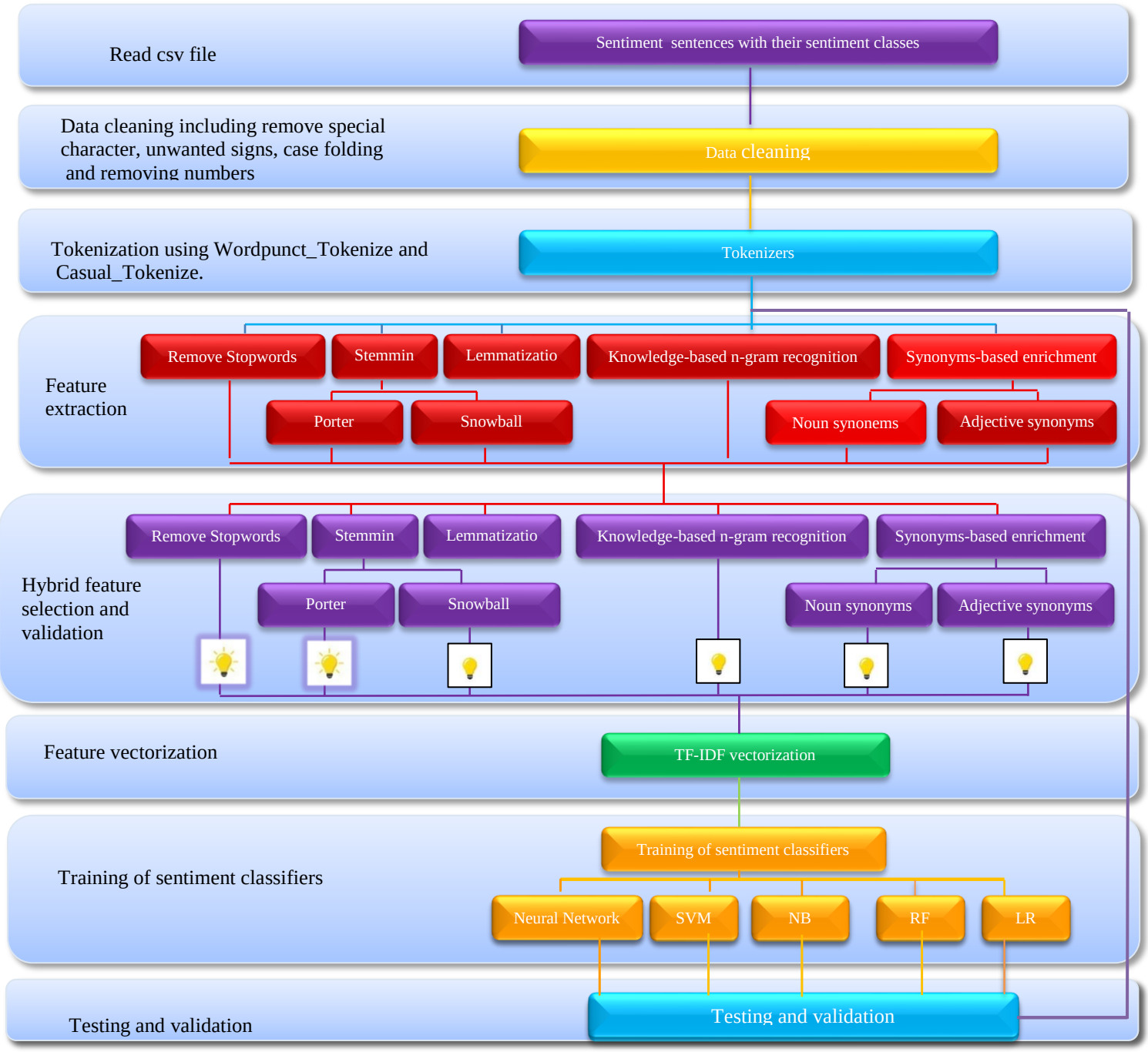


Figure 12: Phases of the first proposed sentiment analysis pipeline.

Figure 12 shows the phases of the first proposed sentiment analysis pipeline. First, we read the dataset containing sentences associated with their sentiment classes. Then, we perform data cleaning by removing special characters, unwanted signs, case folding and removing numbers. After this step, we apply text tokenization using two types of tokenizers *Wordpunct_Tokenizer* and *Casual_Tokenizer*. Then, the produced tokens are stemmed using two types of stemmers, Porter stemmer and Snowball stemmer. Lemmatization is also another step that we apply on the tokens using WordNet lematizer, WordNet knowledge-based n-gram recognition, and synonyms-based enrichment techniques, noting that we use synonyms of nouns and adjectives for term enrichment purposes. Then, in the next step, we select and activate a hybrid set of features for TF-IDF vectorization. We use the produced vectorization results for training each of the sentiment classifiers. The details of these steps are presented in Algorithms 1 and 2. First, using Algorithm1, we perform data cleaning (steps 2 to 6). Step 10 presents the tokenization function. In step 11, we invoke the feature selection and activation function. Using this function, we select and activate specific feature extraction functions, such as stemming, lemmatization, n-gram recognition, …, etc. to compose the hybrid features set. In step 13, we submit the composite features for TF-IDF vectorization. Next, we train the classifier at step 18 and the algorithm carries out the testing and validation task and calculates accuracy of the results at step 19. The output of Algorithm 1 is a list of sentiment orientation predictions associated with accuracy metrics for each input dataset.

Algorithm 1. Identifying The Sentiment Orientation Algorithm

INPUT: file containing the list of sentiment sentences with their sentiment polarity. Special Stopwords list, S < Stopwords >. And list of unwanted signs(%,#,%,|), L < unwanted signs >
OUTPUT: list of sentiment orientation predictions for test set reviews and algorithm accuracy.

PREDICTSENTIMENT(R,S,L) : sentiment orientation, algorithm accuracy
1: reviews ← R <r_i, p_i> // list of sentiment sentences with their sentiment polarity

2: while reviews ≠ NIL do:
3: for each word in r_i do
4: do **DELETE**(word) if word ∈ (L or numbers) .
5: word ← **CASEFOLDING**(word)
6: end for
7: F ← < > // F: list of features
8: while Next r_i in reviews do
9: T ← < >// T: list of tokens for r_i
10: T ← **TOKENIZE**(r_i) // tokenization using Wordpunct_Tokenize or Casual_Tokenize
//Next function (Step 11) constructs composite feature using the helper functions that start at step
21.
11: F ← **FEATURESELECTIONANDACTIVATION**(T)
12: class ← < p_i>
13: vectoriser ← **TFIDFVECTORIZER**(F)
14: TrainSet ← 0,70 × (vectoriser, class) //Split vectors in to TrainSet and TestSet
15: TestSet ← **REST**(vectoriser, class)
16: trained classifer← **TRAINCLASSIFER**(TrainSet)
17: P ← < > // P list of sentiment orientation predictions for test set reviews
18: P ← trained classifer .predict(TestSet)
19: accuracy ←**CALCULATEACCURICY**(P, TestSet)
20: return P,accuracy

//**Helper Functions**

21: **STOPWORDSREMOVEAL (T):**
22: for each t_i in T
23: **DELET**(t_i) from T if t_i ∈ S
24: end for
25: Return T

26: **STEMMINGUSINGPORTERSTEMMER (T):**
27: for each t_i in T
28: s← **STEM**(t_i) // porter stemmer
29: Replace t_i with s
30: end for
31: Return T

32: **STEMMINGUSINGSNOWPALLSTEMMER (T):**
33: for each t_i in T
34: s← **STEM**(t_i) // Snow Pall stemmer
35: Replace t_i with s
36: end for
37: Return T

38: **LEMMATIZATION(T):**
39:for each t_i in T
40: l← **LEMMA**(t_i) //using WordNet lemmatizer
41: Replace t_i with l
42: end for
43: Return T

44: **N-GRAMRECOGNITION(T):**
45: N_Grams_T←◊
46: for each t_i, t_{i+1} in T
47: S ←**SYNSETS**(t_i, t_{i+1}) // using WordNet we find synsets for bigram t_i , t_{i+1}
48: If S ≠ NIL then
49: Append(t_i _ t_{i+1}) to N_Grams_T
50: Else
51: Append(t_i) to N_Grams_T

52: end for
53: Return N_Grams_T

54: **NOUNSYNONYMSBASEDENRICHMENT(T):**
55:for each t_i in T
56: post ← **POST**(t_i)// we used nltk.pos_tag to determine which words are nouns.
57: If post = NN
58: ss← **SYNSETS**(t_i) //using WordNet
59: Append ss to T
60: end for
61: Return T

62: **ADJECTIVESYNONYMSBASEDENRICHMENT (T):**
63: for each t_i in T
64: post ← **POST**(t_i) // we used nltk.pos_tag to determine which words are adjectives.
65: If post = Adj
66: ss← **SYNSETS**(t_i) //using WordNet
67: Append ss to T
68: end for
69: Return T

Algorithm 2. Identifying The Sentiment Orientation using SentiwordNet lexicon.

INPUT: CSV File contains list of Reviews (sentences) with their sentiment polarity (positive or negative),R <reviews, polarity>. The SentiWordNet Lexicon, S < SentiWordNet Lexicon >. And list of unwanted signs(%,#,%,|), L < unwanted signs>
OUTPUT: Sentiment Orientation for dataset reviews and algorithm accuracy.

INITIALIZATION: SumPos and SumNeg equal zero. Where
SumPos: Accumulate the positive score of sentence tokens.
SumNeg: Accumulate the negative score of sentence tokens.

PREDICTSENTIMENT(R,S,L) : sentiment orientation, algorithm accuracy
1: reviews ← R <r_i, p_i>.
2:while reviews ≠ NIL do:
3: for each word in r_i do
4: do **DELETE**(word) if word ∈ (L or numbers) .
5: word ← **CASEFOLDING**(word)
6: end for
7: Polarity ← ◊
8: while Next r_i in reviews do
9: T ← < >. // T: list of tokens for r_i
10: T ← **TOKENIZE**(r_i)
11: for each $t_i \in T$ do
12: PosScore , NegScore← **SEARCHANDGETSENTIWORDNETSCORES** (t_i, S)
13: SumPos ← SumPos + **PosScore**(t_i)
14: SumNeg ← SumNeg + **NegScore**(t_i)
15: end for
16: $Polarity_{r_i}$ ← "Positive" if SumPos >=SumNeg
17: $Polarity_{r_i}$ ← "Negative" if SumPos<SumNeg
18: accuracy ←**CALCULATEACCURACY**($Polarity$, reviews)
19: return Polarity,accuracy

We use Algorithms 2 for identifying the sentiment orientation using SentiWordNet lexicon. In particular, we use this lexicon to find SentiWordNet scores for every token in each review (step 12). Steps 13 and 14 show the accumulate positive score and

negative score of review tokens. We used these accumulate scores to determine sentence sentiment orientations and produce predictions for the used datasets associated with their accuracy metrics. Our goal of using SentiWordNet in this context is to find out how a sentiment classifier's accuracy can be affected by the incorporation of prior term polarity information.

3.4.2 Second Proposed Sentiment Analysis Pipeline

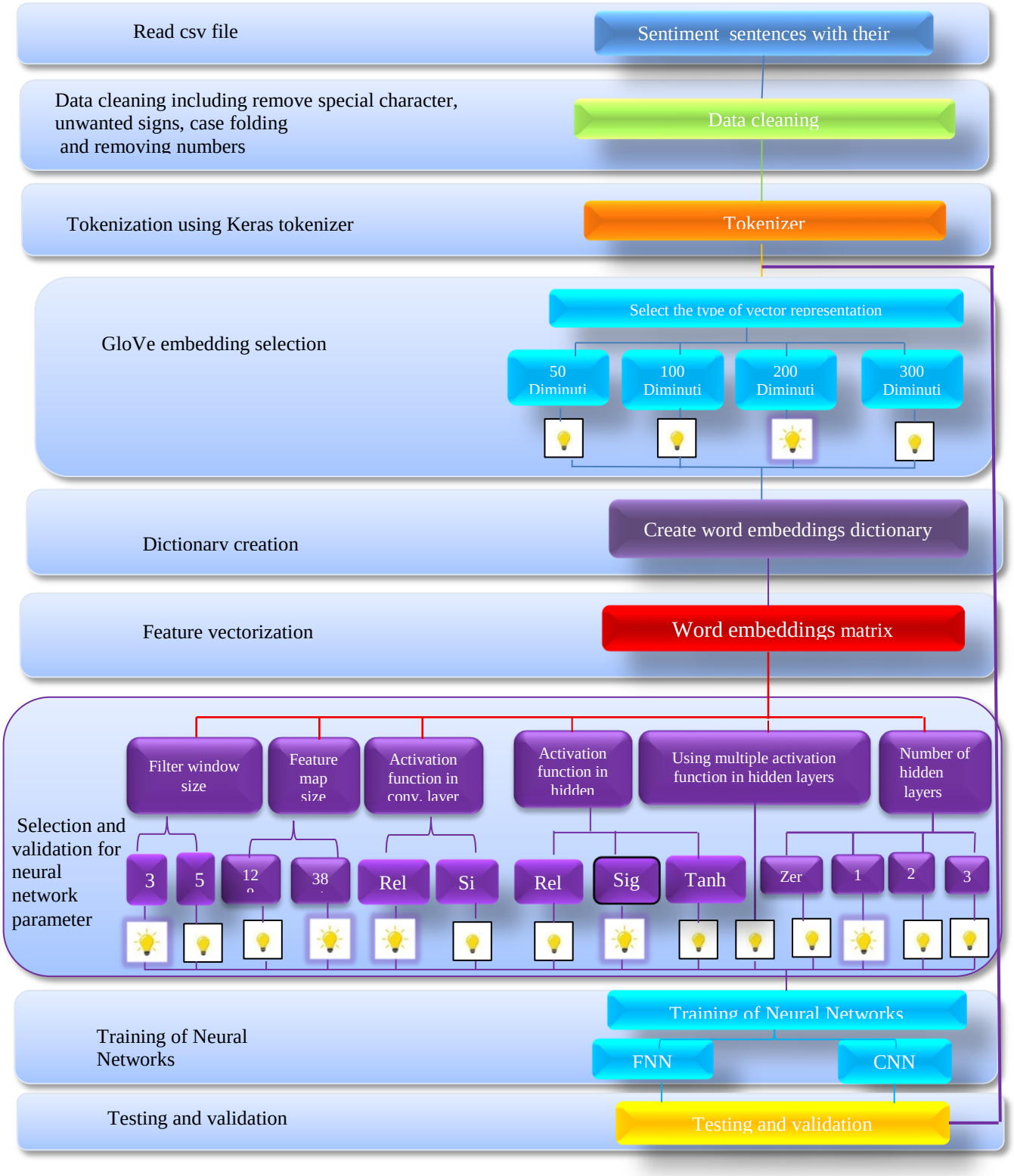


Figure 13:Phases of the second proposed sentiment analysis pipeline.

As shown in Fig. 13. First, we read the dataset containing sentences associated with their sentiment classes. Then, we perform data cleaning by removing special characters, unwanted signs, case folding and removing numbers. After this step, we apply text tokenization using Tokenizer provided by Keras model. Then, we select the type of word vector representation, which will be used to represent features. Next, we load the word embedding's according to the selected type for vector representation. At this step, we create a dictionary for words with their vectors. In the fourth step, we use the produced tokens in the previous step to create the embedding matrix that contains vectors for corpus word. Then, in the fifth step, we select and activate some neural network parameters including Filter window size, feature map size, activation function in convolutional layer and number of hidden layers. If activation functions are to be used in hidden layers, then they also should be selected. In next steps, an embedding matrix should be used depending on selected parameters for neural networks. We have trained the FNN and CNN neural networks. Then we have tested and validated the proposed model. The details of these steps are presented in Algorithm 3. First, we perform data cleaning (steps 2 to 6). Step 10 presents the tokenization function. In step 11, we select specific feature vector representation. Then, in step 12, we load Glove word embedding's based on the selected word representation in the previous step. In step 14, we create embedding matrix for corpus features. Next, at step 18, we train the classifier carry out the testing and validation task and calculates accuracy of the results in steps 20 and 21. The output of Algorithm 3 is a list of sentiment orientation predictions associated with accuracy metrics for each input dataset.

Algorithm 3. Identifying The Sentiment Orientation Algorithm

INPUT: file containing the list of sentiment sentences with their sentiment polarity. List of glove embedding's, G<LIST OF EMBEDDINGS> And list of unwanted signs(%,#,%,|), L < unwanted signs >
OUTPUT: list of sentiment orientation predictions for test set reviews and algorithm accuracy.

PREDICTSENTIMENT(R,S,L) : sentiment orientation, algorithm accuracy
1: reviews ← R <r_i, p_i> // list of sentiment sentences with their sentiment polarity
2: while reviews ≠ NIL do:
3: for each word in r_i do
4: do **DELETE**(word) if word ∈ (L or numbers) .
5: word ← **CASEFOLDING**(word)
6: end for
7: S ← < > // F: list of vector representations
8: while Next r_i in reviews do
9: T ← < >// T: list of tokens for r_i
10: t_i ← **TOKENIZE**(r_i) // tokenization using keras _Tokenize
11: S ← **SELECTION**TYPE OFVECTORREPRESENTATION( )
12: G ← glove embeddings (S)
13: class ← < p_i>
14: E← embedding matrix(T,G)
15: TrainSet ← 0,70 × (E, class) //Split vectors in to TrainSet and TestSet
16: TestSet ← **REST**(E, class)
17: NP ← **SELECTIONANDACTIVATIONFOR NEURALNETWORKPARAMETER**( )
18: trained classifer← **TRAINNEURALNETWORK**(TrainSet,NP)
19: P ← < > // P list of sentiment orientation predictions for test set reviews
20: P ← trained neural network .predict(TestSet,NP)
21: accuracy ←**CALCULATEACCURICY**(P, TestSet)
22: return P,accuracy

3.4.3 Third Proposed Sentiment Analysis Pipeline

As depicted by Fig. 14, the third proposed pipeline combines the two previous pipelines by extracting features using the NLP techniques explained in the first section and presenting the selected features using GloVe pre-trained embeddings. Testing and evaluation were carried out by passing the embedding matrix to both the utilized FNN and CNN networks. Using the results of the second part of the experiments, we selected parameters for the CNN and FNN, using the combination that produced the most accurate results. More details on the experiments that have carries out to evaluate this pipeline as provided in the next chapter.

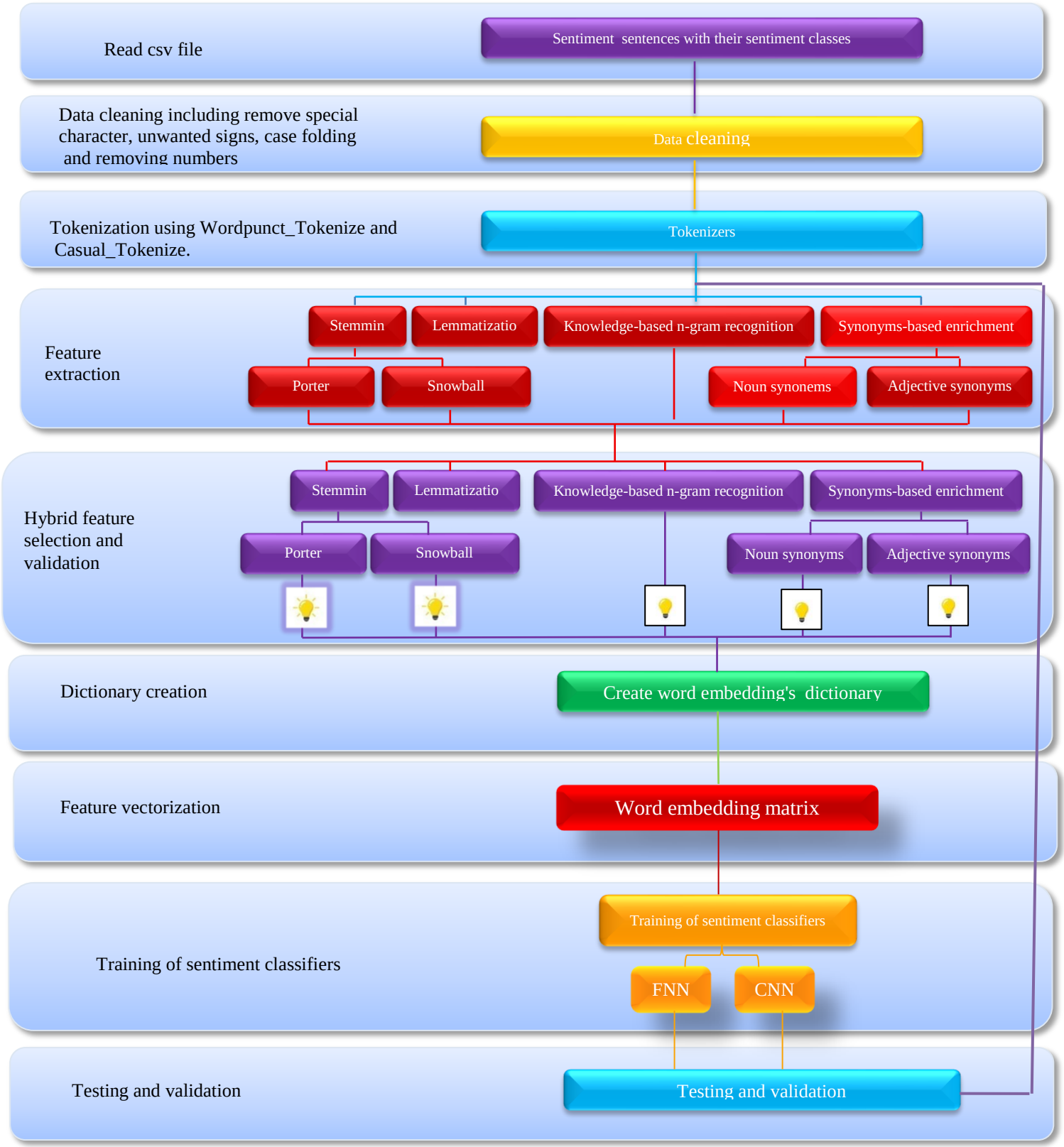


Figure 14: Phases of the third proposed sentiment analysis pipeline.

3.5 Summary

In this chapter, we present our three proposed sentiment analysis pipelines. The first model is based on machine learning techniques. The two other models are based on neural networks. Our proposed system has three main phases (Data Acquisition and Cleaning, Tokenization and Feature Extraction, and Features Selection and Sentiment Classification Techniques). In addition, we explained the details of proposed pipeline using diagrams and pseudo code algorithms describing the various parameters that are incorporated as part of the algorithmic flow of the pipeline sequences.

CHAPTER 4

Experimental Setup and Evaluation

For the purpose of improving sentiment classification accuracy, our experiments were split into three parts. In each part, we have used three different sentiment analysis pipelines. In the first part, we experimentally evaluate five sentiment classifiers, namely Support Vector Machines (SVM), Naive Bayes (NB), Logistic Regression, Random Forests and Artificial Neural Network (FNN). Moreover, we have used TF-IDF approach for feature vectorization and studied the impact of a variety of NLP pipelines on the quality of the predicted sentiment orientations. Additionally, we measured the impact of incorporating lexical semantic knowledge captured by WordNet on expanding original words in sentences. In the second part, we focused on studying the impact of utilizing GloVe pre-trained embedding's and Word2Vec embeddings on sentiment classification accuracy. For testing and evaluation, we used Feedforward neural network and convolutional neural network. In addition, we studied a set of FNN and CNN parameters to determine their effect on sentiment analysis accuracy. Finally, in the third part, we combined the previous two sentiment analysis pipelines by extracting features using the NLP processes explained in first section and representing the selected features using glove pre-trained embeddings. For testing and evaluation, we passed the embedding matrix to FNN and CNN And the parameters of the CNN and the FNN were selected based on the second part of experiments. We used the combination of parameters that produced the most accurate results.

4.1 Experiments – First Part

To perform this part of experiments, we used Python programming language for implementing the proposed NLP pipeline and utilize it for processing the three publicly-available datasets described in Table 3**.** These are: LightSide's movie reviews dataset, sentiment sentence dataset provided by [26], and the IMDB dataset. Each dataset is broken down into a training set and a testing set. The training set makes up 70% of the total dataset While the test set makes up the remaining30%. Below, we present some preliminary statistics about the used datasets.

Table 3:Statistics about the Used Sentiment Review Datasets.

Dataset	Sentiment Sentences	Positive	Negative	Average Unigrams Per Review	Average Unigrams & Bi-grams Per Review	Average Unigrams Without Stopwords Per Review
IMDB	50,000	25,000	25,000	270.82	261.4352	148.5876
Sentiment Sentences from Reference **[26]**	10,662	5,331	5,3331	22.71264	21.09316	13.75654
LightSide's Movie Reviews	3000	150	150	698.5533	674.4867	396.12

In our experiments, we classified the reviews using five machine learning algorithms, namely Support Vector Machine(Kernel=linear, Regularization parameter C= 1 default value), Naïve Bays, Logistic Regression, Random Forest, and Artificial neural network classifiers(FNN without hidden layer, input layer with 256 unit and ReLU activation function, Dropout layer with 0.2 dropout, output layer with sigmoid activation function). We have also used accuracy metric to compare between these classifiers. Accuracy in this context is the ratio of the number of correctly classified reviews to the total number of reviews. Equation (13) explains how to calculate accuracy, where a

sentiment prediction is classified into true positive (TP), true negative (TN), False Positive (FP) and False Negative (FN).

$$\text{accuracy}=\frac{\text{number of correctly classified reviews}}{\text{number of reviews}}=\frac{\text{tp+tn}}{\text{tp+tn+fp+fn}} \quad (13)$$

We also used the Natural Language Toolkit (NLTK) to apply some NLP techniques. Using the NLTK library, we employed two types of tokenizers, namely wordpunct and casual tokenizers. We have also used uni-grams features and n-grams features (uni-grams and bi-grams). Moreover, we have updated NLTK stopwords list to filter out sentences. It is important to point out that we removed all negation words from the predefined stopwords list. This is because such words have an important impact on identified correct sentiment orientations. We also enriched the list with additional words that appear frequently in the datasets (domain-specific stopwords). Example of such words are: story, movies, watching, acting, actors, director, character, characters, and film. Accordingly, we exploited a number of NLP pipelines to study the impact of each pipeline phases on the quality of each classifier. Tables 4-6 illustrate the variations on the accuracy results when utilizing each pipeline.

In our experiments, we have used 19 different combinations of NLP techniques. Each combination was applied twice, using Wordpunct and casual tokenizers. Each combination of NLP techniques produced different set of features. Which means different results for the sentiment reviews polarity. Table 4 presents the set of features when applying different combinations of NLP techniques.

Table 4:Set of Features when Applying Different Combination of NLP Techniques.

Composite NLP Techniques	Features
Original Sentence	it's nice to see piscopo again after all these years, and chaykin and headly are priceless .
Wordpunct tokenize	['it', "'", 's', 'nice', 'to', 'see', 'piscopo', 'again', 'after', 'all', 'these', 'years', ',', 'and', 'chaykin', 'and', 'headly', 'are', 'priceless', '.']
casual tokenize	["it's", 'nice', 'to', 'see', 'piscopo', 'again', 'after', 'all', 'these', 'years', ',', 'and', 'chaykin', 'and', 'headly', 'are', 'priceless', '.']
Remove Stopwords	['nice', 'piscopo', ',', 'chaykin', 'headly', 'priceless', '.']
Porter stemmer	['it', "'", 's', 'nice', 'to', 'see', 'piscopo', 'again', 'after', 'all', 'these', 'year', ',', 'and', 'chaykin', 'and', 'headli', 'are', 'priceless', '.']
Porter stemmer+ Remove Stopwords	['nice', 'piscopo', 'year', ',', 'chaykin', 'headli', 'priceless', '.']
Lemmatization	['it', "'", 's', 'nice', 'to', 'see', 'piscopo', 'again', 'after', 'all', 'these', 'year', ',', 'and', 'chaykin', 'and', 'headly', 'be', 'priceless', '.']
Knowledge-based n-gram recognition	['it', "'", 's', 'nice', 'to', 'see', 'piscopo', 'again', 'after_all', 'these', 'years', ',', 'and', 'chaykin', 'and', 'headly', 'are', 'priceless']
Synonyms-based enrichment (noun synonems)	['it', "'", 's', 'nice', 'to', 'see', 'piscopo', 'again', 'after', 'all', 'these', 'years', ',', 'and', 'chaykin', 'and', 'headly', 'are', 'priceless', '.', 'old_age']
Synonyms-based enrichment (adjective synonems)	['it', "'", 's', 'nice', 'to', 'see', 'piscopo', 'again', 'after', 'all', 'these', 'years', ',', 'and', 'chaykin', 'and', 'headly', 'are', 'priceless', '.', 'second', 'Nice', 'invaluable']

Table 5:Experimental Results Using Wordpunct_Tokenize

	Datasets	SVM	NB	LG	RF	FNN
Original Text	IMDB	90.27%	85.96%	89.84%	84.42%	87.82%
	Sentiment_sentences	76.52%	77.61%	75.52%	68.23%	76.45%
	Movie Review	70.00%	52.22%	67.78%	75.56	71.11%
Remove unwanted symbols and numbers	IMDB	90.20%	85.87%	89.71%	84.38%	87.733%
	Sentiment_sentences	76.64%	77.55%	75.45%	68.95%	76.26%
	Movie Review	71.11%	52.22%	67.78%	68.89%	72.22%
Stopwords Removal	IMDB	89.81%	86.61%	89.77%	86.22%	87.08%
	Sentiment_sentences	74.64%	76.27%	74.68%	68.11%	76.36%
	Movie Review	64.44%	56.67%	61.11%	68.89%	72.22%
Porter stemmer	IMDB	89.95%	85.25%	89.61%	84.08%	87.87%
	Sentiment_sentences	77.49%	77.11%	76.17%	69.92%	76.01%
	Movie Review	70.00%	52.22%	63.33%	72.22%	58.88%
Porter stemmer +Stopwords Removal	IMDB	89.51%	85.66%	89.37%	85.37%	87.586%
	Sentiment_sentences	75.23%	76.11%	74.36%	69.45%	76.14%
	Movie Review	64.44%	56.67%	64.44%	65.56%	66.66%
Snowball stemmer	IMDB	89.90%	85.24%	89.60%	84.61%	87.77%
	Sentiment_sentences	**78.08%**	77.64%	76.49%	70.33%	76.04%
	Movie Review	71.11%	52.22%	65.56%	75.56%	64.44%
Snowball stemmer +Stopwords Removal	IMDB	89.44%	85.57%	89.18%	85.12%	87.14%
	Sentiment_sentences	76.14%	76.64%	75.20%	69.54%	75.20%
	Movie Review	63.33%	57.78%	63.33%	58.89%	57.77%
Lemmatizer	IMDB	90.07%	85.63%	89.74%	84.40%	87.91%

	Sentiment_sentences	76.83%	77.08%	75.52%	69.14%	76.14%
	Movie Review	71.11%	51.11%	66.67%	72.22%	76.66%
Lemmatizer +Stopwords Removal	IMDB	89.63%	86.37%	89.51%	85.85%	87.39%
	Sentiment_sentences	74.89%	76.17%	74.14%	68.51%	75.29%
	Movie Review	66.67%	55.56%	64.44%	56.67%	57.77%
Uni-gram &Bi-gram	IMDB	90.39%	86.01%	89.97%	84.88%	87.88%
	Sentiment_sentences	77.11%	77.55%	75.08%	68.36%	76.14%
	Movie Review	71.11%	52.22%	65.56%	67.78%	64.44%
Uni-gram &Bi-gram Stopwords Removal	IMDB	90.17%	86.80%	90.02%	86.27%	87.08%
	Sentiment_sentences	74.86%	76.95%	74.73%	68.76%	75.42%
	Movie Review	64.44%	56.67%	61.11%	73.33%	67.77%
Uni-gram &Bi-gram +Porter stemmer	IMDB	90.15%	85.51%	89.79%	84.30%	87.88%
	Sentiment_sentences	77.20%	77.24%	76.05%	69.95%	75.92%
	Movie Review	68.89%	52.222%	66.67%	71.11%	62.22%
Uni-gram &Bi-gram + Snowball stemmer	IMDB	90.09%	85.35%	89.73%	84.68%	87.86%
	Sentiment_sentences	77.49%	77.77%	76.49%	69.45%	76.61%
	Movie Review	68.89%	52.22%	66.67%	77.78%	75.55%
Uni-gram &Bi-gram + Lemmatizer	IMDB	**90.43%**	85.99%	89.95%	84.75%	87.7%
	Sentiment_sentences	76.67%	77.39%	74.92%	69.04%	76.36%
	Movie Review	70.00%	51.11%	64.44%	72.22%	73.33%
Uni-gram + adj synonyms	IMDB	90.19%	86.59%	89.98%	84.45%	87.72%
	Sentiment_sentences	76.45%	77.17%	76.08%	69.23%	76.29%
	Movie Review	72.22%	54.44%	70.00%	73.33%	67.77%
Uni-gram +Noun synonyms	IMDB	90.07%	84.75%	89.24%	84.01%	87.00%
	Sentiment_sentences	74.80%	75.30%	73.89%	69.48%	75.51%
	Movie Review	64.44%	52.22%	62.22%	74.44%	58.88%
Uni-gram ,Bi-grams +Noun synonyms + lemmatization	IMDB	89.99%	84.99%	89.33%	84.28%	87.27%
	Sentiment_sentences	74.67%	75.39%	73.42%	68.76%	75.67%
	Movie Review	65.56%	51.11%	65.56%	71.11%	71.11%
Uni-gram ,Bi-grams +Adj synonyms + lemmatization	IMDB	90.30%	86.83%	90.11%	84.95%	88.01%
	Sentiment_sentences	76.08%	77.55%	75.30%	68.54%	76.17%
	Movie Review	74.44%	51.11%	68.89%	65.56%	62.22%
Uni-gram ,Bi-grams +Adj synonyms	IMDB	90.33%	86.77%	90.01%	85.15%	87.91%
	Sentiment_sentences	76.42%	77.05%	75.92%	68.89%	75.79%
	Movie Review	73.33%	52.22%	67.78%	**82.22%**	74.44**%**

Table 6: Experimental Results Using Casual_Tokenize

	Datasets	**SVM**	**NB**	**LG**	**RF**	**FNN**
Original Text	IMDB	**90.50%**	86.14%	89.85%	84.54%	87.46%
	Sentiment_sentences	76.77%	77.08%	75.20%	67.89%	76.17%
	Movie Review	68.89%	52.22%	66.67%	74.44%	74.44%
Remove unwanted numbers and symbols	IMDB	90.16%	85.94%	89.63%	84.33%	87.59%
	Sentiment_sentences	76.70%	77.11%	75.33%	68.42%	76.26%
	Movie Review	68.89%	52.22%	65.56%	71.11%	67.77%
Stopwords Removal	IMDB	89.75%	86.56%	89.57%	85.70%	87.08%
	Sentiment_sentences	74.45%	75.89%	74.05%	69.25%	73.67%
	Movie Review	62.22%	58.89%	60.00%	63.33%	60.00%
Porter stemmer	IMDB	89.72%	85.34%	89.56%	83.78%	87.57%
	Sentiment_sentences	76.77%	76.58%	76.39%	69.26%	75.26%
	Movie Review	68.89%	51.11%	64.44%	72.22%	72.22%
Porter stemmer +Stopwords Removal	IMDB	89.51%	85.67%	89.38%	85.01%	87.12%
	Sentiment_sentences	75.17%	76.11%	74.70%	68.98%	74.57%

	Movie Review	65.56%	57.78%	63.33%	67.78%	65.55%
Snowball stemmer	IMDB	89.85%	85.34%	89.65%	84.43%	87.73%
	Sentiment_sentences	**77.64%**	77.58%	76.52%	69.42%	75.79%
	Movie Review	70.00%	52.22%	65.56%	64.44%	67.77%
Snowball stemmer +Stopwords Removal	IMDB	89.85%	85.34%	89.65%	84.43%	87.14%
	Sentiment_sentences	76.36%	76.55%	75.30%	69.95%	75.14%
	Movie Review	62.22%	57.78%	63.33%	72.22%	74.44%
Lemmatizer	IMDB	90.09%	85.67%	89.56%	84.57%	87.70%
	Sentiment_sentences	76.14%	76.87%	74.98%	68.92%	75.64%
	Movie Review	70.00%	51.11%	65.56%	65.56%	72.22%
Lemmatizer &Stopwords Removal	IMDB	89.65%	86.27%	89.27%	85.79%	87.05%
	Sentiment_sentences	74.89%	75.86%	74.33%	69.32%	75.04%
	Movie Review	62.22%	55.56%	62.22%	66.67%	64.44%
Uni-gram &Bi-gram	IMDB	90.35%	86.21%	89.95%	85.19%	87.76%
	Sentiment_sentences	76.64%	77.11%	74.83%	67.92%	76.17%
	Movie Review	67.78%	51.11%	66.67%	**75.56%**	75.55**%**
Uni-gram &Bi-gram Stopwords Removal	IMDB	89.92%	86.68%	90.06%	86.13%	87.12%
	Sentiment_sentences	74.52%	76.17%	74.23%	69.76%	74.54%
	Movie Review	62.22%	56.67%	58.89%	67.78%	74.44%
Uni-gram &Bi-gram+ Porter stemmer	IMDB	90.17%	85.70%	89.77%	84.29%	**87.87**%
	Sentiment_sentences	76.49%	76.58%	76.02%	69.45%	76.26%
	Movie Review	65.56%	52.22%	64.44%	**75.56%**	73.33**%**
Uni-gram &Bi-gram+ Snowball stemmer	IMDB	90.03%	85.54%	89.74%	84.19%	87.77%
	Sentiment_sentences	77.20%	77.30%	76.11%	69.64%	76.17%
	Movie Review	65.56%	52.22%	64.44%	71.11%	75.55%
Uni-gram &Bi-gram+Lemmatizer	IMDB	90.30%	86.02%	89.91%	84.866%	**87.90**%
	Sentiment_sentences	76.17%	76.77%	75.05%	68.48%	76.39%
	Movie Review	66.67%	51.11%	63.33%	73.33%	73.33%
Uni-gram+ ADJ synonyms	IMDB	90.15%	86.60%	89.79%	84.92%	**87.81**%
	Sentiment_sentences	76.42%	77.36%	75.02%	68.92%	75.73%
	Movie Review	70.00%	54.44%	68.89%	71.11%	67.77%
Uni-gram+Noun synonyms	IMDB	90.15%	86.60%	89.79%	84.92%	87.19%
	Sentiment_sentences	75.27%	74.80%	74.17%	68.92%	74.92%
	Movie Review	64.44%	54.44%	63.33%	**75.56%**	68.88**%**
Uni-gram Bi-grams+ADJ synonyms	IMDB	90.34%	86.91%	89.86%	85.36%	**87.91**%
	Sentiment_sentences	76.45%	77.14%	75.23%	68.29%	75.89%
	Movie Review	68.89%	51.11%	70.00%	**75.56%**	**77.77%**
Uni-gram Bi-grams+Noun synonyms	IMDB	90.11%	85.38%	89.57%	84.83%	87.62%
	Sentiment_sentences	75.14%	75.14%	73.61%	68.64%	75.26%
	Movie Review	68.89%	53.33%	62.22%	66.67%	69.99%

Based on the results in Tables 5 and 6, we notice that all classifiers are positively affected when we used n-grams, snowball stemmer, sentence enrichment with adjective synonyms, lemmatization, and when we removed unwanted signs and numbers and also when we have a combination of them. This means that these NLP techniques have an important and positive impact in determining the polarity of reviews. In addition, we note that synonyms adjective-based enrichment led to better results than when we perform noun-based enrichment. Using the Wordpunct_tokenizer with the IMDB

dataset, we obtained the best result (90.43%). In particular, when we used SVM classifier with uni-grams and bi-grams with lemmatization. Both the NB, ANN and LR classifiers produced (86.83%), (88.01%) and (90.11%) accuracy results when we used n-grams feature with sentence enrichment by Adj synonyms and lemmatization. The RF classifier produced best the result (86.20%) when we used n-grams with stopwords removal.

Considering the Sentiment Sentences dataset that is used in[26], we obtained the best result (78.08%) when using the SVM classifier with Snowball stemmer. Besides, the RF classifier produced the best predictions (70.33%) when we used Snowball stemmer. As far as the NB, ANN and LR classifiers are concerned, they produced the best results (77.77%), (76.61%) and (76.49%), respectively when using n-grams with Snowball stemmer.

For the LightSide's Movie_Reviews dataset, we obtained the best result (82.22%) when we used the RF classifier with n_grams and Adjective synonyms combination. Also the SVM classifier produced the best result (74.44%) when we used n_grams, sentence enrichment with Adjective synonyms and lemmatization. The NB classifier produced the best result (57.78%) when using Snowball stemmer with StopWords removal. The ANN classifier produced the best result (76.66%) when we have used wordNet lemmatizer. While the LR produced best result (70.00%) when we used sentence enrichment with Adjective synonyms.

When using the Casual_tokenizer, we find that we got the best result (90.50%) for the IMDB dataset using the SVM classifier applied on the original text. On the other hand, the NB classifier produced the best result (86.91%) when using n-gram features with

sentence enrichment by Adj synonyms and lemmatization. Both the LR and RF classifiers produced the best results (90.06%) and (86.13%) when we used n-grams with stopwords removal. While the ANN classifier produced the best result (87.91%) when using n-gram features with sentence enrichment by Adj synonyms.

In Sentiment_sentences dataset We got the best result (77.64%) when with the SVM classifier when employing the Snowball stemmer. Also both the NB and LR classifiers produced the best results (77.58%) and (76.52%), respectively when we used Snowball stemmer. For the RF classifier, it produced the best result (69.95%) when we used the Snowball stemmer with Stopwords removal. While the ANN classifier produced the best result (76.39%) when we used n_grams, sentence enrichment with lemmatization.

For the LightSide's Movie_Reviews dataset, we obtained the best result (75.56%) when using the RF classifier with n_grams and sentence enrichment with Adjective synonyms. Also, the LR classifier produced the best result (70.00%) when we used n_grams with sentence enrichment with Adjective synonyms. On the other hand, the NB classifier produced the best result (58.89%) when we used Stopwords removal. While the SVM classifier produced the best result (70.00%) when we used sentence enrichment with Adjective synonyms, lemmatization or snowball stemmer. The ANN classifier produced the best result (77.77%) when using n-gram features with sentence enrichment by Adj synonyms.

Table 7: Experiments Results Using Lexicon-based Sentiment Analysis

Technique / Dataset	SentiWordNet	Wordpunct Tokenize	Casual Tokenize
IMDB	65.00%	90.43%	90.50%
Sentiment Sentences	58.00%	78.08%	77.64%
Movie Review	59.00%	82.22%	75.56%

As illustrated from the results in Table 7, we can see that the obtained results using the lexicon-based sentiment analysis technique are less precise than their counterparts. This is mainly due to the fact that this approach relies on the accumulative sum of the pre-defined polarity of review tokens. Compared to the results of the previous approaches, we note that this approach ignores a word's contextual polarity in a given review. Additionally, the latent semantic dimensions of tokens are ignored in this model.

4.2 Experiments – Second Part

To perform this part of experiments, also we used Python program language for implementing the proposed Neural Networks and utilize them for processing the four publicly-available datasets described in Table 8**.** These are: LightSide's movie reviews dataset, sentiment sentences dataset [26], sentiment 140,and the IMDB dataset. Below, we present some basic information about the used datasets. In this part of experiments, we have used both GloVe and Word2veC embedding for feature representation.

Table 8:Statistics about the Used Sentiment Review Datasets

Dataset	Sentiment Sentences	Positive	Negative
IMDB	50,000	25,000	25,000
Sentiment Sentences from Reference[26]	10,662	5,331	5,3331
LightSide's Movie Reviews	3000	150	150
Sentiment 140	1,600,000	800,000	800,000

After cleaning the data by removing HTML tags, punctuation, and numbers, along with all unnecessary characters and white spaces, it becomes ready for analysis. Then we have created a word-to-index dictionary using the tokenizer class in keras module where each word in the corpus is a key and the corresponding unique index is the value. We then load the GloVe word embeddings and create a dictionary that contain words as keys and their corresponding embedding lists as values. In the next step, we created the embedding matrix where each row number corresponds to an index of a word in the

corpus. In addition, the matrix columns will contain GloVe word embeddings for words in our corpus. Glove word embeddings supports four different vector representations, which are represented by four dimension classes: 50, 100, 200 and 300. By using GloVe pertained model in our experiments, we have generated vectors using four different representations. In our experiments, we classified the reviews using two types of neural networks, namely FNN, and CNN And then we performed the training and testing phases using neural networks. Each dataset is broken down into a training set and a testing set. The training set makes up 70% of the total dataset While the test set makes up the remaining30%. In our experiments, we have used FFNs with a zero hidden layer, one hidden layer, two hidden layers, and three hidden layers. For CNNs we have used it with different feature map size, different filter window size, and different activation functions in convolution layer. After the convolution layer we have the max pooling layer, followed by FFNs with a zero hidden layer, one hidden layer, two hidden layers, or three hidden layers. Finally, we have used accuracy metric to compare between these different cases. In the next phase, we used word2vec embeddings for feature representation. Tables 9 and 10 illustrate FNN and CNN networks parameters respectively. We studied the effect of each of the parameters listed below on FNN and CNN networks classification accuracy.

- Using multidimensional vector representations of a word.
- Using different numbers of hidden layers.
- Utilizing different activation functions in hidden layers.
- Utilizing multiple activation functions in hidden layers.
- Using a different activation function in the convolution layer.

- Changing feature map size used in the convolution layer.

- Changing filter window size used in the convolution layer.

Table 9:FFN parameters

FFN parameters	
First hidden layer	300 unit
Dropout	0.3
Second hidden layer	50 unit
Dropout	0.2
Third hidden layer	10 unit
Dropout	0.2
Optimizer	Adam
Loss function	Binary-cross entropy
Activation function for output layer	Sigmoid
Batch size	128
Epochs	6
Activation function	Sigmoid, ReLU, Tanh
Weight matrix	Embedding matrix
Bias	Default (zero)

Table 10: Convolutional layer Parameters

Convolutional layer Parameters	
Feature Map	128, 384
Filters Window Size	3,5
Activation function	Sigmoid, ReLU
Learning rate	Default (0.001)

In Tables 11-15 illustrate the variations on the accuracy results when utilizing GloVe embedding's and Word2Vec embeddings.

Table 11: Experimental Results Using FNN and CNN without hidden layers(Convolutional Layer with 128 unit and ReLU Activation Function, Filters Window Size=5)

No hidden layers		GloVe Embedding's		Word2Vec Embedding's	
input vector dimensions	Dataset	FNN	CNN	FNN	CNN
300	IMDB	76.33%	89.48%	76.95%	88.94%
	sentiment_sentences	70.60%	76.73%	53.09%	54.15%
	MovieReviews	51.11%	52.22%	44.44%	53.33%
	Sentiment 140	72.75%	79.63%	73.97%	80.49%
200	IMDB	76.04%	89.10%	76.94%	88.55%
	sentiment_sentences	70.07%	75.45%	52.12%	55.00%
	MovieReviews	53.33%	51.11%	52.22%	46.66%

	Sentiment 140	72.03%	79.31%	73.97%	81.19%
100	IMDB	71.60%	88.07%	76.12%	88.01%
	sentiment_sentences	68.26%	74.92%	54.19%	54.12%
	MovieReviews	53.33%	51.11%	53.33%	53.33%
	Sentiment 140	70.49%	79.27%	72.43%	80.19%
50	IMDB	69.20%	85.94%	75.25%	86.50%
	sentiment_sentences	66.51%	71.85%	53.72%	56.12%
	MovieReviews	57.77%	58.88%	44.44%	53.33%
	Sentiment 140	67.69%	77.21%	70.14%	79.85%

Table 12:Experimental Results Using CNN without hidden layer with diffrent feature map, diffrentfilter window size and diffrent activation functions

Input Vector Dimentions=300	Dataset	Feature Map =128, Activation Functions =ReLU, Filter Window Size =5	Feature Map=384, Activation Functions= Sigmoid, Filter Window Size=5	Feature Map=384, Activation Functions= ReLU, Filter Window Size=5	Feature Map=384, Activation Functions= ReLU, Filter Window Size=3
GloVe	IMDB	89.48%	87.53%	90.20%	90.56%
	sentiment_sentences	76.73%	76.51%	76.76%	77.64%
	MovieReviews	52.22%	61.11%	63.33%	63.33%
	Sentiment 140	79.63%	79.24%	79.09%	79.98%
Word2Vec	IMDB	88.94%	89.43%	89.62%	89.48%
	sentiment_sentences	54.15%	52.56%	54.37%	54.81%
	MovieReviews	53.33%	46.66%	55.55%	53.33%
	Sentiment 140	81.06%	80.58%	81.14%	81.09%

Based on the results in Table 11, we notice that accuracy increases when using high dimension vectors to represent the words. The highest classification accuracy was achieved when representing words using Glove word embeddings with 300 dimensions. Therefore, this number of dimensions was used to perform the rest of the experiments that used Glove word embeddings. As shown in Table 12, using the ReLU activation function in the convolution layer gives better results than using the sigmoid activation function. In addition, the accuracy of classification is improved when bigger feature map size is used in then convolution layer. We got better results when using features map with a size of 384 in the convolutional layer compared to the results when using

features map with a size of 128. Therefore, in the rest of the cases, we used features map with size 384 in the convolutional layer. Furthermore, we found that using a filter window with a smaller value improves classification accuracy, especially when using CNNs with zero hidden layers and one hidden layer. As shown in Table 12, we got better results when using filter window with a size of 3 in the convolutional layer compared to using filter window with size 5.

Table 13: Experimental Results Using FNN and CNN with one hidden layer

One hidden layer /Dimention=300		GloVe				Word2Vec			
activation function	Dataset	FNN	CNN (Feature Map=384, Activation Functions= Sigmoid, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=3)	FNN	CNN (Feature Map=384, Activation Functions= Sigmoid, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=3)
ReLU	IMDB	76.66%	89.50%	89.26%	88.48%	77.43%	88.50%	89.24%	88.96%
	sentiment_sentences	69.07%	77.11%	77.20%	77.11%	52.75%	50.09%	54.59%	54.19%
	MovieReviews	46.66%	46.66%	53.33%	53.33%	48.88%	54.44%	46.66%	53.33%
	Sentiment 140	75.68%	79.62%	79.72%	80.73%	77.67%	80.86%	81.61%	81.69%
Sig	IMDB	77.14%	89.25%	89.72%	89.96%	78.47%	89.30%	89.63%	89.85%
	sentiment_sentences	68.63%	75.95%	76.86%	75.10%	52.84%	54.62%	54.03%	54.28%
	MovieReviews	46.66%	53.33%	46.66%	46.66%	53.33%	53.33%	46.66%	46.66%
	Sentiment 140	75.97%	79.60%	79.75%	80.21%	77.28%	80.93%	81.71%	81.93%
Tanh	IMDB	76.53%	89.26%	89.11%	89.50%	76.06%	88.42%	89.05%	89.17%
	sentiment_sentences	68.48%	76.54%	76.39%	76.67%	52.43%	49.312%	53.28%	54.22%
	MovieReviews	48.88%	64.44%	46.66%	54.44%	51.11%	46.66%	46.66%	53.33%
	Sentiment 140	75.70%	78.96%	79.69%	80.37%	76.06%	80.79%	81.30%	81.74%

Table 14: Experimental Results Using FNN and CNN with two hidden layers

Two hidden layer /Dimension=300		GloVe				Word2Vec			
activation function in hidden layers	Dataset	FNN	CNN (Feature Map=384, Activation Functions= Sigmoid, Filter Window	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=3)	FNN	CNN (Feature Map=384, Activation Functions= Sigmoid, Filter Window	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=3)

			Size=5)	Size=5)			Size=5)	Size=5)	
ReLU	IMDB	76.04%	89.10%	86.92%	86.26%	77.25%	87.98%	89.35%	89.27%
	sentiment_sentences	68.60%	76.61%	75.10%	76.01%	51.03%	49.31%	49.31%	49.31%
	MovieReviews	47.77%	46.66%	46.66%	52.22%	53.33%	46.66%	53.33%	53.33%
	Sentiment 140	75.66%	79.56%	79.99%	80.21%	77.82%	80.60%	81.60%	82.02%
Sig	IMDB	76.36%	89.07%	89.08%	87.34%	77.39%	88.78%	88.58%	89.52%
	sentiment_sentences	68.07%	77.01%	76.89%	77.45%	52.15%	49.31%	49.78%	52.93%
	MovieReviews	51.11%	53.33%	53.33%	46.66%	51.11%	46.66%	53.33%	46.66%
	Sentiment 140	76.08%	79.60%	79.52%	80.36%	77.18%	80.74%	81.63%	81.62%
Tanh	IMDB	74.06%	89.36%	87.20%	89.26%	76.37%	86.89%	88.78%	89.31%
	sentiment_sentences	67.44%	76.64%	76.26%	76.70%	50.71%	52.22%	52.87%	53.84%
	MovieReviews	50.00%	53.33%	62.22%	53.33%	46.66%	53.33%	53.33%	46.66%
	Sentiment 140	75.07%	79.38%	80.10%	80.53%	75.83%	79.99%	81.20%	81.37%
Sig, ReLU	IMDB	76.32%	88.51%	88.71%	89.02%	77.77%	87.84%	89.07%	88.52%
	sentiment_sentences	68.29%	75.51%	75.39%	76.61%	51.68%	49.31%	49.46%	49.56%
	MovieReviews	53.33%	46.66%	53.33%	53.33%	53.33%	46.66%	53.33%	53.33%
	Sentiment 140	75.82%	78.89%	80.00%	80.72%	77.10%	80.82%	81.66%	81.88%
ReLU,sig	IMDB	75.40%	89.36%	89.00%	89.20%	77.51%	88.53%	89.15%	89.45%
	sentiment_sentences	69.85%	76.57%	76.17%	77.14%	50.28%	53.18%	51.43%	53.62%
	MovieReviews	53.33%	53.33%	60.00%	53.33%	53.33%	46.66%	53.33%	53.33%
	Sentiment 140	75.71%	79.59%	80.10%	80.46%	77.64%	80.68%	79.99%	79.36%

Table 15: Experimental Results Using FNN and CNN with three hidden layers

Three hidden layer with /Dimention=300		GloVe				Word2Vec			
Activation Function in hidden layers	Dataset	FNN	CNN (Feature Map=384, Activation Functions= Sigmoid, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=3)	FNN	CNN (Feature Map=384, Activation Functions= Sigmoid, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=5)	CNN (Feature Map=384, Activation Functions= ReLU, Filter Window Size=3)
ReLU	IMDB	75.28%	89.16%	88.56%	85.14%	77.03%	86.89%	87.78%	89.39%
	sentiment_sentences	67.54%	76.51%	75.54%	76.23%	49.31%	49.31%	49.31%	49.31%
	MovieReviews	53.33%	53.33%	50.00%	47.77%	53.33%	53.33%	53.33%	46.66]%
	Sentiment 140	75.87%	79.49%	79.54%	80.57%	77.79%	80.42%	81.41%	81.86%
Sig	IMDB	76.35%	89.11%	88.94%	88.43%	77.66%	87.50%	89.48%	89.28%
	sentiment_sentences	69.04%	76.11%	76.86%	74.92%	52.62%	49.31%	53.06%	49.31%
	MovieReviews	52.22%	46.66%	46.66%	46.66%	50.00%	53.33%	53.33%	53.33%
	Sentiment 140	76.04%	79.36%	79.74%	80.69%	77.06%	80.62%	81.44%	81.91%
Tanh	IMDB	75.93%	89.46%	88.96%	88.15%	75.92%	88.02%	88.67%	88.25%
	sentiment_sentences	67.54%	76.14%	74.67%	76.07%	51.00%	50.68%	49.46%	52.62%
	MovieReviews	48.88%	46.66%	53.33%	46.66%	50.00%	53.33%	46.66%	53.33%
	Sentiment 140	75.06%	79.38%	79.75%	80.20%	75.64%	80.36%	81.11%	81.23%

Based on the results in Tables 13, 14 and 15, we notice that the Feedforward neural network's classification accuracy is best when sigmoid activation functions are used in

hidden layers. Nevertheless, the classification accuracy of convolutional neural networks was better in most cases when Relu activation function was used in the hidden layers. According to Table 14, using multiple activation functions in hidden layers improves classification accuracy. We obtained the best results when using the Relu activation function in the first hidden layer followed by the Sigmoid activation function in the second hidden layer. Finally, according to the results, not using hidden layers with FNN and CNN networks or just using one hidden layer produced better results than when using two and three hidden layers.

As we can see in Tables 13-15, using the FNN with the IMDB dataset, the best result we obtained is (77.14%). In particular, when using one hidden layer with the sigmoid activation function and using Glove word embeddings with 300 dimensions. Considering the Sentiment Sentences dataset that is used by the authors of [26], we obtained the best result (70.60%) when using FNN without hidden layers and using Glove word embeddings with 300 dimensions. For the LightSide's Movie_Reviews dataset, the best result we obtained is (57.78 %) when using FNN without hidden layers and using Glove word embeddings with 50 dimensions. For Sentiment 140, we obtained the best result (76.08%) when using FNN when with two hidden layers with the sigmoid activation function. Moreover, when using CNN, the best result obtained when we using 384 feature filter with ReLU activation function and a filter window of size 3 in the convolutional layer. For the IMDB dataset, the best result we obtained is (90.56%) when using CNN without any hidden layers and using Glove word embeddings with 300 dimensions. For the Sentiment Sentences dataset, the best result we obtained is (77.64%) when using CNN without hidden layers and using Glove word embeddings with 300 dimensions. For the LightSide's Movie_Reviews dataset, the best result we

obtained is (64.44%) when using CNN flowed by MLP with one hidden layer and tanh activation function. For Sentiment 140, the best result we obtained is (80.73 %) when using a convolutional layer flowed by MLP with one hidden layer and the ReLU activation function.

Based on Tables 11-15, it can be noticed that when using the GloVe embeddings, we got better results than when using Word2Vec embedding's especially when using small size datasets. When using FNN with the IMDB dataset, the best result we obtained is (78.47%) and it is (89.85%) when using CNN with one hidden layer and the sigmoid activation function and using Word2Vec word embeddings with 300 dimensions. While when we have used FNN with the sentiment140, the best result we obtained is (77.82 %) and it is (82.02%) when using CNN with two hidden layers and the ReLU activation function and using Word2Vec word embeddings with 300 dimensions. For both the Sentiment Sentences and LightSide's Movie_Reviews datasets, we notice that they did not provide reliable results.

4.3 Experiments – Third Part

To perform the experiments in this part, we used three datasets, namely LightSide's Movie_Reviewsdataset, sentiment sentence dataset provided by [26], and the IMDB dataset. In this section, we combined the previous two sentiment analysis pipelines where we used different NLP pipelines to extract the features used in first section. Then we have used glove pre-trained embeddings to represent selected features because it produced better results than wod2vec in the previous section. For testing and evaluation, we passed the embedding matrix to FNN and CNN and the parameters for CNN and FNN were selected based on the second part of the experiments. In fact, we used the

combination of the parameters that produced the most accurate results. We used glove embeddings with 300 dimensions to represent features and FNN and CNN without hidden layers and then with one hidden layer using the RelU activation function. In the convolutional layer we used RelU activation function, feature map with size 384 and feature window with size 3.

Tables 15 and 16 illustrate the variations on the accuracy results when utilizing each parameter.

Table 16:Experimental Results Using Wordpunct_Tokenize and Using FNN and CNN without hidden layers

No hidden layers		
Porter stemmer		
	FNN	CNN
IMDB	71.89%	88.10%
Sentiment sentences	64.10%	72.35%
Movie_Review	53.33%	56.66%
Snowball stemmer		
IMDB	72.35%	89.03%
Sentiment sentences	64.50%	73.51%
Movie_Review	53.33%	62.22%
Lemmatization		
IMDB	76.20%	90.42%
Sentiment sentences	69.38%	78.01%
Movie_Review	53.33%	68.88%
n-gram		
IMDB	76.09%	90.29%
Sentiment sentences	70.23%	77.64%
Movie_Review	52.22%	55.55%
n-gram / lemmatization		
IMDB	76.03%	90.365%
Sentiment sentences	70.79%	77.61%
Movie_Review	53.33%	64.44%
Adj- synonyms		
IMDB	77.64%	90.17%
Sentiment sentences	71.32%	78.11%
Movie_Review	52.22%	62.22%
NOUN- synonyms		
IMDB	75.79%	90.16%
Sentiment sentences	69.575%	77.20%
Movie_Review	53.33%	55.55%
Adj /lemmatization		
IMDB	76.99%	90.21%
Sentiment sentences	70.575%	77.11%

Movie_Review	52.22%	55.55%
Noun synonyms /lemmatization		
IMDB	75.72%	90.31%
Sentiment sentences	70.13%	76.32%
Movie_Review	48.88%	62.22%
n-gram /adj synonyms		
IMDB	77.45%	90.26%
Sentiment sentences	71.20%	77.92%
Movie_Review	53.33%	54.44%
n-gram /Noun synonyms		
IMDB	75.55%	89.84%
Sentiment sentences	70.48%	76.70%
Movie_Review	46.66%	54.44%
n-gram /adj synonyms /lemmatization		
IMDB	77.71%	90.17%
Sentiment sentences	70.98%	76.76%
Movie_Review	53.33%	50.00%
n-gram /Noun synonyms /lemmatization		
IMDB	75.36%	90.07%
Sentiment sentences	70.13%	74.95%
Movie_Review	48.88%	51.11%

Table 17:Experimental Results Using Casual_Tokenize and using FNN and CNN with one hidden layer

One hidden layer		
porter stemmer		
	FNN	CNN
IMDB	71.55%	88.78%
Sentiment sentences	63.97%	72.38%
Movie_Review	53.33%	57.77%
Snowball stemmer		
IMDB	72.52%	88.30%
Sentiment sentences	64.72%	73.51%
Movie_Review	53.33%	55.55%
Lemmatization		
IMDB	76.13%	90.57%
Sentiment sentences	68.98%	78.14%
Movie_Review	53.33%	51.11%
Uni-gram &Bi-gram		
IMDB	76.15%	89.65%
Sentiment sentences	69.82%	76.79%
Movie_Review	50.00%	61.11%
Uni-gram + Adj- synonyms		
IMDB	77.76%	89.06%
Sentiment sentences	69.442%	76.64%
Movie_Review	53.33%	57.77%
Uni-gram + NOUN- synonyms		
IMDB	75.12%	89.43%
Sentiment sentences	70.20%	76.57%
Movie_Review	55.55%	46.66%
Uni-gram &Bi-gram + lemmatization		
IMDB	76.18%	90.431%
Sentiment sentences	70.29%	77.54%

Movie_Review	53.33%	52.22%
NOUN- synonyms + lemmatization		
IMDB	75.55%	90.08%
Sentiment sentences	69.38%	76.86%
Movie_Review	48.88%	55.55%
Adj- synonyms + lemmatization		
IMDB	77.35%	89.83%
Sentiment sentences	70.10%	77.01%
Movie_Review	52.22%	55.55%
Uni-gram &Bi-gram + adj synonyms		
IMDB	77.35%	88.96%
Sentiment sentences	70.10%	75.85%
Movie_Review	53.33%	48.88%
Uni-gram &Bi-gram + noun synonyms		
IMDB	75.62%	90.07%
Sentiment sentences	70.48%	77.07%
Movie_Review	53.33%	62.22%
Uni-gram &Bi-gram + adj synonyms + lemmatization		
IMDB	77.17%	89.30%
Sentiment sentences	70.07%	76.64%
Movie_Review	54.44%	53.33%
Uni-gram &Bi-gram + noun synonyms + lemmatization		
IMDB	75.45%	88.97%
Sentiment sentences	69.91%	76.82%
Movie_Review	53.33%	48.88%

As we can see in Table 16 and 17, using CNN with IMDB, Sentiment sentences and LightSide's Movie_Reviews dataset, the best obtained result was (90.57%), (78.14%) and (68.88%). respectively, obtained when using lemmatization with the convolutional neural network. When using FNN, the best result we obtained is (77.76%) and (71.32%) respectively where sentence enrichment was performed by Adj synonyms. For LightSide's Movie_Reviews, the best obtained result is (55.55%) when using FNN with sentence enrichment by noun synonym.

4.4 Comparison between Classifiers Based on Their Execution Time

In the following tables, we present the execution times of the six classifiers that we used in the three models we proposed. Where we used same dataset to conduct the comparisons. Table 18 presents a comparison between the five classifiers used in the first proposed model. To perform the comparison we used the sentiment sentences

dataset used in the reference [26]. Hence, we have used only one combination of NLP techniques which used in our first model. That is, Porter Stemmer with remove stopwords. For tokenization step we have used Wordpunct tokenizer. The NB classifier takes the shortest amount of time to execute, as shown in Table 18. Followed by LR, SVM, FNN, and RF respectively. In Tables 19 and 20, we have compared the FFN and CNN classifiers used in the second and third proposed models respectively, where we have used FNN and CNN without hidden layers. To compare neural network classifiers using the third model, porter stemmer has been used to extract features. Result shows that the CNN takes more time to execute. The results generally confirm that machine learning classifiers are faster to implement than neural networks.

Table 18: Execution time for different classifiers used in the first model

First Proposed Model	Execution Time
SVM	17.733144521713257 seconds
NB	8.21594762802124 seconds
Logistic Regression	8.334060192108154 seconds
Random Forest	28.166030883789062 seconds
FNN	18.358084678649902 seconds

Table 19: Execution time for Neural Networks classifiers used in the second model

Second Proposed Model		Execution Time
GloVe	FNN	53.01811337471008 seconds
	CNN	99.319251537323 seconds
Word2Vec	FNN	40.89871525764465 seconds
	CNN	102.19667649269104 seconds

Table 20: Execution time for Neural Networks classifiers used in the third model

Third Proposed Model	Execution Time
FNN	859.6143283843994 seconds
CNN	15919.963688135147 seconds

4.5 Comparison with Other SA Models

In this section, we present a comparison between the results that we obtained when using the IMDB dataset with respect to similar previous works in [14] and [30]. Where

in [30] the researchers used a group of supervised learning algorithms such as the SVM, Naïve Bayes, K-Nearest Neighbor (KNN) and Maximum Entropy as we done in first part of experiments. The results obtained by the researchers are shown in Table 21. As shown in this table, the Maximum Entropy classifier proved to outperform the rest of the classifiers as it produced the highest accuracy result which is 83.93%. It was followed by the SVM classifier with an accuracy of 67.60%. In Reference [14], researchers employed SentiWordNet and n-grams feature selection to perform feature extraction and ranking. A group of supervised learning algorithms were used where the best accuracy achieved was 88.95% when using the Random Forest classifier. With the SVM classifier, we were able to achieve an accuracy of 90.43%, which confirmed the superiority of our proposed model. The authors of [9] proposed model to classify the IMDB reviews by using six layers in neural network architecture , and they utilized word weights to predict sentiment classes. In their work, the authors used two methods to extract the word polarity (positive or negative). In particular, they used two manually-constructed lists of pre-defined positive and negative words. In addition, they created word ranks by calculating sentiment and measuring domain relevance parameters. The researchers obtained a training accuracy of 91.90% and an accuracy of 86.67% for validation. In [16, 63, 64], the authors focused on using a convolutional neural network (CNN) and Long Short-term Memory (LSTM) to examine sentiment polarity . The best sentiment class prediction accuracy that was obtained using this model was 89.50%. On the basis of Table 20, we can see that our neural network-based models outperform models proposed in references [16, 63, 64]. By using the convolutional neural networks in same dataset IMDB, we were able to achieve an accuracy of 90.56 with the second model, and an accuracy of 90.57 with the third model.

Table 21:Comparison with Existing SA Models

System	Employed Classifier	Accuracy
Our Result from First Model	SVM	90.43%
Our Result from Second Model	CNN	90.56%
Our Result from Third Model	CNN	90.57%
Sentiment Analysis on IMDB Movie Reviews Using Hybrid Feature Extraction Method [30].	Maximum Entropy	83.93%
Sentiment analysis of movie reviews: A study on feature selection & classification algorithms [14]	Random Forest	88.95%
Sentiment analysis on IMDB using lexicon and neural networks. [9]	lexicon and neural networks	86.67%
Sentiment Analysis of IMDB Movie Reviews Using Long Short-Term Memory [63]	Long Short-Term Memory	89.90%
Single and Multibranch CNN-Bidirectional LSTM for IMDB Sentiment Analysis[64]	Single and Multi-branch CNN-Bidirectional LSTM	89.54%
Deep CNN-LSTM with combined kernels from multiple branches for IMDB review sentiment analysis [16]	CNN-LSTM	89.50%

4.6 Summary

In this chapter, we present the experiments that we conducted to validate the effectiveness of our proposed approach. We have also compared the results produced by our approach with some of the state-of-the-art approaches. In this chapter, we divided the experiments into three parts. In the first part, we focused on utilizing different NLP processes to extract features using the TF-IDF approach. We used SNM, NB, LR, RF, FNN classifiers for testing and validation. In the second part, we focused on using word embedding using GloVe and Word2Vec methods in addition to examining the impact of FNN and CNN parameters on classification accuracy. In the third part, we combined NLP processes from first section with word embeddings using GloVe and used FNN and CNN for testing and Validation. The experiments showed that our proposal model outperformed a group of similar works that use the same dataset (IMDB).

CHAPTER 5

Conclusions and Future Work

5.1 Conclusions and Future Work

With the advancement of the Web and social medial platforms, users are actively generating a tremendous amount of sentiment reviews on the products and services that they use. For organizations, as well as individuals, analyzing such sentiment sentences has become indispensable. To tackle this issue, several sentiment analysis techniques have been proposed, among which are the lexicon-based and machine learning based approaches. The ultimate goal of these techniques is to identify, extract and analyze the sentiment orientation of sentences that are encoded in texts which are characterized by unstructured and semantically-heterogeneous nature. However, there are still some limitations that hinder the quality of current approaches. For instance, the lexicon-based methods are still not sufficient and may result in inaccurately classifying the overall's polarity of sentiment sentences. This is mainly because a word's prior polarity doesn't reflect its contextual polarity in a sentence. On the other hand, machine learning based approaches still suffer from the issue of manually training the machine to predict the sentiment class of a given sentence, which is a time consuming, domain-dependent and error-prone task. In this research, we experimentally evaluated six probabilistic and machine learning based sentiment classifiers, namely Support Vector Machines (SVM), Naive Bayes (NB), Logistic Regression, Random Forests, CNN and FNN neural networks in first part of experiments. We evaluated the quality of (SVM, NB, LR, RF, FNN) techniques in predicting the sentiment orientation using three real-world datasets that comprised a total of 60,962 sentiment sentences. Our main goal was to study the

impact of a variety of Natural Language Processing pipelines on the quality of the predicted sentiment orientations. We also compared that with the utilization of lexical semantic knowledge captured by WordNet. Findings demonstrate that there is an impact of varying the employed NLP pipelines and the incorporation of semantic relationships on the quality of the sentiment analyzers. In particular, results indicate that coupling lemmatization and knowledge-based n-gram features proved to produce higher accuracy results when applied on the IMDB dataset. With this coupling, the accuracy of the SVM classifier has improved to be 90.43%. For the three other classifiers (NB, Random Forest, Logistic Regression and Artificial Neural Network), the quality of the sentiment classification has also improved to be 86.83%, 90.11%, 86.20%, 88.01%, respectively. It is important to highlight the fact that the used pipeline phases are of less complexity in terms of their structure, as well as computational cost when compared with other deep learning models. As such, we conclude that obtaining highly accurate sentiment prediction results can still be achieved using a composition of conventional NLP processes that are comparable and, in some cases, more superior than complex architectures. In second part of experiments, we employed two types of neural networks for sentiment analysis, namely, the Convolutional neural network CNN and the feedforward neural network (FNN). We studied a set of variables in the neural network to determine how they affect the accuracy of sentiment classification. These variables include the number of hidden layers used in the network, the activation function used in these layers, as well as the size of the feature maps, the size of the filter window, and the activation function used in the convolutional neural network. In addition, we used the glove and word2vec embedding for word vectorization and different representations supported by glove. While in first part we used TF-IDF for feature vectorization. In the second part of our experiments, we used four data sets that included 50,000 movie

reviews, 10,662 sentences, 300 public movie reviews, and 1,600,000 tweets. Results show that that when we used GloVe embedding's, we got better results than when we used Word2Vec embeddings. The result also shows that when Glove or word2vec word embedding is used with a large word dimension, accuracy increases. Moreover, we found that the convolutional neural network's accuracy improved with a larger feature map, a smaller filter window, and using ReLU activation functions. The neural network's classification accuracy was also improved by using multiple activation functions in the hidden layers. In this part of experiments, it can be noticed that the accuracy has improved to (90.56%) and (80.73%) when tested on the IMDB dataset t and the sentiment 140 dataset respectively. In the third part of experiments, we used the feature extraction methods of the first model in addition to using GloVe embeddings for feature vectorization and (CNN, FNN) for testing and validation. In particular, results indicate that the highest accuracy obtained when using lemmatization with the convolutional neural network. when applied this combination on the IMDB dataset we obtained (90.57%) accuracy.

In the future work, we will study the impact on varying the composite futures extracted from sentiment sentences on the quality of neural network and pre-trained text processing models. For this purpose, we will utilize the pipeline phase introduced in this thesis to evaluate the BERT models.

References

1. Hussein, D.M.E.-D.M., *A survey on sentiment analysis challenges.* Journal of King Saud University - Engineering Sciences, 2018. **30**(4): p. 330-338.
2. Wang, M., S. Chen, and L. He. *Sentiment Classification Using Neural Networks with Sentiment Centroids*. in *PAKDD*. 2018.
3. Alam, S. and N. Yao, *The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis.* Computational and Mathematical Organization Theory, 2019. **25**(3): p. 319-335.
4. Yang, P. and Y. Chen. *A survey on sentiment analysis by using machine learning methods*. in *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. 2017. IEEE.
5. Mohey El-Din, D., *A Survey on Sentiment Analysis Challenges.* Journal of King Saud University - Engineering Sciences, 2016.
6. Renault, T., *Sentiment analysis and machine learning in finance: a comparison of methods and models on one million messages.* Digital Finance, 2020. **2**(1): p. 1-13.
7. Rao, A.V.S.R. and P. Ranjana, *Empower good governance with public assessed schemes by improved sentiment analysis accuracy.* Electronic Government, an International Journal, 2020. **16**(1-2): p. 118-136.
8. Araque, O., G. Zhu, and C.A. Iglesias, *A semantic similarity-based perspective of affect lexicons for sentiment analysis.* Knowledge-Based Systems, 2019. **165**: p. 346-359.
9. Shaukat, Z., et al., *Sentiment analysis on IMDB using lexicon and neural networks.* SN Applied Sciences, 2020. **2**.
10. Latha, I., *Sentiment Analysis Tool using Machine Learning Algorithms.* 2019: p. 14791-14794.
11. Patel, R. and K. Passi, *Sentiment Analysis on Twitter Data of World Cup Soccer Tournament Using Machine Learning.* IoT, 2020. **1**(2): p. 218-239.
12. Al Amrani, Y., M. Lazaar, and K.E. El Kadiri, *Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis.* Procedia Computer Science, 2018. **127**: p. 511-520.

13. Ahmad, M., et al., *Machine learning techniques for sentiment analysis: A review.* Int. J. Multidiscip. Sci. Eng, 2017. **8**(3): p. 27.
14. Sahu, T.P. and S. Ahuja. *Sentiment analysis of movie reviews: A study on feature selection & classification algorithms*. in *2016 International Conference on Microelectronics, Computing and Communications (MicroCom)*. 2016.
15. Vielma, C., A. Verma, and D. Bein. *Single and Multibranch CNN-Bidirectional LSTM for IMDb Sentiment Analysis*. 2020.
16. Yenter, A. and A. Verma. *Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis*. in *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*. 2017.
17. Zhang, L., S. Wang, and B. Liu, *Deep learning for sentiment analysis: A survey.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2018. **8**(4): p. e1253.
18. Moraes, R., J.F. Valiati, and W.P. Gavião Neto, *Document-level sentiment classification: An empirical comparison between SVM and ANN.* Expert Systems with Applications, 2013. **40**(2): p. 621-633.
19. Haddi, E. *Sentiment analysis: text, pre-processing, reader views and cross domains*. 2015.
20. Horakova, M., *Sentiment Analysis Tool using Machine Learning.* Global Journal on Technology, 2015.
21. Kharde, V. and S. Sonawane, *Sentiment Analysis of Twitter Data: A Survey of Techniques.* International Journal of Computer Applications, 2016. **139**: p. 5-15.
22. Meškelė, D. and F. Frasincar, *ALDONAr: A hybrid solution for sentence-level aspect-based sentiment analysis using a lexicalized domain ontology and a regularized neural attention model.* Information Processing & Management, 2020. **57**(3): p. 102211.
23. Chen, R., et al., *Word-level sentiment analysis with reinforcement learning.* IOP Conference Series: Materials Science and Engineering, 2019. **490**: p. 062063.
24. Wilson, T., J. Wiebe, and P. Hoffmann, *Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis.* Computational linguistics, 2009. **35**(3): p. 399-433.

25. Kim, S.-M. and E. Hovy, *Determining the sentiment of opinions*, in *Proceedings of the 20th international conference on Computational Linguistics*. 2004, Association for Computational Linguistics: Geneva, Switzerland. p. 1367–es.
26. Maree, M. and M. Eleyat, *SEMANTIC GRAPH BASED TERM EXPANSION FOR SENTENCE-LEVEL SENTIMENT ANALYSIS.* International Journal of Computing, 2020. **19**(4): p. 647-655.
27. Tierney, B., *Sentiment Classification of Reviews Using SentiWordNet.* 2009.
28. Kundi, F.M., et al., *Detection and scoring of internet slangs for sentiment analysis using SentiWordNet.* Life Science Journal, 2014. **11**(9): p. 66-72.
29. kumar, v., *Sentiment Analysis Techniques for Social Media Data: A Review*. 2019.
30. Keerthi Kumar, H.M., B.S. Harish, and H. Darshan, *Sentiment Analysis on IMDb Movie Reviews Using Hybrid Feature Extraction Method.* International Journal of Interactive Multimedia and Artificial Intelligence, 2018. **InPress**: p. 1.
31. Pang, B., L. Lee, and S. Vaithyanathan, *Thumbs up? Sentiment Classification Using Machine Learning Techniques.* EMNLP, 2002. **10**.
32. Bespalov, D., et al., *Sentiment classification based on supervised latent n-gram analysis*, in *Proceedings of the 20th ACM international conference on Information and knowledge management*. 2011, Association for Computing Machinery: Glasgow, Scotland, UK. p. 375–382.
33. Sharma, A. and S. Dey. *A comparative study of feature selection and machine learning techniques for sentiment analysis*. in *Proceedings of the 2012 ACM research in applied computation symposium*. 2012.
34. Pak, A. and P. Paroubek, *Twitter as a Corpus for Sentiment Analysis and Opinion Mining*. Vol. 10. 2010.
35. Du, C., M. Tsai, and C. Wang. *Beyond Word-level to Sentence-level Sentiment Analysis for Financial Reports*. in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019.
36. Basha, S.M. and D.S. Rajput, *Evaluating the Impact of Feature Selection on Overall Performance of Sentiment Analysis*, in *Proceedings of the 2017 International Conference on Information Technology*. 2017, Association for Computing Machinery: Singapore, Singapore. p. 96–102.

37. Haddi, E., X. Liu, and Y. Shi, *The Role of Text Pre-processing in Sentiment Analysis.* Procedia Computer Science, 2013. **17**: p. 26-32.
38. Sohrabi, M.K. and F. Hemmatian, *An efficient preprocessing method for supervised sentiment analysis by converting sentences to numerical vectors: a twitter case study.* Multimedia tools and applications, 2019. **78**(17): p. 24863-24882.
39. Krouska, A., C. Troussas, and M. Virvou, *The effect of preprocessing techniques on Twitter sentiment analysis*. 2016. 1-5.
40. Stojanovski, D., et al. *Twitter sentiment analysis using deep convolutional neural network*. in *International Conference on Hybrid Artificial Intelligence Systems*. 2015. Springer.
41. Pang, B. and L. Lee, *A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts.* arXiv preprint cs/0409058, 2004.
42. Maree, M., *Semantics-based key concepts identification for documents indexing and retrieval on the web.* Int. J. Innov. Comput. Appl., 2021. **12**(1): p. 1–12.
43. Makrehchi, M. and M.S. Kamel. *Automatic Extraction of Domain-Specific Stopwords from Labeled Documents*. in *Advances in Information Retrieval*. 2008. Berlin, Heidelberg: Springer Berlin Heidelberg.
44. Maree, M., A.B. Kmail, and M. Belkhatir, *Analysis and shortcomings of e-recruitment systems: Towards a semantics-based approach addressing knowledge incompleteness and limited domain coverage.* Journal of Information Science, 2019. **45**(6): p. 713-735.
45. Nasra, I. and M. Maree. *On the use of Arabic stemmers to increase the recall of information retrieval systems*. in *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*. 2017.
46. Porter, M.F., *An algorithm for suffix stripping.* Program, 1980.
47. Porter, M.F., *Snowball: A language for stemming algorithms*. 2001.
48. Maree, M., et al., *A Knowledge-based Model for Semantic Oriented Contextual Advertising.* KSII Transactions on Internet and Information Systems (TIIS), 2020. **14**(5): p. 2122-2140.

49. Abbasi, A., et al., *Selecting attributes for sentiment classification using feature relation networks.* IEEE Transactions on Knowledge and Data Engineering, 2010. **23**(3): p. 447-462.
50. Nicholls, C. and F. Song. *Comparison of feature selection methods for sentiment analysis*. in *Canadian Conference on Artificial Intelligence*. 2010. Springer.
51. O'Keefe, T. and I. Koprinska. *Feature selection and weighting methods in sentiment analysis*. in *Proceedings of the 14th Australasian document computing symposium, Sydney*. 2009. Citeseer.
52. El Mrabti, S., M. Al Achhab, and M. Lazaar. *Comparison of feature selection methods for sentiment analysis*. in *International Conference on Big Data, Cloud and Applications*. 2018. Springer.
53. Ahmad, S.R., A.A. Bakar, and M.R. Yaakub, *Ant colony optimization for text feature selection in sentiment analysis.* Intelligent Data Analysis, 2019. **23**(1): p. 133-158.
54. Al-Twairesh, N. and H. Al-Negheimish, *Surface and deep features ensemble for sentiment analysis of arabic tweets.* IEEE Access, 2019. **7**: p. 84122-84131.
55. Chi, X., T.P. Siew, and E. Cambria. *Adaptive two-stage feature selection for sentiment classification*. in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2017. IEEE.
56. Kübler, S., C. Liu, and Z.A. Sayyed, *To use or not to use: Feature selection for sentiment analysis of highly imbalanced data.* Nat. Lang. Eng., 2018. **24**(1): p. 3-37.
57. Arafat, H., et al., *Different feature selection for sentiment classification.* International Journal of Information Science and Intelligent System, 2014. **1**(3): p. 137-150.
58. Varela, P.L., et al. *An empirical study of feature selection for sentiment analysis*. in *9th conference on telecommunications, Conftele, Castelo Branco*. 2013.
59. Ni, R. and H. Cao. *Sentiment Analysis based on GloVe and LSTM-GRU*. in *2020 39th Chinese Control Conference (CCC)*. 2020. IEEE.
60. Zhang, Y. and B. Wallace, *A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification.* arXiv preprint arXiv:1510.03820, 2015.

61. Dos Santos, C. and M. Gatti. *Deep convolutional neural networks for sentiment analysis of short texts*. in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 2014.

62. Kamila, N.K., *Handbook of research on emerging perspectives in intelligent pattern recognition, analysis, and image processing*. 2015: IGI Global.

63. Qaisar, S.M. *Sentiment Analysis of IMDb Movie Reviews Using Long Short-Term Memory*. in *2020 2nd International Conference on Computer and Information Sciences (ICCIS)*. 2020.

64. Vielma, C., A. Verma, and D. Bein. *Single and Multibranch CNN-Bidirectional LSTM for IMDb Sentiment Analysis*. in *17th International Conference on Information Technology–New Generations (ITNG 2020)*. 2020. Cham: Springer International Publishing.

الملخص باللغة العربية

أصبح تحليل المشاعر (SA) أحد أكثر الأدوات موثوقية لمساعدة المؤسسات على فهم أفضل لتصور مستخدميها حول المنتجات والخدمات التي يقدمونها. على وجه الخصوص، مع المراجعات المتزايدة للمشاعر التي ينشئها المستخدمون على الويب، أصبحت أدوات SA لا غنى عنها بشكل كبير. إن استغلال هذه الأدوات في مجالات التطبيق الواقعية لا يخدم المؤسسات فحسب، بل يخدم أيضًا الأفراد المهتمين بمعرفة المزيد حول التصورات المختلفة حول المنتجات و/أو الخدمات التي يستخدمها المستخدمون أو العملاء الآخرون. على مدى السنوات القليلة الماضية، كان هناك عدد متزايد من تقنيات SA التي يمكن أن تتميز بعدد من نقاط القوة والضعف؛ أثبتت من خلال معدلات دقتها من حيث النتائج التي تنتجها. كانت مناهج التعلم القائم على المعجم والآلة من بين التقنيات الأكثر شيوعًا المستخدمة لهذا الغرض على وجه الخصوص. لمعالجة قيود هذه التقنيات ،تم اقتراح نماذج أحدث للشبكات العصبية في محاولة لأتمتة عملية تعلم الميزات وإثراء الميزات المكتسبة بتضمين كلمات السياق لتحديد توجهاتها الدلالية. ومع ذلك، فإن معدلات الدقة لمثل هذه النماذج تستلزم أن تندرج الجمل تحت نفس مجال بيانات التدريب المستخدمة مسبقًا.

في هذه الأطروحة، قمنا بتقييم ستة مصنّفات للمشاعر بشكل تجريبي، وهي آلات المتجهات الداعمة(SVM) ، و Naive Bayes (NB) ، والانحدار اللوجستي ، والغابات العشوائية ، والشبكة العصبية التغذوية ، والشبكة العصبية التلافيفية. بالإضافة إلى التقييم التجريبي للطرق التقليدية الأربعة الأولى، نقيس أيضًا تأثير دمج المعرفة الدلالية المعجمية التي تم التقاطها بواسطة WordNetعلى توسيع الكلمات الأصلية في الجمل باستخدام مجموعة متنوعة من خطوط أنابيب البرمجة اللغوية العصبية. تعتمد الطريقتان الأخيرتان على الشبكات العصبية مع المعالجة الآلية للميزات والقدرة على إثراء الميزات المكتسبة بتضمين كلمات السياق. إلى جانب قياس جودتها، فإننا نحقق بشكل تجريبي في تأثير استغلال زخارف GloVe word على إثراء نواقل الميزات المستخرجة من جمل المشاعر. في التجارب التي تم إجراؤها، استخدمنا أربع مجموعات بيانات حقيقية تتكون من 1600000 تغريدة و 50000 مراجعة فيلم و 10662 جملة و 300 مراجعة عامة للأفلام. فيما يتعلق بالطرق الخمس الأولى، تشير النتائج إلى أن اقتران lemmatization وميزات n-

gram القائمة على المعرفة أثبتت أنها تنتج معدلات دقة أعلى. مع هذا الاقتران، تحسنت دقة مصنف SVM إلى 90.43٪، بينما كانت 86.83٪، 90.11٪، 86.20٪، 88.01٪ على التوالي باستخدام المصنفات الأربعة الأخرى. تظهر النتائج أن المصنف NB يستغرق أقصر وقت للتنفيذ. يليه LR و SVM و FNN و RF على التوالي. بالنسبة للطرق القائمة على الشبكات العصبية FNN) و (CNN، تشير النتائج إلى أن استخدام أبعاد أكبر لتضمينات GloVe word يزيد من دقة تصنيف المشاعر. على وجه الخصوص، توضح النتائج أن الدقة المحققة لشبكة CNN باستخدام خريطة ميزات أكبر، وحجم مرشح أصغر، بالإضافة إلى وظيفة تنشيط ReLU في الطبقة التلافيفية كانت 90.57٪ عند تطبيقها على مجموعة بيانات IMDB، بينما كانت 82.02٪ و 78.14 ٪ باستخدام مجموعات بيانات تويتر وجمل المشاعر، على التوالي.