SENTIMENT LEXICON-AUGMENTED TEXT REPRESENTATION MODEL FOR SOCIAL MEDIA TEXT SENTIMENT ANALYSIS

JAMES SAVALI MUTINDA

DOCTOR OF PHILOSOPHY

(Information Technology)

JOMO KENYATTA UNIVERSITY

OF

AGRICULTURE AND TECHNOLOGY

2024

Sentiment Lexicon-Augmented Text Representation Model for Social Media Text Sentiment Analysis

James Savali Mutinda

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Information Technology of the Jomo Kenyatta University of Agriculture and Technology

2024

DECLARATION

This research thesis is my original work and has not been presented for a degree in any other University

Signature……..Date……..

James Savali Mutinda

This thesis has been submitted for examination with our approval as the University Supervisors

Signature……..Date……..

Prof. Waweru Mwangi, PhD

JKUAT, Kenya

Signature……..Date……..

Dr. George Okeyo, PhD

JKUAT, Kenya

DEDICATION

To my late parents Agnes Kaswii and Mutinda Mbuvi, the entire Mutinda family; my brothers and sisters. To my family; wife Faith, my daughters, Patience, Innocent, and Grace, and my son Mark Mutinda.

ACKNOWLEDGEMENT

This Ph.D. thesis is a result of several years of intense research, experiments, and support from several people without whom this research would not have been realized and to whom I am grateful.

First of all, I thank almighty God for the strength he has given me throughout the time I have been carrying out this research work. His grace, care, and mercies have kept me during the entire period. I sincerely thank my supervisors; Professor Waweru Mwangi and Professor George Okeyo for their academic and scholarly guidance. Their probing questions awakened my thinking in the area of Machine Learning and Natural Language Processing. You focused my research and took the time to guide me when I was writing journal papers. You were co-authors in the papers in which you contributed and tirelessly edited before publication. You were patient with my mistakes and corrected me accordingly.

About Kenya School of Government, first of all, I would like to thank Prof. Ludeki Chweya the Director General for approval of the course and financial support, and my colleagues whose comments and advice shaped this research work. Thanks to the staff of the School of Computing and Information Technology, JKUAT for the feedback, comments, and guidelines during and after departmental seminar presentations. Sincere gratitude goes to my family for financial and moral support during the study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

ACRONYMS AND ABBREVIATIONS

SCIT	School of Computing and Information Technology
JKUAT	Jomo Kenyatta University of Agriculture and Technology
NLP	Natural Language Processing
SPSS	Statistical Package for Social Scientists
TF-IDF	Term Frequency-Inverse Document Frequency
POS	Part of Speech
ML	Machine Learning
TC	Text Categorization
SVM	Support Vector Machines
SA	Sentiment Analysis
PMI-IR	Pointwise Mutual Information(PMI) and Information Retrieval (IR)
RT	Re-Tweeting
DBMS	Database Management Systems
PHP	Hypertext Pre-processor server-side scripting language
LSA	Latent Semantic Analysis
MS	Microsoft
BERT	Bidirectional Encoder Representations from Transformers
LeBERT	Lexicon-enhanced Bidirectional Encoder Representations from Transformers
LENFEM	Lexicon-pointed N-gram Feature Extraction Model

DEFINITION OF OPERATIONAL TERMS

Sentiment Term is the discriminator word that is used to portray a certain sentiment in a sentence.

Big Twitter micro blogs are Twitter posts. Big is used to mean enormous since the tweets used in this research are expected to be thousands in number. The term big is from big data analytics. Micro blogs are short sentences posted by users in Twitter or in other social media platforms.

ABSTRACT

Social media platforms have provided dominant arenas for people to express their opinions and thoughts about products, services, individuals, and public policy in the form of posts. These posts are characterized by high volume, unstructured, semi-structured, and normally full of colloquial language thus accurate sentiment analysis models for social media data are required. Text representation is a key determinant of the accuracy and computational cost of machine learning models for such sentiment analysis. Existing text representation techniques do not consider relationships between words, they ignore words' characteristics for instance word sentiment orientation and they suffer high feature dimensionality. The high dimensionality is attributed to the brute-force approach of generating representation vectors from the entire input text. This research aimed to develop and evaluate a sentiment lexicon-augmented text representation model for social media sentiment analysis. Three public datasets (online reviews) from Amazon product reviews, Yelp Restaurants' reviews, and IMDB Movies' reviews were used. Pre-processing involved cleaning the reviews, tokenization, lemmatization, and Part-of-speech (POS) tagging. Text representation was done using a bag of words, N-grams, hybrid representations, word embeddings, and the proposed sentiment lexicon-augmented approaches. Term Frequency-Inverse Document Frequency (TF-IDF) and Binary Occurrences were used as term weighting algorithms. The resultant text representation vectors obtained were used as input in four supervised machine learning base classifiers (Decision Tree, K-Nearest Neighbor, Naïve Bayes, and Support Vector Machines) and deep learning's Convolutional Neural Network (CNN). Experimental results from sentiment lexicon-enhanced approaches showed that they performed better than other baseline approaches with an F-measure score that ranged between 84.68% and 90.15%. Ablation studies on the *N*-grams showed that N=3 performed better than other values of N=1, 2, 4, or *n* with an F-measure score of 88.73%. Using base classifiers, Support Vector Machines were found to perform better than Naïve Bayes, K-Nearest Nei, neighbor, and Decision Tree. Using the sentiment lexicon-augmented word embeddings and CNN, the results showed that Bidirectional Encoder Representation from Transformers (BERT) outperformed the Global Vectors (Glove) and Word2Vec embeddings with an F-measure of 88.63%. The results of binary sentiment classification before and after sentiment lexicon enhancement showed a reduction of resultant vector feature dimensions and improvement in sentiment classification models' performance both in base machine learning classifiers and in deep learning CNN. This study demonstrated that the sentiment lexicon-based approach combined with conventional text representation algorithms can be used in enhancing sentiment analysis models for social media data.

Keywords: Sentiment Analysis, Machine Learning, Text Representation, Social Media, and Word Embedding.

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Text classification and Sentiment Analysis (SA) are Text Mining technologies that are used to generate knowledge and create meaning from large volumes of high-dimensional text data. Advancements in web technologies and social media platforms have led to a fast expansion of social media comments in various industries raising the need to extract useful information from these comments (Talaat, 2023). Social media platforms have emerged as powerful arenas for people to express their opinions and thoughts, making them ideal for capturing and analyzing public sentiments about products, services, individuals, and public policy (Jain et al., 2023). Social media text data contains short sentences in the form of posts. These posts are characterized by texts that are unstructured, semi-structured, and normally full of colloquial language thus knowledge discovery and information extraction from these data consumes a lot of time and is computationally expensive (Brindha et al., 2016). Therefore, sentiment analysis of social media data has become an important area of focus in big data analytics.

This chapter thus, outlines the background of the study which shows the basic concepts in sentiment analysis and the basis of carrying out the research in the area of sentiment analysis. The research gap is also outlined in the chapter as the problem statement. Research objectives, research questions, and the organization of this thesis are also presented.

1.1.1 Concept of Text Mining

Recently, textual data has been availed online through several avenues including user-generated content as is the case of social media, digital libraries, and the entire worldwide web. This has made online data an interest of many; scholars, economists, and psychologists among others (Kaushik et al., 2015). For instance, social media data have had d tremendous impact on people's lives as they help people to interact, share

ideas, as well as influence their peers' decisions. Not only do social media bring convenience to users, but also the businesses for it enables merchants to increase the Quality of Experience QoEE) of their customers (Li et al., 2020). This is achieved by analyzing social media feedback data from customers and then addressing the issues arising therefrom. The availability of social media platforms coupled with their applications at personal and organizational levels has led to the growth of social media data. However, this growth has created serious challenges when analyzing this data because of the nature of the data commonly known as posts which are unstructured, full of colloquial syntax, and attention is unevenly distributed and fast. Consequently, how to analyze tons of this data which is in the form of text, images or videos efficiently has opened up new research opportunities in social media sentiment analysis (Mingyong & Yang, 2012) and (Kaushik et al., 2015).

Text mining is one of the appropriate approaches that can be used to extract the most relevant information from the whole junk of online text data. Text mining also referred to as text data mining or text analytics is the processing and analyzing of text data to derive high-quality information and knowledge from text (Gosh et al., 2012). Text mining can also be defined as the process of structuring the input text, deriving linguistic features from it, removing unnecessary features, inserting them in a database, deriving patterns within the unstructured data, and finally evaluating and interpreting the output (Gosh et al., 2012). Text mining is a process that involves the application of various techniques from information retrieval, natural language processing, information extraction, and data mining (Mingyong & Yang, 2012). Since text mining leads to knowledge discovery from text data, a lot of commercial value has been attached to it.

In text classification, automatic text classifiers are built by use of machine learning (ML) techniques which are capable of labeling natural language texts into a predefined thematic labels' set $C = \{c_1, \ldots, c_n\}$ (Sebastian, 2002). Where; c_1, …, c_n are the possible labels. In binary classification there are only two labels. The construction of an automatic text classifier requires the existence of a collection of documents in an initial corpus $D = \{d_1, \ldots, d_n\}$. Where, $d_1, \ldots, d_n$ are the documents which are to be labeled into the predefined classes. The classes are dependent on the classification task

at hand. The corpus is divided into training (Tr) and testing sets (Ts). The machine learning technique is a general inductive process that automatically builds a classifier for *D* by learning the characteristics of the corpus from a training set $Tr = \{d_1, \ldots, d_t\}$ of the entire documents. Once a classifier has been built, its effectiveness and efficiency in making the right labeling decisions may be tested by applying it to the test set $Ts = \{D - Tr\}$ and checking the degree of correspondence between the decisions of the classifier and those encoded in the corpus. Such classifiers are referred to as supervised machine classifiers since they're dependent on already labeled training and testing documents in the sets. In the absence of the labeled data sets the classifier is referred to as an unsupervised machine learning classifier (Hassan et al., 2022).

The construction of a text machine learning classifier consists of two phases. These are the document indexing phase and the classifier learning phase. The document indexing phase involves the creation of internal representations for documents which can be the input for the machine learning classifier. The internal representations are derived from the extraction of text features that fully represent the document and the selection of the best set of features that yield the best classification. These features in natural language include words, sentences, phrases, and Part of Speech tags among others. At this phase, the document or text is represented as a vector of numbers that represent it thus also called text vectorization. The classifier learning phase involves the creation of a machine learning model by using the internal representations of the training documents. Text mining has several applications in information technology including sentiment analysis (Liu, 2010).

1.1.2 Concept of Sentiment Analysis

Sentiment analysis is the process of categorizing a text into two or more sentiment or opinion labels. Specifically, the process of sentiment analysis involves determining whether a text or document expresses a positive or a negative opinion. In this respect, sentiment analysis can be interpreted as a text classification task where each category represents a sentiment or an opinion.

Sentiment analysis focuses on the whole document, sentences, or the opinion holder. In document sentiment analysis the sentiments represented are categorized for the

entire document. In sentence-level sentiment analysis, the semantic label is assigned to a specific sentence. Since social media is made of short statements, this research focused on sentence-level sentiment analysis. Sentiment analysis is divided into four steps:

Step I: Object identification. This step involves discovering the object about which an opinion has been provided. An object can be a product, service, an occurrence, or an event.

Step II: The second phase is the feature extraction and representation phase. At this stage, the specific features of the opinion text are identified and used to create a representation of the text that can be input into a computer.

Step III: The third phase is the opinion orientation classification. The objective of this phase is to find out whether there is an opinion in a given sentence or document. In the case of sentence-level sentiment analysis, the sentences are either classified as objective or subjective sentences. Objective sentences are sentences without any opinion while subjective sentences have an opinion that is positive, negative, or neutral.

Step IV: The fourth phase is sentiment analysis (Liu B., 2010). In this phase, the specific sentiment or opinion of the subjective sentence is identified. The focus of this phase is to analyze the nature of the opinion or sentiment portrayed. One of the key issues is to identify opinion words and phrases, which are instrumental in sentiment analysis. The problem is that there are seemingly an unlimited number of expressions that people use to express opinions, and in different domains, they can be significantly different. Even in the same domain, the same word may indicate different opinions in different contexts. According to Appel et al. (2015), mining opinions and sentiments from sentence and document text is challenging since it requires a deep understanding of the explicit and implicit, regular and irregular, and syntactical and semantic language rules. This research work focused on opinion orientation classification of social media text data. This is because short text data in the form of sentences is daily increasing and is availed mostly through social media and the World Wide

Web in the form of user comments and reviews commonly referred to as posts (Li et al., 2020).

Sentiment analysis of these comments and reviews can enable businesses to optimize their decisions on customer's opinions about the products they offer (Gu, 2020). People are also keen on making decisions based on other people's opinions, governments and political leaders use sentiment analysis to discover citizens' preferences and opinions on governance issues (Brindha, 2016) (Santur, 2019). Not only do social media platforms bring convenience to users but also to businesses for they enable merchants to increase the Quality of Experience (QOE) of their customers (Gu, 2020). For instance, sentence-level sentiment analysis of such data enables businesses to understand their customers' opinions about the products they offer and their experiences (Brindha et al., 2016). The analysis results are then utilized by the businesses to improve their services based on their opinions. Social media data can also be utilized in other areas including governance in collecting citizens' opinions and also by security agencies. Therefore, as businesses and other institutions use these platforms to collect opinions, it's paramount that sentiment analysis of such data is done for insightful knowledge and information that can lead to the growth of the businesses.

Despite the potential benefits, sentiment analysis of social media texts still poses challenges to data analytics. This is because such content is characterized by short sentences that are unstructured, semi-structured, high in volume, and normally full of colloquial language thus posing a challenge in the effort to analyze them (Brindha et al., 2016), (Mingyong & Yang, 2012), (Haddi et al., 2013). Thus identification of appropriate features to represent the sentences for input in classification models is an active research problem (Talaat, 2023) (Jain et al., 2023). Further coming up with robust models to classify the sentences effectively and efficiently into specified sentiment classes is an active research area.

1.1.3 History of Sentiment Analysis

Sentiment analysis (SA) is an area in machine learning whose research and study have grown rapidly in the recent past. SA is as old as verbal communication is. This is

because the interest in other people's opinions can be traced to the invention of verbal communication itself (Mäntylä et al., 2018). Although SA started as a manual process, it has grown into a computer-based process with the most current technologies of machine learning. The roots of sentiment analysis can be traced from the beginning of the 20th century when the focus of sentiment analysis was on public opinion analysis. Sentiment analysis on a subject or a leader was done through voting where public opinion was quantified and measured from questionnaires or questions directed to the public (Droba, 1931). This was referred to as traditional sentiment analysis where the analysis was on someone's opinion towards something as positive, neutral, or negative opinion (Dave et al., 2003).

The traditional SA was closely followed by text subjectivity analysis studies performed by the computational linguistics community in the 1990's. The community gave birth to computer-based sentiment analysis where techniques to detect subjective sentences from a document were proposed and studied (Wiebe et al., 1999). Later, in the early 2000s, there was an outbreak of computer-based sentiment analysis which was necessitated by the growth of the World Wide Web. This growth led to the availability of large amounts of text in online platforms. For instance, in 2002, research by Turney (2002) was done on SA using Machine Learning (ML) Techniques that utilized movie review datasets and found that machine-learning classification outperformed human-produced baselines. Generally, there was an influx of SA analysis studies using online reviews available on the World Wide Web such as movie and product reviews. The inventors of Machine Learning-based SA argued that automated analysis of such reviews is better than human-based due to the high volumes of such reviews which may not be handled manually.

Since then, Machine Learning SA has become an active research area with current trends moving towards the application of deep learning techniques in sentiment analysis.

1.1.4 Text Representation in Sentiment Analysis

Text representation is the process of converting text into vectors that represent the text and can be used to predict the sentiment class of the text (Chug et al., 2019). Text

representation is also referred to as vectorization. After vectorization, the formed vector may contain irrelevant and redundant features. Feature selection is the process of determining the most robust attributes or features from the vector. Therefore, text representation and feature selection play an important role in determining the performance of SA models (Aytug & Serdar, 2017), (Bhadane, Dalal, & Doshi, 2015). There are several text representation techniques applicable in Sentiment Analysis. They include; Bag of Words (Mozetic et al., 2016), TF-IDF (Aytug & Serdar, 2017) (Ankit , 2018), Natural Language Processing (NLP)-based features such as word counts, nouns count or N-grams (Chug et al., 2019) and deep learning-based word embeddings techniques such as Glove, Word2vec, and BERT (Kenton & Toutanova, 2019) (Hu et al., 2022). Most of these techniques depend on a bag of words and word-frequency algorithms to generate text vectors. However, it has been observed that the frequency-based algorithms yield better sentiment analysis results when applied in large documents than in shorter texts or sentences since in a short text such as a sentence the frequency of a sentiment term is mostly one (Tian & Yanmei, 2011) (Ramos, 2003). Thus, such techniques yield poor sentiment analysis results when applied to short texts such as social media posts.

Recently attention to feature extraction techniques that focus on a part or a section of a document or text has increased. This has mainly been done on document-level sentiment analysis where the topic, introduction, or conclusion of the document is considered (Srivastava, 2018). This implies that an opinion in a subjective sentence or document may not necessarily be in the entire sentence or document but in a specific part. Most of the earlier techniques generate text vectors from the entire sentence or document. Nevertheless, existing works such as that of Srivastava (Srivastava, 2018) that focus on specific parts of a sentence for feature extraction do not clearly show how the sentences are classified but concentrate on the generation of word dictionaries and word sentiments. Most research experiments that consider specific words in a sentence use word2vec models (Mikolov et al., 2013) to calculate word orientations and their similarities with other words as is the case of a continuous bag of words (CBOW) and Skip-gram models (Rezaeinia et al., 2019). However, vectors obtained through such models do not consider word arrangements, the semantics of the words, and their contextual information (Parwez, 2019). For effective text representation of

short texts for sentiment classification, it is necessary to include other linguistic aspects such as POS tags, word semantic orientations, and word arrangement.

Further, machine learning algorithms have been widely used in text classification with deep learning techniques achieving great attention recently in many fields including sentiment analysis (Gu, 2020). Such machine learning algorithms include decision trees, naïve bayes (Brindha et al., 2016), support vector machines (Hassan et al., 2022), and K-Nearest Neighbour among others. Brindha et al., (2016) used decision trees, naïve bayes, and K-Nearest Neighbour for sentiment analysis. In deep learning, Convolutional Neural Networks (CNN) have brought new potential in sentiment analysis due to their powerful feature extraction ability which can better learn context information and the semantics of words (Samira et al., 2018). However, these classifiers have their drawbacks when used to model sentiment analysis. For instance, existing research on CNN mainly depends on pre-trained word vectors and ignores the semantics of words (Yang 2022). Moreover, machine learning algorithms are not domain-specific but sentiments are domain-specific thus impacting the accuracy and generalizability of such sentiment analysis models. Little research has been done on combining deep learning techniques and traditional text representation techniques for sentiment analysis (Santur, 2019). Research on combining traditional machine learning and deep learning text representation techniques in sentiment analysis is thus low and an area worth investigating.

The main purpose of this research work was to develop and investigate a systematic approach to text representation that can be used to effectively and efficiently classify subjective social media posts (sentences). The research work investigated the use of sentiment lexicon in enhancing existing text representation techniques for sentiment analysis models applicable in short text such as social media posts. A sentiment lexicon is a dictionary or a list of sentiment words showing their sentiment orientation as either positive or negative (Kukkar et al., 2023). The study sought to investigate how sentiment lexicon can be combined with existing text representation techniques to build a sentiment-rich text vector. The research work advances sentiment analysis research by solving the problem of high feature dimensionality of text vectors built from state-of-the-art text representation techniques. The performance of the text

representation and selection approaches were validated using supervised machine learning classifiers which are widely used (Ankit, 2018) (Bird et al., 2019).

1.2 Problem Statement

Various industries are using social media data collected from their clients and the public's opinions about their products and services through reviews and posts. Social media data have been processed for decision-making in organizations, governments, and other social formations (Jain et al., 2023). This social media data is regarded as big data characterized by high volume and short texts that are unstructured, semi-structured, and normally full of colloquial language (Brindha et al., 2016). Most importantly, social media short texts do not provide sufficient context and are often malformed meaning the text is incomplete (Sakor et al., 2019). Thus, knowledge discovery and information extraction from these data is a challenging task in text mining, sentiment analysis, and information retrieval communities (Talaat, 2023). Automated Sentiment Analysis and Text Mining come in handy to analyze the opinions from such data due to its diversity and size. However, sentiment analysis of social media data is a challenge to data analysts (Han & Kwangmi, 2017) and (D'Silva & Sharma, 2022). The challenges arise since current text representation techniques used in sentiment analysis of social media data generate word vectors that have high feature dimensions. This implies that text representation vectors formed contain irrelevant and redundant features that affect the performance of sentiment analysis models. Further, these approaches focus on word frequencies, inter-word distances, and word relationships ignoring other aspects of words such as the sentiment orientation of the words thus yielding poor text and sentiment classification results (Samira et al., 2018). In Natural Language Processing, the presence of a sentiment word in a text influences the sentiment classification of the entire text and can also point to a specific section of a text where the opinion can be found. Therefore, there is a need for alternative text representation approaches applied in social media data sentiment analysis that can generate word vectors with low feature dimensionality and rich in sentiment.

1.3 Research Objectives

1.3.1 General Objective

The main objective of this research work was to develop and evaluate a sentiment lexicon-augmented text representation model for social media sentiment classification.

1.3.2 Specific Objectives

To achieve the main objective, the research was guided by the following specific objectives:

1. To analyze existing text representation techniques for sentiment analysis to improve them for sentence-level sentiment analysis;
2. To develop a social media text representation model based on sentiment lexicon-enhanced text representation algorithms;
3. To implement the sentiment lexicon augmented text representation model in different machine learning classifiers;
4. To validate the sentiment lexicon-augmented text representation model for sentiment analysis.

1.4 Research Questions

To achieve the stated objectives, the study sought to answer the following research questions;

i. What are the strengths and weaknesses of the existing text representation algorithms in social media text sentiment analysis?
ii. How do you develop a social media text representation model based on a sentiment lexicon-enhanced text representation algorithm?
iii. How do you integrate the sentiment lexicon-based text representation algorithm with different machine learning classifiers for an effective sentence-level sentiment analysis model?
iv. How does the performance of the developed classification model compare with the performance of state-of-the-art classification models?

1.5 Justification

The advancement of web technology and social media platforms has led to a proliferation of user-generated content in the form of short sentences commonly referred to as posts. Users have used these platforms to express their opinions on objects, events, people, or other users. Sentiment analysis of such text can be used by business people to analyze their customers' feedback and improve their products. Government agencies can also analyze social media texts to get citizens' opinions on governance issues. Generally, the proliferation of social media has created incomparable opportunities for social media users on one hand to publicly voice their opinions and on the other, the urgency to gain a real-time understanding of citizens, customers, and relevant others' opinions has grown (Kaushik et al., 2015)

Despite the potential benefits, how to automatically analyze tons of this text data, efficiently, has become a concern of artificial intelligence researchers (Mingyong & Yang, 2012). These challenges range from the identification of feature extraction and selection algorithms to text classification algorithms. Text data analysts and researchers have been developing feature extraction and selection algorithms to be used in sentiment analysis. As highlighted in section 1.2, the conventional algorithms based on the Term Frequency-Inverse Document Frequency (TF-IDF) algorithm and the N-gram models suffer high feature dimensionality and they are highly applied at the document level of sentiment analysis. This is attributed to the fact that features are extracted from the entire document whereas the opinion may be found in a specific part of the document. A need thus arises for a better feature extraction algorithm that can identify a specific part of a sentence where the opinion is and thus improve sentence-level sentiment analysis. This thesis proposed and developed a lexicon-based approach to identify a specific N-gram where opinion can easily be identified for a sentence-level sentiment analysis model.

There are several ways of improving existing algorithms which include combining two or more algorithms, elimination of redundant steps, or addition of more steps to the algorithm. Further, artificial intelligent classification models are highly dependent on the input features. This study mainly focused on feature extraction and selection of

existing algorithms and their application in machine text classification models. The aim was to develop a realistic, scalable, easy, and reusable approach that could identify a specific part of a subjective sentence, identify features at that part, and use the features as input into machine learning classifiers. This simplifies social media sentiment analysis.

1.6 Scope of the Study

Sentiment analysis focuses on the whole document, sentences, or the opinion holder. Social media is made of short statements in the form of text posts therefore this research focused on sentence-level sentiment analysis. Twitter Social media platform was used in the experiments. The Sentiment analysis research was advanced in this research work by focusing on the feature selection and classification steps of the entire sentiment analysis process. Review-based tweets are used as sentence texts to train and test the proposed model.

1.7 Thesis Organization

This thesis is divided into five chapters. Chapter 1 is the introduction chapter of this thesis which provides the background of the study. In the background, the concepts of sentiment analysis, feature extraction, and selection are discussed chronologically. Chapter One, further describes the problem statement, research objectives, justification, and thesis organization. The next four chapters are organized as follows.

Chapter 2 starts with the theoretical foundations of the research which includes an introduction to sentiment analysis and its process. A description of sentiment analysis trends, feature extraction algorithms applicable to sentiment analysis, and feature selection algorithms are discussed. This is then followed by a critical review of various machine learning classifiers applicable to sentiment analysis and the entire text classification problems. An empirical review of existing work done in the area of sentiment analysis is then represented. This is followed by the presentation of the proposed model, process, algorithms, and an illustration. Finally, performance metrics of evaluating classifiers are discussed.

Chapter 3 presents the research methodology used. Experiments carried out in this research are also described. This chapter starts by describing the text data collection process, datasets used, sentiment classification steps, and data preprocessing. Finally, a detailed description of the proposed model is presented.

Chapter 4 is a presentation of the results obtained from the experiments. The results are discussed to evaluate the performance of the proposed model in comparison with existing models.

Chapter 5 starts with a description of Knowledge contributions, which is then followed by a conclusion and finally describes areas for future work.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter discusses the trends in sentiment analysis and opinion mining. A review of studies done in the area of sentiment analysis is presented in this chapter. The first section discusses the general concepts of opinion mining and sentiment analysis. The subsequent sections discuss and critique research done on specific areas related to this research work. They include trends in text representation techniques, supervised and unsupervised machine learning classifiers, web 2.0, and micro-blogging. A review of statistical analysis and optimization techniques is also done. Performance evaluation methods applicable to classification models are also discussed later in the chapter.

2.2 The Concept of Text Classification

The recent expansion of web technologies encourages users to contribute and express themselves through various avenues including blogs, videos, and audio through social media sites. All these platforms provide a huge amount of valuable user-generated text and visual data that researchers are interested in mining for insightful knowledge and information. Some of these text present users' opinions on objects or subjects.

Text Classification (TC) is the activity of automatically building, using machine learning (ML) techniques, automatic text classifiers in the form of computer programs that are capable of labeling natural language texts from a domain D into thematic classes from a predefined set of classes $C = \{c_1, \ldots, c_n\}$ (Sebastian, 2002). The number of classes is dependent on the opinions to be considered. The construction of an automatic text classifier relies on the existence of a collection of documents or sentences in a corpus $\Omega = \{d_1, \ldots, d_n\}$. The documents are then classified under C. A general inductive process referred to as the learner automatically builds a classifier for C by learning the characteristics of C from a training set $Tr = \{d1, \ldots, dt\}$ of documents. Once a classifier has been built, its effectiveness which is its capability to take the right categorization decisions may be tested by applying it to the test set $Te = \Omega - Tr$ and

checking the degree of correspondence between the decisions of the classifier and those encoded in the corpus. This is called a supervised learning activity since learning is "supervised" by the information on the membership of training documents in categories. The construction of a text classifier may be seen as consisting of essentially two phases. The first phase is the document indexing phase, which involves the creation of internal representations for documents. This phase typically consists of term selection, consisting of the selection, from the set that contains all the terms that occur in the documents of the training set, of the subset of terms that, when used as dimensions for document representation, are expected to yield the best effectiveness. The phase also includes term weighting, in which, for every term *tk* selected and for every document *dj*, a weight $0 \leq wkj \leq 1$ is computed which represents, loosely speaking, how much term *tk* contributes to the discriminative semantics of document *dj*. The terms include words, part-of-speech tags, and any other language syntax that can be used to represent a document or a sentence. In social media, there are several syntaxes used including nonlinguistic symbols such as emoticons and other symbols.

The second phase is the classifier learning phase, which involves the creation of a classifier by learning from the internal representations of the training documents. The representation of the document is in the form of a vector which can be an input to the machine learning classifiers (Mutinda et al., 2021). The main application tasks of Text Classification in Natural Language Processing are Sentiment Analysis and Opinion Mining.

2.2.1 Opinion Mining

Opinion mining is defined as a sub-discipline of computational linguistics that focuses on analyzing people's opinions (Bhadane et al., 2015). It is an advanced level of sentiment analysis in which a deeper analysis is done to understand the drivers behind the presented opinion or sentiment. For instance, analyzing the reason why the person liked the food is opinion mining. Opinions are easy to understand for human beings, but it is not that easy for a computer to have the same level of understanding. (Devika et al., 2016).

The concept of opinion mining, as discussed by Liu (2010), consists of the following items: First, the opinion targets which are the entities and their features or aspects. For instance, an opinion may be targeted on a mobile phone or a specific aspect of the mobile phone like the camera. The second item is the sentiments which are either positive or negative. The third item is the opinion holder(s) which includes the people who hold the opinions. Lastly is the time when the opinions were expressed.

2.2.2 Sentiment Analysis

Sentiment analysis is the process of determining whether a text is subjective or objective. A subjective text contains an opinion (sentiment) of the writer about an object or subject while an objective text does not. For example, a statement: "I liked the meal", is subjective while a statement: "I went to the hotel to take lunch", is objective. Sentiment analysis also involves determining the polarity of the subjective text which may be positive or negative. Further in sentiment analysis, the polarity strength which may be weakly positive, mildly positive, or strongly positive is determined. Therefore, the main objective of sentiment analysis is to identify the mood, opinion, or sentiment of the writer in the text (Kaushik et al., 2015) (Bhadane et al., 2015).

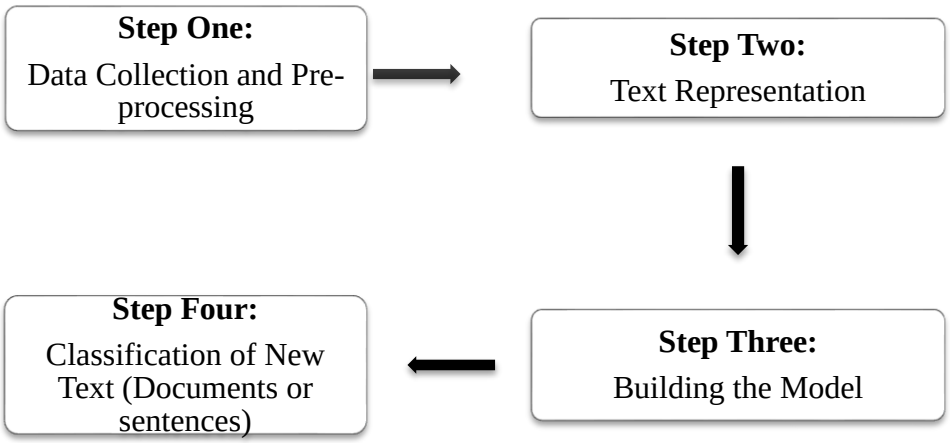
Sentiment analysis is a text classification task in which the document or the sentence is categorized into a specific class of opinion (Appel et al., 2015) (Devika et al., 2016). Sentiment analysis as a subarea of text classification focuses on the whole document, sentences, or the opinion holder. Since social media is made of short statements this research will focus on sentence-level sentiment analysis.

Sentiment analysis is divided into four steps. The first step is the object identification step. This step involves discovering what the object is, about which an opinion has been provided. An object can be a product, service, a person, an occurrence, or an event. The second step is aspect extraction and synonym grouping. At this stage, the specific feature of the object is identified to which the opinion is directed. The second step is the opinion orientation classification. The objective of this step is to find out whether there is an opinion on a feature in a given sentence or not. If there is one, is it positive, negative or neutral? (Liu, 2010). One of the key issues at this step is the

identification of features that can be used in sentiment analysis. The problem is that there are seemingly an unlimited number of expressions that people use to express opinions, and in different domains, they can be significantly different. Even in the same domain, the same word may indicate different opinions in different contexts. The fourth step is integration. At this step, the main objective is to discover the classes where a given sentence or document as an input to a machine classifier is correctly classified. To achieve this, there is a need to integrate the discussed steps.

2.3 Process of Building Sentiment Analysis Models

The main objective of a sentiment analysis model is to determine automatically the opinion class of unlabeled text or a set of texts such as sentences or documents. The SA model could be built through supervised or unsupervised learning. Supervised learning also called learning from example is the one employed in this study. In supervised learning a set of labeled documents or sentences is used to train a model which is then used to classify new unlabeled ones (Appel et al., 2015). The supervised Sentiment Analys



is process includes four steps as presented in Figure 2.1.

Figure 2.1: Sentiment Analysis Process

The first step in SA is data collection and preprocessing. This step involves the collection, cleaning, and organization of the documents or sentences. In some instances, the cleaned data is subdivided into training and testing sets. The training set

is used to create the model while the testing set is for evaluating the constructed model. Pre-processing is the first step in sentiment analysis after text data collection. In this phase, the collected text either the documents or the sentences is subjected to certain refinement which includes removal of stop-words and punctuation marks, removal of HTML tags, sentence lemmatization, word tokenization, and changing the text into lowercase among others. This process is also called text data cleaning since it focuses on removing irrelevant information and junk from the collected text (Appel et al., 2015).

Stop words are words that frequently occur in a document and do not represent any significant content but are used to help in the construction of sentences. They include articles, prepositions, conjunctions, and pronouns (Bhadane et al., 2015). There are several strategies for removing stop words. The most common is sorting the words of the text by occurrence frequency and then removing the ones with high frequency as stop words (Baharudin & Khan, 2011). Stop words can also be taken as words with very few characters such as one, two, or three. Very short words are treated as articles or conjunctions and are removed as stop words. However, some short words may not be articles for example, “do”, and “not” which is a verb and may affect opinion orientations of other words (Mutinda et al., 2021). The stop words removal strategy applied therefore is dependent on the text classification task.

Tokenization is the process of dividing text into words or terms called tokens (Chandrasekaran et al., 2021). A token is thus a word or a character representing a section of the whole text. After tokenization, each token is converted into a standard or basic form of the word. This process is called stemming or lemmatization. Stemming reduces a word to its basic form. A word has various syntactical forms depending on the context in which it is used. For example, in English, a noun can be written in plural or singular form, similarly, a verb can be written in various forms such as past tense, present tense, or continuous present tense. Through stemming, all these syntactic variations come from the same “word-root” form called the stem or lemma. Lemmatization also referred to as stemming is therefore the process of reducing words into their stems after removing the word prefixes and suffixes. For example, ‘reading’, and ‘reads’ are reduced to the word ‘read’ after stemming. The

main objective of stemming is to reduce the text by reducing related words to be the same.

The second step is text representation. Text representation is the process of converting the original textual data into a numerical form that can be understood by machine classifiers. The output of this phase is data mining-ready data which represents the input text. This is a complex process through which each document, sentence, or word is represented by a set of numbers. It is also called the vectorization method since most of the words, sentences, or words are represented by a vector. This is a very critical phase since the machine learner or classifier uses the vector to classify the input text (Deisy, 2010).

Various text representation techniques are used to transform a document from its irregular and implicitly unstructured representation into an explicitly structured representation. These techniques are also called weighting techniques or word vectorization techniques. They can be categorized into natural language processing-based techniques and deep learning-based techniques. The techniques are used to convert the entire input text into its vector representation. However, a short document may have a large number of words, phrases, sentences, typographical elements, and layout artifacts posing a need to identify a subset of the document features to represent the whole document (Han & Kwangmi, 2017). Text representation produces a set of features in vector form referred to as the text presentational model of a document or text. The most commonly used features to represent documents are; words, characters, phrases (co-occurrence of words), and Part of Speech Tags (POS) among others.

The third step is building the SA model. This phase involves building a text classification model that can predict the sentiment class of any input text. It involves the application of Machine Learning (ML) algorithms to come up with the model. Machine Learning is a branch of artificial intelligence that deals with the exploration and construction of models from data. In ML data is used to train computers to handle similar problems through analysis of the data and identification of patterns or relationships within the data (Srivastava, 2018). Machine Learning can be classified into supervised learning or unsupervised learning. In supervised learning, the machine

learns from labeled data by establishing a model of categorizing the data into classes. Once learned from the example, the model can then be used to classify new unlabeled data. In unsupervised learning, the machine learns from unlabeled data by identifying the relationships within the data.

Sentiment analysis and text classification models are mostly built through supervised machine learning algorithms. Thus building a sentiment analysis model involves using supervised machine learning algorithms to construct models that can classify an input text into two or more sentiment classes. Therefore in building sentiment analysis models training data (which acts as the usual human experience for the machine to learn from) is used to build the model. Once the machine has been trained, another set of data called a testing set is used to evaluate the performance of the model. During testing, the model is applied to the documents from the test set and their actual class labels compared to the labels predicted by the constructed model. Generally, this phase of building the model can be divided into two steps; the training and testing phases.

The fourth step involves the classification of new documents. In this step, the model trained and tested in step three is used to classify new documents (unlabeled). This is the final step where a new document is subjected to the model and its sentiment class predicted.

2.4 Sentiment Analysis Trends

Sentiment Analysis started in the 1980s. This is according to Appel who argues that no remarkable work applicable to SA was done until the 1980s (Appel et al., 2015). It started with classifying sentences as subjective or objective. Several researchers used subjective and objective sentences as indicators of sentiment they included Banfield in 1982 and Winograd in 1983 according to (Appel et al., 2015).

The concept of Natural Language processing started in the 1990s with the development of WordNet by Miller et al. (Miller et al., 1990). WordNet is a database of English words, their meaning, and synonyms. This was closely followed by the concept of Part-of-Speech tagging (Brill, 1994). This became an important aspect in identifying parts of sentences and extracting meaning from the sentences. The semantic orientation

of adjectives which carry most semantic orientation of phrases was also done. (Hatzivassiloglou & W., 2000).

Sentence classification was later started by Hearst in an agent 'Text-Based Intelligent System' that focused on the ability of an agent to classify a sentence according to its sentiment orientation (Hearst, 1992). The researcher focused on the internal structure of a sentence. Later the concept of extraction of subjective words was articulated (Wiebe J., 1994).

Esuli and Sebastiani, (2006) improved WordNet, by assigning each set of synonyms for groups of English words three sentiment scores: positivity, negativity, or objectivity. They developed a new dictionary – Senti WordNet. The orientation of a sentence depends on the orientation of the terms used in the sentence. A term in a sentence can be replaced by more words of the same meaning thus senti Word Net became a very important addition in SA.

Machine Learning Techniques have been used to solve the SA problem by several researchers (Cambria et al., 2013). They include both Supervised Learning Techniques and Unsupervised Learning Techniques (Kumar & Sebastian, 2012) and (Pang & Lee, 2008). Some Unsupervised Learning techniques have been very successful as well, as it is unsupervised techniques based on the PMI-IR algorithm that is used to estimate the semantic orientation of a phrase by measuring the similarity of pairs of words or phrases (Turney, 2002).

Alternative methods have been proposed, like the Bootstrapping Method for Building Subjectivity Lexicons for Languages with Scarce Resources, the techniques for generating a quality lexicon (Taboada et al., 2011) the recognition of contextual polarity (Wilson et al., 2009), and the granularity of subjective sentences based on adjective orientation (Hatzivassiloglou & W., 2000).

When one attempts to establish the orientation of the sentiment in a document, one is faced with the need to summarise somehow all the contents (Appel et al., 2015). Similarly, for sentence-level text classification, the sentence terms need to be analyzed. (Suanmali et al., 2009) proposed a fuzzy logic-based method for improving the

summarization of text, while Liu (Liu B., 2010) stressed one more time the important aspect of SA/OM of determining subjectivity by differentiating between objective and subjective sentences. This is based on document classification where the orientation of the sentences is important. At sentence level classification the terms (words) in the sentences become the focus. Some researchers have also looked into the possibility of applying semi-supervised learning methods like the one used for opinion summarization and classification for online product reviews (Dalal & Zaveri, 2013)

2.5 Applications of Sentence Level Sentiment Analysis

Sentence-level sentiment analysis has several applications in society. Areas, where this task is applied, include information retrieval, event detection, security preparedness, e-commerce, and biometrics.

Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers). This task involves searching for information from a collection of data. Sentence classification is very important in this area since it is the basis of document classification. To extract meaning from social media data, sentence-level text classification is one of the solutions.

Event detection is a core Natural Language Processing (NLP) task that focuses on the automatic identification and classification of various event types in text. Event detection is applied in question-answering systems on the websites. Sentence classification is the basis of the automatic identification of events that are relevant to a certain question since any Question is related to a certain event.

Sentence-level text classification is also applied by security agents. Massive online public discourse data has the potential to predict crowd behavior using text analysis methods. One of those methods referred to by Lam is Sentence-level text classification. Through mining public discourse data from social media, one can predict security-related activities. Further, false information can also be identified through sentence-level sentiment analysis. Here the information is classified as false or true. This can

also lead to information being flagged or deleted before spreading to social media audiences.

E-commerce has been improved significantly by text analytics and mining. E-commerce vendors use social media for marketing, collecting customer feedback, and identifying potential customers through social media networks. One of the most current technologies being used is the voice of the customer. Voice of the customer (VOC) is a term used in business and Information Technology to describe the in-depth process of capturing customer's expectations, preferences, and aversions (Wikipedia). The purpose of this technology is to collect customer needs and wants and analyze them depending on the alternatives being offered. Text analytics are used in this technique specifically the frequency-based sentence classification where the terms customer uses to describe his/her satisfaction levels are considered. In this case, the customer reviews are analyzed to provide insights that can be used to improve the customer experience and satisfaction. The customers give their feedback through social media platforms such as Twitter and Facebook.

Sentence-level classification can also be applied in identification and authentication systems. Voice biometrics is one technique of identifying a user by voice pattern. Voice biometrics uses the pitch, tone, and rhythm of speech. In sentence-level classification the pattern of words can be used to identify a user. A person tends to have a unique pattern of speech and consequently of written speech. Sentence text classification can associate a certain pattern of words with a certain user.

2.6 Machine Learning Algorithms for Text Representation

Machines cannot understand and comprehend human language and the actual meaning that underlies it. Therefore, machines require parameters provided as input that enable them to model the meaning. In this endeavor, accurate and sufficient document and sentence text representation is crucial (Makrehchi & Kamkarhaghighi, 2017).

There are several techniques used for text representation in Sentiment Analysis. These techniques include Bag of Words, TF-IDF, word embedding, and NLP (Natural Language Processing) based features such as part of speech tags (POS) (Chug et al.,

2019). Most of the techniques are used to generate the word vectors that fully represent the documents or sentences and are thus used as input to machine learning techniques (Ankit, 2018) (Mozetic et al., 2016). These feature extraction techniques are used to generate word vectors to represent the text.

2.6.1 Term Frequency-Inverse Document Frequency (TF- IDF)

Term Frequency–Inverse Document Frequency (TF-IDF) is one of the most used document representation algorithms in Sentiment analysis (Tian & Yanmei, 2011). TF-IDF is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. TF-IDF determines the relative frequency of words in a specific document compared to the inverse proportion of that word over the entire document collection (Robertson, 2004). The Term Frequency (TF) is calculated as the term count in the given document or sentence. The Frequency of Sentiment Terms in a sentence is very low mostly one. For the term t within the particular document d, its term frequency is defined as follows:

$$TF(t) = \frac{N(t_i,d)}{S} \tag{2.1}$$

Where, $N(t_i, d)$ is the frequency of term t in document d and S is the total number of words in document d. For social media, Document D is a single review or a post. Since social media posts are normally very short, the value of S is small.

The Inverse Document Frequency is calculated as the ratio of the number of documents in the corpus to the total number of documents with the term t in it in the corpus. If there are N documents in the corpus and $N(t_i)$ documents have term t then the IDF for the term t is defined as follows;

$$IDF = \log(\frac{N}{N(t_i)}) \tag{2.2}$$

Where IDF is the Inverse Document Frequency for term t_i.
Thus, the TF-IDF weight is calculated as follows;

$$\mathrm{TF} - \mathrm{IDF} = \mathrm{TF(t)} \times \mathrm{IDF} = \frac{N(t_i,d)}{S} X \log(\frac{N}{N(t_i)}) \tag{2.3}$$

The importance increases proportionally to the number of times a word appears in the document but is normalized by the frequency of the word in the corpus (Robertson, 2004). In sentence-level sentiment classification, each sentence is considered a document. TF-IDF has several advantages when used in sentiment analysis. According to Robertson (Robertson, 2004), TF-IDF is an efficient and simple algorithm for matching words in a query to documents that are relevant to that query. Deisy (2010) agrees that Term Frequency (TF) is the simplest measure to weigh each term in a text. However, at the sentence level, the frequency of the word is mostly very low if not one hence making this technique inefficient in Sentence level SA. However, the drawback of TF is the difficulty in finding the optimal thresholds of features. The vector produced has very many variables and thus to establish the optimal ones is a challenge. While TF considers term occurrence within a text, Inverse Document Frequency (IDF) is concerned with the term occurrence across a collection of texts. The intuitive meaning of IDF is that terms, which rarely occur in a collection of texts, are valuable which improves precision. Thus, the combination of TF and IDF improves recall and precision respectively that gives better performance. Although TF-IDF improves the precision in term weighting, its accuracy should be improved for best results in text categorization (Mingyong & Yang, 2012). Since TF-IDF is about frequency it does not consider information value in the words and also neglects relationships between words.

2.6.2 N-Grams

N-grams are also used in the representation of text in text classification and sentiment analysis. Different researchers have different definitions of *N*-grams. *N*-gram is a statistical language model (LM) where a document or a sentence is broken down into a sequence of words w_i (w_1, w_2, …..,w_n). *N*-gram is the most used language model (Noaman H. M., 2018) which makes a Markov assumption and defines the context Φ (W_{i-1}) as;

$$\Phi\,(W_{i-1}) = w_{i-n+1}, w_{i-n+2}, \ldots, w_{i-1} = h \qquad (2.4)$$

For, $N = 1$ the context does not exist hence it's the normal bag of words

N=2 the context becomes $\Phi\ (W_{i-1}) = w_{i-1}, w_i$ Thus two words are considered

N=3 the context becomes $\Phi\ (W_{i-1}) = w_{i-2}, w_{i-1}, w_i$. Thus, three words are considered.

From the formal definition, an *N*-gram is thus a textual sequence containing *N* adjacent 'textual units' from a particular sentence or document. A 'textual unit' can be identified as a character, word, or phrase depending on the context of interest from which a vector representation of the *N*-grams is formed. In this work, we identify the *N*-gram at the word level.

Each of the *N*-grams is a coordinate in a vector that represents the text under study. The frequency, occurrence, or any other metric of this n-gram in the text becomes the value of this coordinate (Brindha et al., 2016). The simplest n-gram is the unigram, where $n = 1$, which is the normal "bag-of-words" (BOW) representation. Generating the vector from a huge text dataset can be challenging. *N*-gram models are widely used in NLP tasks since they are simple and effective (Aisopos et al., 2016). However, in *N*-grams, each sentence is converted into a bag of *N*-grams and represented as a vector of occurrence frequency without taking into consideration the information encapsulated in the n-grams of the original text. This leads to so many irrelevant and redundant attributes in the vector. The sliding window of the n-gram also makes the variables more and thus some become less relevant.

In *N*-gram, each sentence or tweet is converted into a bag of n-grams. All n-grams are extracted from the sentences or documents and their frequencies are noted. In sentiment analysis, a vector is formed containing the frequencies of the grams and the sentiment class. In the n-gram vector, each sentence is converted into overlapping character n-grams by running a predefined window of size n to construct its character n-gram vector representation. Typically, n is a fixed number, highly dependent on the particular corpus of documents and the queries made against that corpus. Each of the n-grams is a coordinate in a vector that represents the text under study and the frequency that this n-gram appears in the text can be the number of this coordinate (Bouras & Tsogkas, 2016). If we consider two sentences; sentence 1: *Loved this place* and sentence 2: *the meal was great and enjoyable*

N-grams from the sentences can be represented as;

For $N = 1$; loved, this, place, the meal, was, great, and, enjoyable

$N = 2$; loved_this, this_place, the_meal, meal_was, was_great, great_and, and_enjoyable

N-gram models are widely used in NLP tasks since they are simple and effective (Aisopos et al., 2016). However, in n-grams, each sentence is converted into a bag of n-grams and represented as a vector of occurrence frequency without taking into consideration the information encapsulated in the n-grams of the original text. This leads to so many irrelevant and redundant attributes in the vector. The sliding of the n-gram makes the variables more and even less relevant. The simplest n-gram is the unigram, where $N = 1$, which is the same as the normal "bag-of-words" (BOW) representation.

Researchers have proposed improvements in text feature representation algorithms TF-IDF and *N*-Grams. They include (Mingyong & Yang, 2012) and (Tian & Yanmei, 2011) who proposed an introduction of term distribution weight in the frequency. Tian and Yanmei investigated other statistical characteristics of terms and found an important discriminator: term distribution. Further, they found that, in a single document, a term with higher frequency and close to hypo-dispersion distribution usually contains much semantic information and should be given higher weight. On the other hand, in a document collection, the term with higher frequency and hypo-dispersion distribution normally contains less information. They used Chi-square to show the distribution of the terms. The experiments they performed yielded better SA results. This improvement is substantial for it considered the word distribution in the document but did not consider the sentence level classification a gap this research seeks to address.

Researchers in Tian and Yanmei (2011) proposed a new improvement to the conventional TF-IDF weighting algorithm by coming up with a term distribution algorithm. In their work, they argued that the term that is evenly distributed in the entire document should be given more weight than the term that appears more in a

section of the document. However, stop words are distributed all over the document and don't have any significance. For sentence-level classification, this may not be applicable since the frequency of the term may be not more than twice in the sentence.

In their research, Baharudin and Khan (2011) suggested that sentence structure and contextual information in the review are important for sentiment orientation and classification. In the proposal, the classification was in three classes negative, positive, and neutral. To classify the sentences, a rule-based approach was employed. Each term in the sentence was assigned a sentiment score from SentiWordNet. The overall classification of the sentence is the sum of the individual sentiment scores of the terms in the sentence. While I agree with the proposal, one of the limitations of the approach is that words can be of the same orientation but negating one another thus giving the wrong classification.

The most recent research work reviewed is by Chug et al. (2019) where TF-IDF and N-Grams methods of feature extraction were accessed on six classification algorithms. The TF-IDF algorithm performed better than N-gram. To advance this work they proposed an investigation on word polarity scores and word embedding in feature selection.

2.6.3 Bag of Words

Bag-of-Words (BOW) refers to the natural language processing method that handles a text document as a collection of its words. A text is represented as a vector of word weight (Passalis & Tefas, 2018). The words in a text are separated and considered as individual features in a vector. BOW text representation is computationally cheaper and also simpler compared to other methods (Qader et al., 2019). However, this representation suffers from two main limitations: (1) it breaks terms into their constituent words, e.g. it breaks 'Text Classification' into the words 'Text' and 'Classification'; (2) it treats synonymous words as independent features, e.g. 'Classification' and 'Categorization' are considered as two independent words with no semantic association. Mathematically the *N*-gram vector is represented by the chain rule of probability and thus we write the probability of any sequence as;

$$P(w_1\, w_2, \ldots, w_i) = \prod_i P(w_i|w_1, \ldots, w_{i-1}) \quad (2.5)$$

An *N*-gram model approximates this probability by assuming that the only words relevant to predicting $P(w_i|w_1, \ldots, w_{i-1})$ are the previous *n-1* words; that is, it assumes the Markov *n*-gram independence assumption:

$$P(w_i|w_1, \ldots, w_{i-1}) = P(w_i|w_{i-n+1}, \ldots, w_{i-1}) \quad (2.6)$$

This is also related to statistical patterns of human word usage that can be used to figure out what people mean.

N-gram models thus can be improved by considering the information contained in the grams and making the sliding window 'immovable'. All the methods discussed in the review can be improved by incorporating feature selection approaches. Selection of a subset of relevant features would reduce noisy, irrelevant, and redundant features (Miao & Niu, 2016), (Milos et al., 2017). Researchers have also attempted to address the shortcomings of N-gram models. Specifically for *N* = 1, the BOW has been improved by representing text as concepts rather than words, using an approach known as Bag-of-Concepts (BOC).

2.6.4 Part of Speech Tagging

Another technique used in Feature selection is the use of Part-of-speech (POS) tags. POS tagging is the assignment of each word in a sentence the part of speech that it assumes in that sentence. There are several parts of speech which include nouns, verbs, adverbs, adjectives pronouns, and others. Some tags are very important in showing the subjectivity of a sentence.

There are several tools used in POS tagging. The tools are based on supervised methods or unsupervised methods. POS tagging tools include POS tagging online, Stanford NLP tagger, Open Xerox tagger, NLP DotNet tagger, and NL toolkit tagger among others. POS tagging online and Stanford NLP tagger will be used in this research since they have several advantages over the others. POS tagging is considered a fundamental part of Natural Language Processing, which aims to computationally

determine a POS tag for a token in a text context. POS tagger is a useful preprocessing tool in many NLP applications such as information extraction and information retrieval (Salvetti et al., 2014). As social media becomes popular, Twitter POS tagging poses additional challenges to existing tagging models due to the conversational expression style and the free spelling style of the text. (Ji et al., 2012)

In the subjectivity domain, the presence of adverbs and adjectives in a text affects the subjectivity or objectivity of the text (Koto & Adriani, 2015). The experiments done by Adrian and Koto (2015) affirm that the POS sequence can be utilized for performing Sentiment Analysis over Twitter. However, they did not consider the effect of the sequence on the sentiment term an area which this research work will investigate.

Some POS tags are more commonly used in showing the subjectivity of a sentence than others for instance; people tend to use adjectives or adverbs rather than nouns in uttering their opinions, emotions, or beliefs (Koto & Adriani, 2015). POS tagging is considered a fundamental part of natural language processing, which aims to computationally determine a POS tag for a token in a text context. POS tagger is a useful preprocessing tool in many NLP applications such as information extraction and information retrieval (Salvetti et al., 2014). As social media becomes popular, Twitter POS tagging poses additional challenges to existing tagging models due to the conversational expression style and the free spelling style of the text (Ji et al., 2012). POS tagging tools used are based on supervised or unsupervised learning methods. They include POS tagging online, Stanford NLP tagger, Open Xerox tagger, NLP DotNet tagger, and NL Toolkit Tagger. Stanford NLP Tagger was used in this research since they have several advantages over the others.

In their survey of sentiment analysis techniques up to the year 2015, Appel, Chiclana, and Carter (Appel et al. 2015), evaluated both lexicon-based methods and machine learning methods. They indicated that POS tags are good indicators of subjectivity and sentiment. They also challenged researchers to consider combining machine learning methods and opinion lexicon methods to improve SA accuracy. Consequently, (Koto & Adriani, 2015) conducted a study on POS tags sequence and how they affect the subjectivity and objectivity of sentences. In their work, they considered a combination

of two POS tags, three POS tags, and Five POS tags. They found out that some tags are used to show emotions while others like nouns are used to portray facts. They affirmed that the POS sequence can be utilized for performing Sentiment Analysis over Twitter. This is an important insight in this work since the concept of POS tag is utilized in the work. The limitation of Adrian and Koto is that they classified the sentiments into subjective and objective sentences (Tweets). Thus any work on the detailed subjectivity (extent of negativity or positivity) advances their work.

Bouazizi and Ohtsuki in their research came up with a multiclass SA in Twitter using seven classes (Bouazizi & Ohtsuki, 2017). They developed an open-source classification application called SENTA. They employed the proposal by Baharudin and Khan (2011) of sentence structure. Here they used a pattern-based approach for the Multiclass SA. The POS pattern of a sentence was used to give the class of the sentence. POS tags in a sentence were classified into three categories; POS tags containing emotional content (EI), POS tags containing non–emotional words (CI), and POS tags whose grammatical function is important (GFI). The sequence of the categories was the basis of the developed SENTA. However, they did not consider the sentiment orientation of the terms thus a weakness in their approach. In their conclusion, they suggested that the position of a term within a text can affect how much the term makes a difference in the overall sentiment of the text.

2.6.5 Sentiment Lexicon-Based Text Representation Techniques

Sentiment lexicon-based techniques involve the extraction of emotional value or the use of an emotional dictionary in a text to determine the sentiment class of the text. Sentiment models based on sentiment lexicons determine the sentiment class of a whole text, a sentence, or a group of sentences based on the semantic orientation of single words. The Semantic orientation of a word can be positive, negative, or neutral (Kukkar et al., 2023). The semantic orientation of the words is contained in a dictionary of lexicons which is built either manually, mechanically, or both. The most common lexicon dictionary is the WorldNet dictionary (Liu, 2010). Mainly adjectives and adverbs carry semantic information of the writer hence they are used to determine the text's semantic orientation. Once the sentiment orientation of each word in the text

is determined, an overall sentiment score of the text is calculated by weighting the number of positive and negative words. Generally, the sentiment score is given as;

$$Sentiment\ Score = \frac{Number\ of\ Positive\ Words - Number\ of\ Negative\ Words}{Total\ number\ of\ words\ in\ the\ text} \quad (2.7)$$

Liu and Tsou (2010) used WordNet seed words to create a vocabulary of negative and positive sentiment terms, and then classified sentences using the dictionary. Baharudin & Khan (2011) argued that sentence structure and context are important aspects in the sentiment classification and orientation of text. They therefore proposed a model in which each word in a text was given a sentiment score from the SentiWordNet dictionary and then the cumulative sum of the individual sentiment scores for each of the words in the text gives the overall sentiment class of the text. Although the proposed method is interesting, one of its limitations is that words with the same orientation but opposing meanings may be classified as having incorrect lexicon labels hence the wrong classification of the text. Bermingham and Smeaton (2010) in their study established that sentiment analysis in small corpus documents or shorter text is easier than in larger texts or corpus. Here we note that sentiment lexicon-based text representation techniques are suitable for shorter texts such as social media posts and reviews. Most research has been done on how to develop and build sentiment lexicons rather than how to apply the existing lexicons for text classification tasks such as sentiment analysis. For instance, Velikovich et al. (2010) proposed and investigated a method for building a sentiment lexicon from the entire internet. The resultant lexicon had significantly covered more words than current dictionaries. Lexicon-based techniques of text representation are straightforward to apply. However, the main focus of such techniques is on the sentiment dictionary thus overlooking other text characteristics such as word position and relationships.

2.6.6 Word Embedding Text Representation Techniques

Recently, word embeddings-based text representation techniques have played an important role in natural language processing tasks including sentiment analysis (Araque et al., 2017). According to Mikolov et al. (2013), research in word embedding text representation gained momentum in 2013. This is because they are better than the

normal bag of words representation, since they cater for synonyms and produce vectors with lower dimensionality than the bag of words (Rezaeinia et al., 2019) (Mikolov et al., 2013). Further, pre-trained word embedding vectors are used as inputs of machine learning classifiers in sentiment analysis research since they are more accurate and compatible with deep learning neural networks (Giatsoglou et al., 2017). However, pre-trained word embeddings ignore the sentiment orientation of words and their context, hence affecting sentiment classification accuracy (Rezaeinia et al., 2019) (Araque et al., 2017). This is because they use word distances and synonyms to calculate word vectors.

The main word embeddings algorithms are Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014), and FastText (Bojanowski, 2017) (Chandrasekaran, et al., 2021), which are used to convert words to vectors.

Word2Vec is a two-layer neural network word embedding model that is trained to take a large corpus of words as an input and produce a vector space comprising unique word vectors. The word vectors are such that words that share common contexts in the corpus have almost the same vectors. Word2Vec is a probabilistic model used to predict the probability of a word given the context and vice versa. Where; o is the target word and c is the context word(s). V_c and U_o are the word vectors for c and o respectively.

There are two architectures of Word2Vec models, the Continuous Bag-of-Words (CBOW) model and the Skip-Gram model. In the CBOW model, word embeddings are generated by predicting a word given its surrounding context words. The model applies the bag-of-words assumption that the order of the neighboring words does not affect the embedding (Mikolov, 2013). Visualization of the Word2Vec architectures is shown in Figure 2.2.

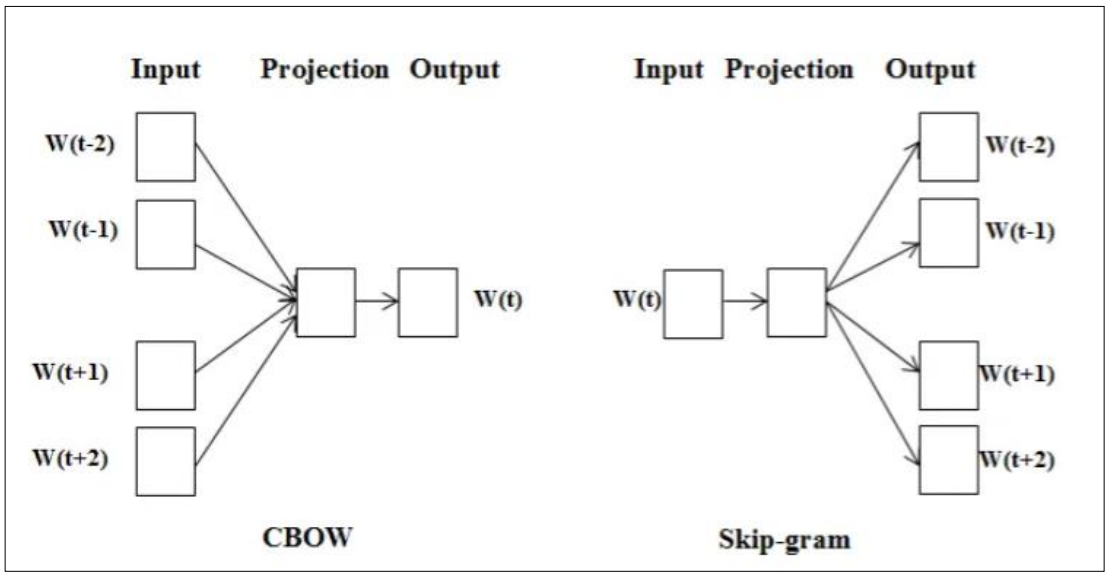


Figure 2.2: Word2Vec Model Architectures

The glove is an unsupervised text representation method that stands for Global Vectors for Model Representation which is an algorithm developed at Stanford University. In Glove, word embeddings are created by aggregating global word co-occurrence matrices from a given word or text corpus (Passalis & Tefas, 2018). Each word is represented as a vector where words with similar meanings have similar representation. The glove algorithm uses a text corpus to create a co-occurrence matrix Γ. The co-occurrence matrix shows the frequency of occurrence of a particular word pair. Each value in the co-occurrence matrix represents a pair of words occurring together. Γ (t/k) is the frequency of term t appearing in the context of term k, for some pre-established context window size. The matrix can be used to estimate the occurrence probability of term t in the context of term k as follows:

$$P(t|\kappa) = \frac{\Gamma(t,k)}{\sum_t [t,k)} \tag{2.8}$$

The matrix is then factorized using stochastic gradient descent to determine term vectors such that the dot product of vectors for any two terms t and κ approximates logP(t|κ). The dimensionality of this representation can be controlled and is typically orders of magnitude less than the dimensionality of the bag of words representation.

Each word vector can be represented in various dimensions - 50d, 100d, 200d, and 300d (Zaidi et al., 2020).

To get global document vectors we perform a weighted summation of term vectors for all terms occurring in a document, with term frequencies used as weights (Cichosz, 2018). The vector for document x would be calculated as;

$$a(x) = \sum_{t \in x} a(t) TFt(x) \tag{2.9}$$

Recently, the bidirectional encoder representations from the transformers (BERT) model have received much attention due to their bidirectional and attention mechanisms (Kenton & Toutanova, 2019). The BERT model which was first proposed by Google AI is a pre-training language model based on deep deep-learning neural network. As a neural network language model, it can directly train many untagged texts thus applicable in various natural language processing tasks such as text classification and sentiment analysis. The advantage of such models is that they are only fine-tuned without retraining when applied to different tasks (Wen et al., 2023). BERT or Bi-Directional Encoders Representation from Transformers is a text representation method of converting text into vectors that can be used to create machine learning models for NLP tasks such as sentiment analysis. BERT takes input text which should have special tokens [CLS] and [SEP] showing the start and end of the input text (such as a sentence) respectively. For more than one sentence, the sentences should be of the same length, if one sentence is shorter, [PAD] tokens are added to make the sentence of equal length. Further, the length of any sentence should not exceed 512 tokens including special tokens (Prottasha et al., 2022). BERT is an unsupervised and deeply bidirectional word embedding model. BERT is unsupervised since it was trained using an unlabeled plain text corpus. The BERT model is bidirectional because for it to generate a word representation it considers the words before and after the reference word. Therefore, each word is contextualized using words to its left and its right. The main advantage of BERT is its ability to be adapted to various NLP tasks easily. A visualization of the BERT model is presented in Figure 2.3 (Prottasha et al., 2022).

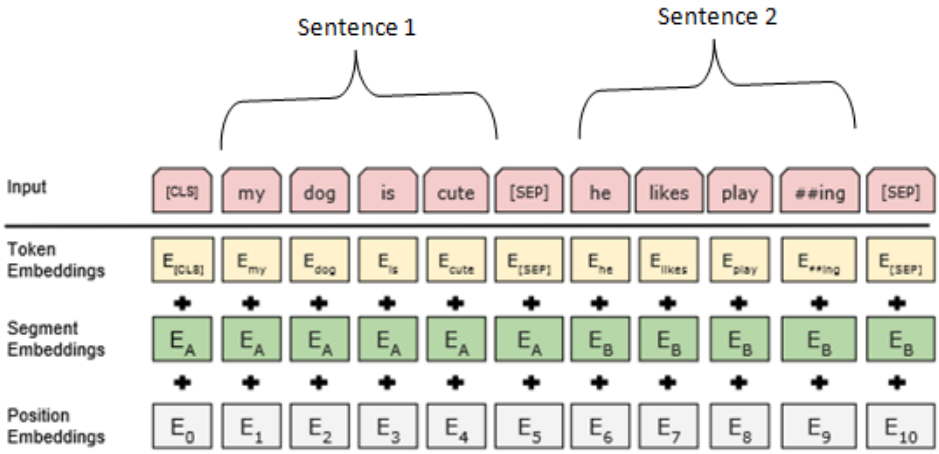


Figure 2.3: BERT Model Architecture

As shown in Figure 2.3, the two sentences are packed together as a contiguous text but separated by a special token ([SEP]). For each token, its vector representation is obtained by summing the corresponding token, segment, and position embeddings. For single-sentence classification tasks such as sentiment analysis, each token is converted to a contextualized vector as shown in Figure 2.4 (Prottasha et al., 2022).

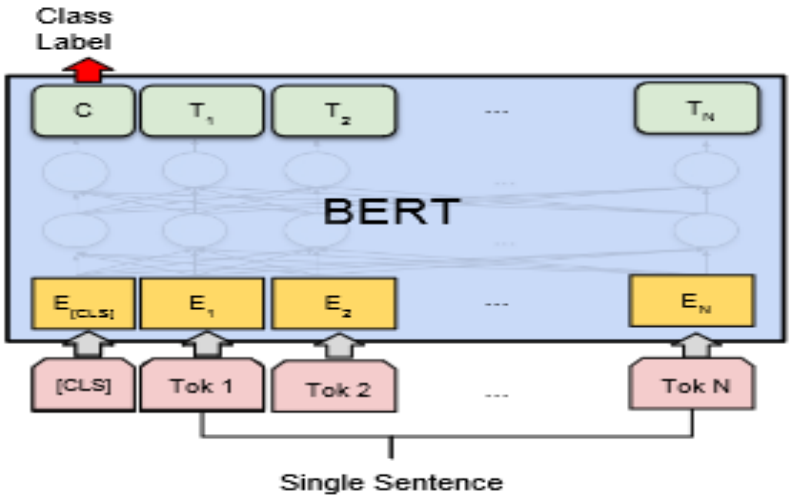


Figure 0.1: BERT Embedding Model for a Single Sentence.

The use of BERT embedding-based models outperforms other models due to their ability to show relationships between words (Jain et al., 2022).

Other transformer-based models have been proposed with the capability of modeling bidirectional contexts like BERT. They include XLNet (Yang et al., 2019) which was developed by Google Brain and Carnegie Mellon University. XLNet is a generalized autoregressive pre-training algorithm that overcomes the limitations of BERT by learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order. XLNet integrates ideas from Transformer-XL which is an autoregressive model into pre-training. From empirical studies, XLNet outperforms BERT on various text classification tasks including sentiment analysis (Yang et al., 2019). BERT on the one hand, uses a masked language model while XLNet on the other hand uses a permutation-based training objective that allows it to learn from all the words in the sentence. This capability of XLNet enables it to overcome the pre-training-fine-tuning discrepancy of BERT. Therefore, XLNet combines the advantages of autoregressive language modeling such as GPT with those of auto-encoding such as BERT in NLP tasks such as sentiment analysis. The drawbacks of XLNet are that it requires significant computational resources and time for training due to its large size and complexity, it can easily overfit on small datasets if not properly regularized, and XLNet performs well with tasks involving longer text sequence (Yang et al., 2019).

Several studies have been done on word embeddings. Garg and Subrahmanyam (2022) researched word embeddings and established that Word2Vec embeddings performed better than the other word-embedding algorithms. Kim (2014) studied the use of pre-trained Word2Vec vectors as inputs to convolutional neural networks and improved their performance by hyperparameter tuning of the CNN model. Wang et al. (2016) used pre-trained Glove vectors as inputs for attention-based LSTM models for aspect-level sentiment analysis. Liu et al. (2018) used pre-trained Word2Vec in the idiom recommendation model in essay writing. Liu et al. (2021) used a pre-trained Word2Vec model and improved it for cross-domain classification by extending the vector to include domain information. Recently D'Silva and Sharma (2022) used FastText pre-trained word embeddings and neural networks to classify Konkani texts. Hu et al (2022) used BERT to integrate mental features and short text vectors to improve topic classification and false detection in a short text.

Although most studies showed better performance of word embeddings, they did not compare their proposals with other word embedding models. They also suggested more research to be performed on the application of BERT in other contexts of text classification. Prottasha et al. (2022) did a study to compare Word2Vec, Glove, FastText, and BERT. They demonstrated that transformer architectures, such as BERT models, are state-of-the-art models for text representation and play a crucial role in sentiment analysis. The superiority of BERT is that it can read a series of words in either direction, unlike other word embedding algorithms. Further, BERT employs the attention mechanism of the transformer that assigns a word its vector, depending on the surrounding words. This mechanism enhances the semantic representation of the target text. However, the series of input words to be read by the BERT algorithm maintains the entire words of the target text. We propose that the performance of the BERT algorithm and other word embedding algorithms can be enhanced by reducing the input text to a few words, which contain sentiment information and their contexts. This can be guided by the utilization of sentiment lexicon and word N-grams.

2.7 Feature Selection Techniques

Feature selection is also referred to as variable selection, attribute selection, feature dimensionality reduction, and variable subset selection (Alnuaimi et al., 2019). It is the process of reducing input features to the optimal features that can be used in prediction or classification model building. It's aimed at choosing a smaller subset of the relevant features from the bigger original set of features by removing irrelevant, redundant, or noisy features. This leads to better machine learning model development and interpretation (Miao & Niu, 2016). Removing too many features might result in a loss of information, but keeping too many might not result in the desired benefit. Feature selection and feature extraction are closely related but different. Whereas the two are used to reduce the number of features in a dataset, in feature extraction the features selected are changed from the corpus but in feature selection, the robust features are selected from the input set without any change. The main objective of feature extraction and selection techniques is to provide the most relevant features that support the development of robust and accurate machine learning models (Alnuaimi

et al., 2019), (Aytug & Serdar, 2017) and (Bhadane et al., 2015) Such features are the most informative at the same time not redundant.

Feature selection techniques are broadly categorized into two categories which are static feature selection techniques for static data and streaming feature selection for streaming data. In this research static data was investigated thus the static flat features selection techniques were discussed. Figure 2.5 shows the static feature selection techniques taxonomy.

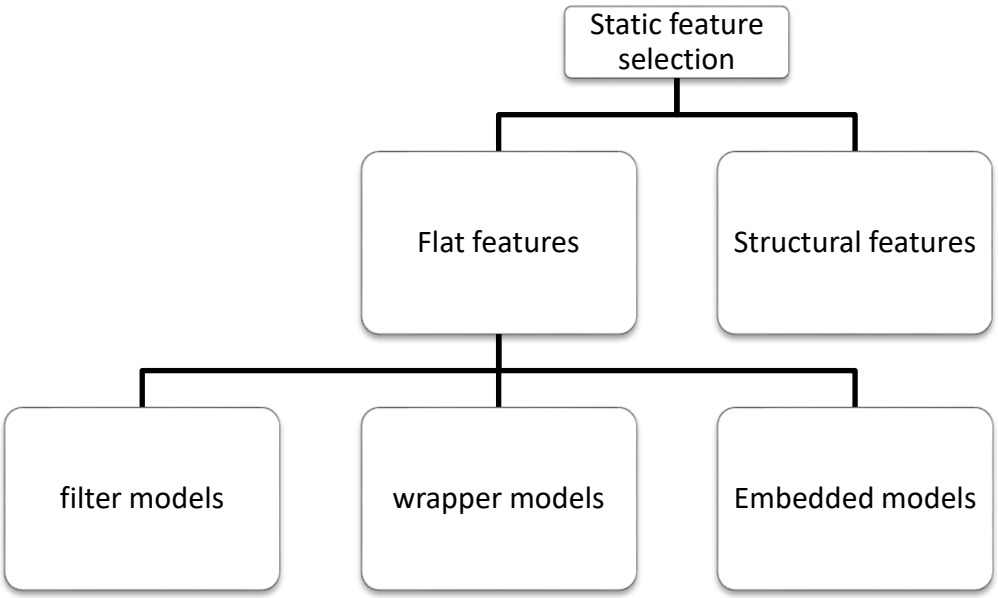


Figure 2.5: Static Feature Selection Techniques Taxonomy

Flat features are independent while structural features are dependent and thus graphical. There are three main groups of algorithms in the flat-features selection: filters, wrappers, and embedded models. (Alnuaimi et al., 2019)

Filter methods involve the use of statistical scores assigned to each feature using certain criteria. The scores are based on the characteristics of the data and thus independent of the machine learning algorithms used. As shown in Figure 2.6, the features are first ranked using the criteria, and the features with the highest ranks are selected and then used for training the Machine learning classifiers. Most information theory-based methods are examples of filter methods.

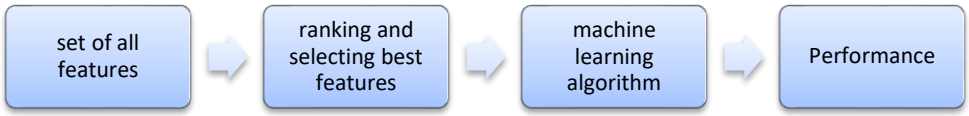


Figure 0.2: Filter Selection Methods Architecture

In wrapper methods, a subset of features is first selected and applied to the intended learning algorithm to evaluate how the algorithm performs with the features. Figure 2.7 describes how the wrapper feature selection methods select the most important features. Features are then removed and added recursively to construct a new subset until the best subset is obtained. These methods utilize the search algorithms thus computationally expensive in terms of time. Such methods include the Recursive Feature Elimination (RFE) using Support Vector Machines.

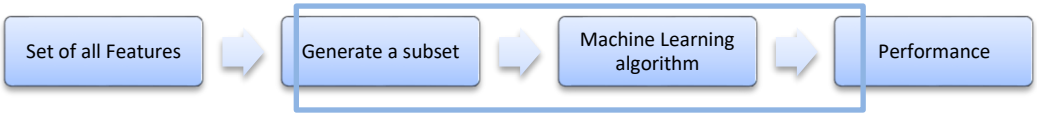


Figure 2.7: Wrapper Feature Selection Methods Architecture

Embedded models are a combination of filter methods and wrapper methods. Here the feature selection component is built into the machine learning algorithm. As shown in Figure 2.8, the best set of features is built during model formulation and development. The machine learning algorithms have an inbuilt capacity to identify the best features during classification. Examples include decision trees with pruning like IDE3.

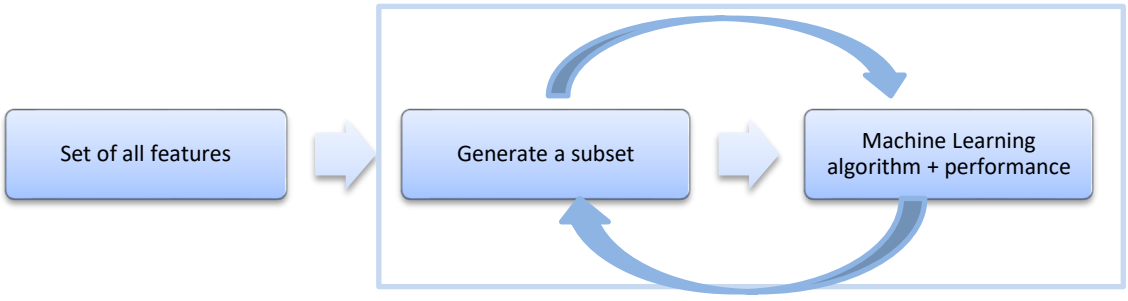


Figure 2.8: Embedded Feature Selection Methods Architecture

Information theory-based methods primarily originate from information theory. To measure the dependence of features, the amount of information that the features share is used. Any two features with high information imply that they are dependent hence one can be replaced by the other. For classification if a feature and a class have high information it implies the class is dependent on the feature thus the feature is relevant to the prediction. There are several Information theory-based feature selection methods. Ghosh and Sanyal (2018) did a study to explore the ability to combine three feature selection methods; IG, Chi-Square, and Gini Index to enhance and refine the performance of four machine learning classifiers namely SMO, MNB, RF, and LR on the multiple domains. The effectiveness of the classification algorithm was evaluated in terms of F measure, precision, and recall. The classifiers such as SMO, MNB, and RF performed best for reviews of movies, electronics, and kitchenware subsequently. The method they proposed in their work had drawbacks. They did not consider expressions of sentiments through images and emoticons. The comment in text format contains sarcasm, linguistic problems, etc. To predict the sentiment of that comment we have to understand the nature and ambience

Minimum redundancy maximum relevance (MRMR) is one of the Information theory-based methods and the most reliable approach due to its accuracy (Gallego et al., 2016). It's a filter method thus faster since filter methods are faster than wrapper and embedded methods (Zhou et al., 2020). The minimum redundancy maximum relevance (MRMR) method selects the features by first eliminating redundant features. This is achieved by measuring the information quantity that the two features share. If two features have a high mutual information quantity are highly correlated, and therefore, one can replace the other without loss of information (Houda et al., 2014). MRMR selects features with a high correlation with the class (relevance) and the least correlation with other features (redundancy). To determine relevance, the F-statistic is used for continuous features, and the Mutual Information Quotient is used for discrete features. For redundancy, the Pearson correlation coefficient is used for continuous variables, and Mutual Information Quotient is used for discrete variables (Gallego, et al., 2016), (Zhou et al., 2020) and (Houda et al., 2014). Mathematically for discrete features first the Mutual Information (Imin) for each feature is determined as the level

of similarity between it and each of the other features calculated as shown in equation 2.10.

$$Imin(f_i: f_y) = \sum_{i \in I} \quad \sum_{y \in Y} P(i, y) \log \frac{P(i,y)}{P(i)P(y)} \tag{2.10}$$

Where *fi* is the feature and *fy* is the other feature being compared. This is calculated for all features in the vector. Secondly, the relevance is determined by calculating the Mutual Information between a feature and the class (Imax) as shown in equation 2.11.

$$Imax(f_i: C_i) = \sum_{fi \in I} \quad \sum_{Ci \in I} P(fi, Ci) \log \frac{P(fi,Ci)}{P(fi)P(Ci)} \tag{2.11}$$

Where *fi* is the feature and *Ci* is the class. This is used to eliminate the irrelevant features since features with high mutual information with the class have a high relationship with the class thus more relevant. To select the most relevant features the Mutual Information Quotient between I*min* and I*max* for each feature is noted and the features are sorted from the one with the maximum quotient. Thus MRMR sorts the features from the one with the highest value of MRMR to the one with the lowest value.

2.8 Supervised Machine Learning Algorithms for Sentiment Analysis

Several supervised machine learning techniques are available for text classifications as well as in sentiment analysis. They include Naive Bayes classifier, Decision Trees, Support Vector Machines, and K- Nearest Neighbour. These techniques are used for classification after text representation and feature selection.

2.8.1 Support Vector Machines

Support Vector Machine (SVM) is a supervised machine learning classifier that is used in both regression and classification problems. SVM performs classification tasks such as sentiment analysis more accurately than regression problems. They are more suitable in classification tasks that involve very high dimensional data such as text classification compared to other machine learning classifiers (Hassan et al., 2022). SVM can classify both linear and non-linear data.

For linear data, the Maximum margin Hyperplane (MMH) is used to separate two data points in which the distance between the points is maximum. This maximum distance is called the support vector as shown in Figure 2.9. The classifier determines the best hyperplane in the vector space for low generalization error (Saraswathi & Tamilarasi, 2014). For text, the purpose of the vector space model is to classify the input documents into a class based on the hyperplane.

For non-linear data, SVM classifies the data by use of kernel function. A kernel function transforms input data into higher dimensions to classify it, as shown in Figure 2.10. There are different types of kernel functions available that we can be used for classification purposes (Devika et al., 2016). During classification, the SVM classifier has to find out the proper kernel function to classify data points appropriately since there are several types of kernel functions.

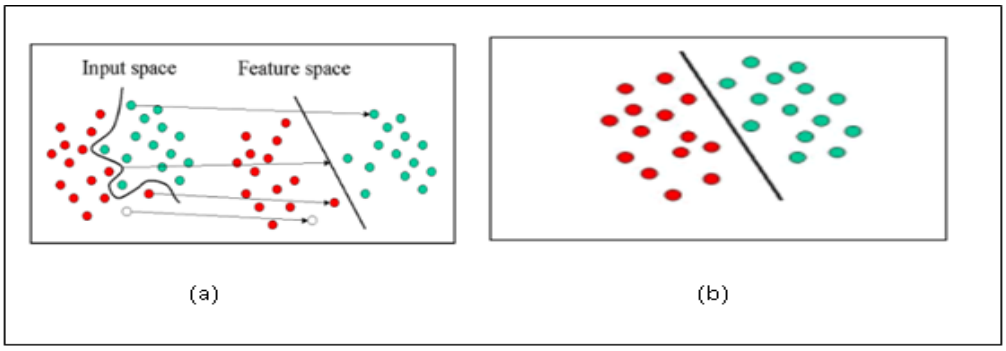


Figure 2.9: Support Vector Space Plane Separator

Original objects in Fig 2.9(a) are mapped or transformed using a mathematical function known as kernel. After transformation, the mapped objects are linearly

separable by a plane as shown in Figure 2.9(b) thus avoiding the complexity of curve separators (Devika et al., 2016).

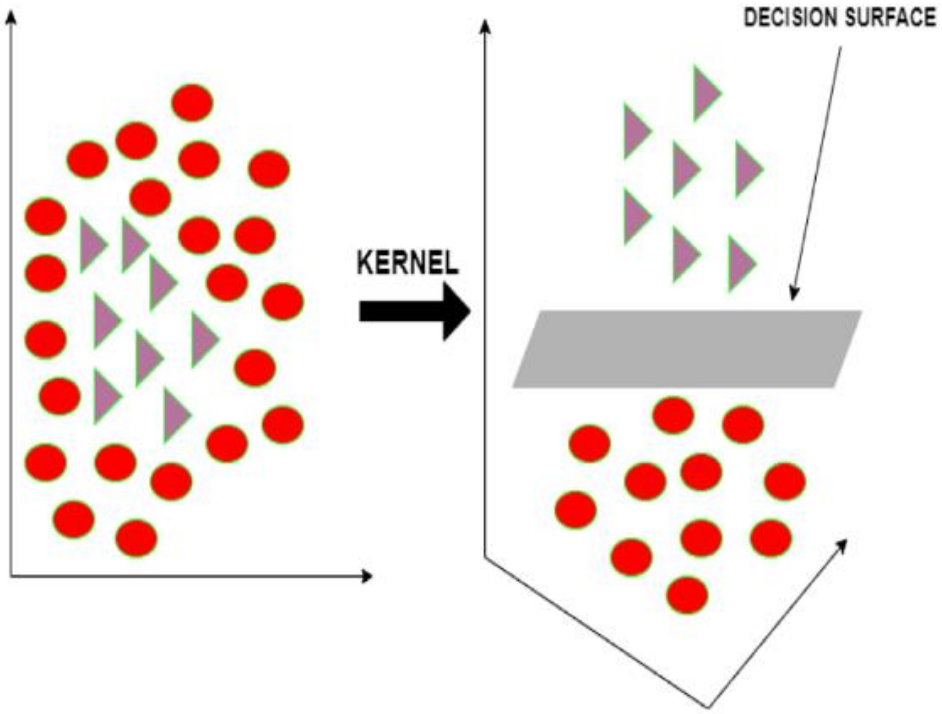


Figure 2.10: SVM Non-linear Classification

Vector space models have several attractive properties which include; they require much less labor than other approaches, they can classify huge data, and that they are robust when there is a small set of examples distributed over a large area (Pang & Lee, 2008) (Devika et al., 2016). However, for more than two classes the SVM classifier tends to be slow (Brindha et al., 2016). In this study, we sought to classify social media text data into two classes thus SVM classifier will be used.

2.8.2 Naïve Bayes Classifier

Bayesian classifiers are based on the Bayes rule, whose main basis is conditional probabilities. Conditional probability is a probability that event X will occur, given evidence Y normally written as$P(X \mid Y)$. The Bayes rule allows us to determine this probability when all we have is the probability of the opposite result and the two components individually:

$$P(X \mid Y) = P(X)\, P(Y \mid X) / P(Y) \quad (2.12)$$

The central assumption of Naive Bayes classification is that, within each class, the values of attributes are all independent of each other. Then by the laws of independent probability:

$$P(a_1, a_2, \ldots a_n \mid c) = P(a_1 \mid c) P(a_2 \mid c) \ldots P(a_n \mid c) \quad (2.13)$$

$$P(a_1, a_2, \ldots a_n \mid c) = \prod_{i=1}^{n} P(a_i \mid c) \quad (2.14)$$

In this case, we are trying to estimate the probability that a sentence is positive or negative or the extent of negativity or positivity, given the Sentiment Term Position features. Naïve Bayesian (NB) classifier is one of the most effective and efficient classification algorithms. Compared to other classifiers, NB requires relatively little data for training. It trains very quickly, requires little storage space during both training and classification, is easily implemented, robust to missing values, robust to noise or irrelevant attributes, and does not have a lot of parameters such as Neural Networks and Support Vector Machines (Kotsiantis et al., 2006), (Brindha et al., 2016) and (Simha & Iyengar, 2007). In this work, social media data has a lot of noise. The disadvantage of the NB classifier is that it does not work well with qualitative and continuous data and therefore such data needs to be discretized which can lead to information loss.

2.8.3 Decision Tree Classifier

Another commonly used classifier in sentiment analysis is the decision tree. Decision Trees are implemented using several algorithms with the most popular being the Classification and Regression Tree (CART) and Independent Dichotomous (ID3) algorithms. CART is used for constructing a binary decision tree a dichotomous classification model. Each node of the tree in the division has only two ancestors. Thus this algorithm works well when dealing with a binary type of classes. The number of

descendants in a node of the algorithm is not limited to the ID3 algorithm. The algorithm cannot work with a continuous target field, and therefore it solves only the problem of classification. In ID3, the selection of attributes takes place based on the information gain, or the basis of the Gini coefficient. *C4.5* is an improved version of the ID3 which is used to normalize the rules generated by ID3. The advantages of Decision Trees are that; they are simple and unambiguous, provide a perceptive classification structure that is simple to interpret, and perform better with huge data (Brindha et al., 2016). The drawbacks of decision trees are that; they are time-consuming O (rm^2k^2), where: m - the number of attributes, k - the greatest value of the variants of value in the attribute, r - the number of tree leaves, and, which is most important, it cannot practically be parallelized and the disadvantage of overfitting (Brindha et al., 2016).

2.8.4 K-Nearest Neighbor Classifier

The k-nearest neighbor classifier is also used in text classification. The K-nearest neighbor algorithm searches the pattern space for the K training tuples as the K nearest neighbor of the unknown tuple. The main idea of the KNN algorithm is to determine similarities between the unknown sample and all training samples to find the top K-nearest neighbors of the unknown sample. Closeness, which is a distance metric such as Euclidean distance, is one way of determining the class of a new sample. The advantages of KNN are that it's simple to classify, effective, easy to understand, and works well in pattern recognition (Brindha et al., 2016). However, the limitations of the KNN algorithm are that when the text has a high-dimension feature space the speed of the algorithm is slow, overreliance on training data, and all training samples are treated equally with no differences between small and huge ones (Brindha et al., 2016). This makes K-NN take time to classify a new sample hence it's also called lazy learner since the classification takes place at the point of classification instead of during training as shown in Figure 2.11.

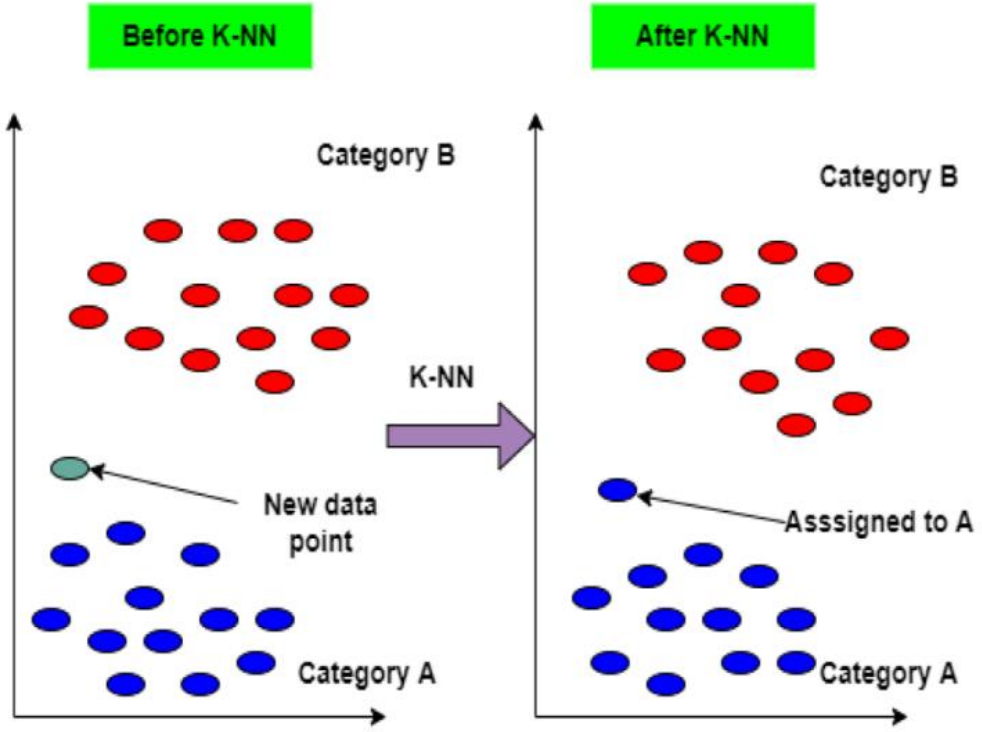


Figure 2.11: Classification Using K-NN

From Figure 2.11, the new data point is classified into category A since it is near the category and its color (green) is close to the color of category A (blue).

2.9 Deep Learning for Sentiment Analysis

Deep learning algorithms are machine learning algorithms that emulate the functioning of the human brain in artificial intelligence decision-making processes such as text classification. Deep learning algorithms are capable of learning unsupervised data that is unstructured or unlabeled. Deep learning is also known as deep neural learning or deep neural network. Deep learning has become one of the promising algorithms in the domain of machine learning (Samira et al., 2018). This is because it has been widely used in image classification and related artificial intelligence tasks (Yang, 2022). Deep learning builds artificial intelligent models by representing abstractions of data in multiple layers of basic machine learning algorithms. (Samira et al., 2018) These layers are interconnected hence the name neural networks. Some of the deep learning algorithms include the Convolutional Neural Network (CNN) and long short-term memory networks (LSTM).

Deep Learning CNNs have widely been used in image classification than in text classification. Nevertheless, some researchers have proposed the use of deep learning in text classification. Word embedding is one of the commonly used feature selection techniques that utilize deep learning algorithms. According to Mikolov in (Mikolov et al., 2013) research in word embedding feature selection gained momentum in 2013. (Al-Azani et al., 2017) used word embedding in the classification of short Arabic texts. The main limitation of his work is that he did not consider the polarity of the words and the context. (Rudkowsky et al., 2018) seemed to be the current research on word embeddings as a feature selection technique. Words that have similar meanings were clustered together. However, the concept of placement can change the sentiment of the word depending on the neighboring words or phrases. Deep learning algorithms have shown promising results in text classification and natural language processing tasks (Khedkara & Shinde, 2020).

The most frequently used deep learning models in text classification and sentiment analysis are based on Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN). In Recurrent Neural Network (RNN)-based models, text is considered as a sequence of textual units like words thus they capture the units' dependencies and structures for text classification. While RNNs are trained to identify and analyze patterns with time aspect, CNNs are trained for patterns within a vector space (Guo, 2020). CNNs have become popular model architectures for text classification compared with RNNs since they can detect local and position-dependent patterns (Noaman H. M., 2018). However, these models have their shortcomings; mainly they are weak in capturing rich information from the text and they perform poorly in long-span word relations in the text thus they often underperform feed-forward neural networks (Noaman H. M., 2018). This is because most of these models depend on pertained word embedding. However, the majority focus on feature extraction using CNN, and a small number on combining CNN with traditional feature extraction techniques. Shrestha and Mahmood in their work (Shrestha & Mahmood, 2019) agreed that deep learning can also overcome the limitations of traditional shallow networks that had limitations of lacking efficient representations of multi-dimensional training data including text data. This idea is supported by Cheng that Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) have

achieved good results in sentiment analysis and text classification compared to the basic networks (Cheng et al., 2020). They argue that although CNN can extract local information between consecutive words of a sentence, it ignores the contextual semantic information between words, a problem they try to solve by combining Bidirectional GRU with CNN. Therefore, they proposed a multi-channel model that combines the CNN and the bidirectional gated recurrent unit network with an attention mechanism. The researcher further objectively points out that the weakness in their work is that it does not consider all the aspects of sentiment analysis. Nevertheless, they put forth two important recommendations which this research has exploited; that multi-channel deep learning can improve sentiment analysis model and the need to explore combining traditional sentiment analysis with deep learning models. These recommendations critically implied that any work that combines traditional word feature representation with deep learning models would improve sentiment analysis accuracy. Such a recommendation is also supported by Zhenyu et al. (2020) that applying a proactive model to obtain multichannel representations of texts could lead to textual data amplification thus texts could be represented in rich semantics.

Despite deep learning algorithms have changed the perception of information processing, there exists a gap in understanding this fast-paced domain. The lack of core understanding renders these powerful methods black-box machines that inhibit development at a fundamental level (Samira et al., 2018). Further manually selecting the best architecture and hyper-parameters for these deep learning models and utilizing them in sentence-level sentiment analysis is an expensive task (Dahou & Abd, 2019). Several research works have been done to improve RNNs, CNNs, and LSTM models for text classification. To optimize CNN and RNN parameters recent research is being done in an area referred to as Neuro-Evolution (NE) which is considered an artificial intelligence area (Dahou & Abd, 2019). This research aims to design architectures and hyper-parameters automatically using algorithms rather than hand configuration. For instance, Young (Young et al., 2015) proposed a multimode evolutionary neural network for deep learning (MENNDL) where they used a genetic algorithm (GA) to optimize CNN hyper-parameters. The weakness in their work is that they chose only filter size and number as the parameters to tune. Further, their approach had a risk of settling at a local minimum. Dahou used a convolutional neural network and

differential Evolution Algorithm to optimize an Arabic classification deep learning model (Dahou & Abd, 2019). They used normally built Arabic word embedding from CBOW58 and tuned parameters of a parallel CNN. Their work yielded better performance results but it was applicable in Arabic sentiment analysis and also they optimized the fully connected layer's neurons which was more of a black box optimization. Loshchilov and Hutter (2016) proposed a covariance Matrix Adaptation Evolution Strategy (CMA-ES) to optimize several parameters in the convolution and fully connected layers.

2.10 Improvement of Machine Learning Techniques

Generally, several methods in sentiment analysis use Machine Learning classifiers as discussed in sections 2.7 and 2.8. The advantages and disadvantages of each of the supervised machine-learning techniques have been discussed in the previous section. Several researchers agree that such methods have weaknesses. They include (Medhat et al., 2014) who presents a survey of Sentiment Analysis algorithms and applications. (Bouazizi & Ohtsuki, 2017) points out that these methods are time-consuming especially when the scoring of the words is done manually (Han & Kwangmi, 2017). To minimize the drawbacks of individual classifiers combination of several classifiers using suitable ensemble classifiers increases the performance of individual classifiers (Catal & Nangir, 2017) and (Wang et al., 2014). Architecture configuration or hyper-parameter is also another method of improving the performance of machine learning classifiers

2.10.1 Ensemble Classification

An ensemble classifier is a fusion of several classifiers, where a result is calculated based on the results of the models being fused at each prediction stage (Bird, et al., 2019). There are three methods of building ensemble classifiers. These are voting, bagging, and stacking. In voting, all base classifiers are implemented parallel to each other. The output class is identified by use of an appropriate voting mechanism such as the majority vote mechanism where the class predicted by most base classifiers is taken to be the vote, Average of probabilities, Maximum/minimum probabilities, or median vote (Bird et al., 2019). In bagging a single base classification is implemented

in series to a portion of the dataset until the best prediction implementation is identified and used to perform the classification task. This technique is suitable for base classifiers that have difficulties with huge datasets since the dataset is subdivided at each bag. In Stacking, at least two base classifiers are combined. These classifiers are trained on the dataset and their results are used as a training set of the stacking classifier. This ensemble utilizes a 'chain' technique of combining the base classifiers. Examples of ensemble classifiers are Adaptive Boosting (AdaBoost) which is a bagging ensemble algorithm, Random Forest which is a bagging and voting ensemble for random trees and Gradient Boosted Decision Tree which is a bagged ensemble classifier of Decision Trees.

Ensemble classifiers provide promising results in classification problems but they have not been applied substantially in Sentence level SA (Ankit, 2018). Ankit (Ankit, 2018) proposed an ensemble classifier using the voting technique of three aggregated base classifiers (Naïve Bayes, Random Forest, Support vector Machines, and Logistic regression). He however did not investigate other ensemble techniques which may provide better results. Similarly, (Wang et al., 2014) investigated bagging, boosting, and Random Subspace on five base classifiers, Naïve Bayes, Maximum Entropy, KNN, and SVM. They did not investigate other ensemble classifier fusion approaches.

2.10.2 Architecture Configuration and Hyper Parameter Tuning

Effective performance of machine learning classifiers and deep learning models requires the configuration of their architecture to suit the classification problem at hand. This configuration is the process of identifying the best parameter combinations of the model that make it more effective and efficient. The manual selection of the best architecture and hyper-parameters for these models and utilizing them in sentence-level sentiment analysis is an expensive task (Dahou & Abd, 2019). Several research works have been done to improve these models such as the SVM, RNNs, CNNs, and LSTM models for text classification. To optimize SVM, CNN, and RNN parameters recent research is being done in an area referred to as Neuro-Evolution (NE) which is considered an artificial intelligence area (Dahou & Abd, 2019). This research aims to design architectures and hyper-parameters automatically using algorithms rather than

hand configuration. For instance, Young et al (Young et al., 2015) proposed a multimode evolutionary neural network for deep learning (MENNDL) where they used a genetic algorithm (GA) to optimize CNN hyper-parameters. The weakness in their work is that they chose only filter size and number as the parameters to tune. Further, their approach had a risk of settling at a local minimum. Abdelghani et al (Dahou & Abd, 2019) used a convolutional neural network and differential Evolution Algorithm to optimize an Arabic classification deep learning model. They used normally built Arabic word embedding from CBOW58 and tuned parameters of a parallel CNN. Their work yielded better performance results but it was applicable in Arabic sentiment analysis and also they optimized the fully connected layer's neurons which was more of a black box optimization. Researchers Loshchilov and Hutter (2016) proposed a covariance Matrix Adaptation Evolution Strategy (CMA-ES) to optimize several parameters in the convolution and fully connected layers.

2.11 Web 2.0 and Micro Blogging

Social media today is mostly being used to show how people think about an issue or their opinion on certain aspects of events, objects, or issues. Social media content by nature reflects opinions and sentiments, while traditional content analysis tends to focus on identifying topics (Pang & Lee, 2008). This has made the accessibility to information about how people feel about things more readily available to the masses through social media. Feldman (2013) agrees that due to the combination of an increase in the volume of data available and more complex concepts to analyze, the need for accurate analysis approaches has grown.

Twitter is the second most popular Social Networking Site, with approximately 310 million estimated unique monthly visitors and 500 million tweets per day. Social Networking Site data has the following characteristics; it's readily available, it's topical in that users' contributions are on a specific topic, and it's sparse such that the posts are short in length (140 characters limit in Twitter). Their limited size implies little extra information that can be used as evidence for identifying their sentiment. Non-standard vocabulary is used in posts that are informal, including slang words and non-standard expressions which is attributed to the limited size of messages (e.g. "4u" instead of

"for you"). The data is noisy since users post their messages without verifying their correctness concerning grammar or syntactic rules thus a lot of misspelled words and incorrect phrases. The use of diverse languages is also evident in social media. According to Jihso et al. (2014), each structured, semi-structured, or unstructured data incident contains Five dimensions; what does the data contain, why did the data occur, where did the data come from, when did the data occur, who received the data and how was the data transferred. The authors demonstrate the dimensions as shown in Figure 2.12 below;

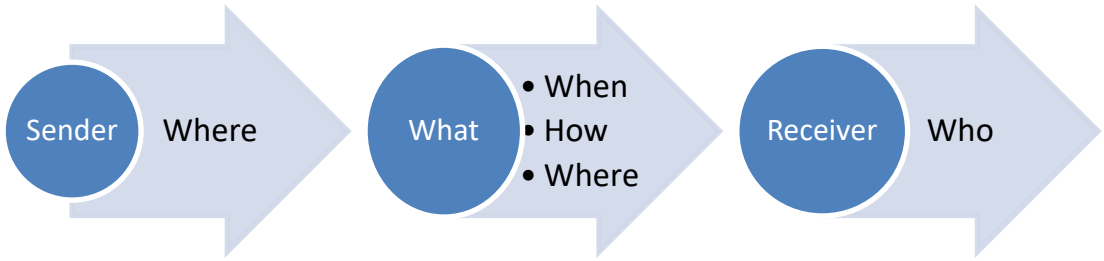


Figure 2.12: 5 Ws Dimensions of Social Media Data

From this model, several attributes will be identified for analysis in this research work.

Understanding the emotions being conveyed by such a post on social media therefore becomes a difficult task for humans and also computers. However, when volumes of opinions are very high, human processing becomes a challenge, hence the need for automated processes to extract sentiments from a variety of sources that keep growing in volume, complexity, and diversity (Appel et al., 2015).

2.12 Evaluation of Classification Process

The effectiveness of the classification process is achieved by measuring the accuracy, precision, and recall value which is effectively displayed in a confusion matrix. According to Egejuru et al. (2017), the square matrix is of order ($Ci * C_i$) where C_i is the number of classes in the prediction model. Therefore, the confusion matrix of order 2 will be as shown in Figure 2.13;

	True Positive	True Negative
Predicted Positive	A	B
Predicted Negative	C	D

Figure 2.13: Confusion Matrix for Binary Classification

From Figure 2.13, weighted accuracy, weighted precision, and weighted recall were used as performance metrics as shown in equations 2.15, 2.16, 2.17, and 2.18. Accuracy is the proportion of correctly predicted cases to the total cases.

$$Accuracy = \frac{A+D}{total\ cases} \tag{2.15}$$

Recall is the proportion of actual cases correctly classified. It is also referred to as sensitivity or true predicted (TP) rate.

$$Weighted\ Recall = \frac{TP\ positive+TP\ negative}{2} \tag{2.16}$$

Where; $TPpositive = \frac{A}{A+B}$

$$TP\ negative = \frac{D}{C+D}$$

Precision is the proportion of correct predictions

$$Weigthed\ Precision = \frac{Precision\ positive+Precision\ negative}{2} \tag{2.17}$$

Where; $Precision\ positive = \frac{A}{A+C}$

$$Precision\ negative = \frac{D}{B+D}$$

$$F1\ =\ 2 * precision *\ Recall/(\ precision + Recall) \tag{2.18}$$

2.13 Contributions of the Study

From the related works discussed above, sentence-level sentiment analysis is an important area of research in the current era of rapidly growing user-generated content. The summary of recent studies reviewed in the area of text representation and sentiment analysis, their performance, and shortcomings is presented in Table 2.1.

Table 2.1: Summary of Recent Studies on Text Representation Techniques

Research Work	Sentiment Analysis Domain Classes	Text Representation Technique	Classification Technique	Accuracy (%)	Limitations/weakness
Onan (2020)	Negative, positive, and neutral	Glove Weighted word embedding	CNN	87.12	The length of the documents/reviews affected the performance
Cheng et al.(2020)	Positive, negative	Word2vec (CBOW)	Bidirectional GRU with CNN	88.82	The entire document was used for word embedding
Cheng et al.(2020)	Positive, negative	Word2vec (CBOW)	Support Vector Machines (SVM)	79.86	SVM does not consider the contextual meaning of a word and ignores deep word semantics
Ren et al. (2020)	Positive, negative, neutral	Word2vec(Glove)	Lexicon-Enhanced Attention Network (LEAN).	79.1	The sentiment section is treated differently from the other parts of the document.
Mutinda et al. 2021)	Positive, negative	Hybrid Vector containing words, POS tags, sentiment orientations	Support Vector Machines, Naïve Bayes, Decision Trees, K-NN	90.15	Use of Baseline machine learning classifiers. There's a need to evaluate the performance with deep learning classifiers
Jain et al. (2022)	Positive, neutral, negative	Pre-trained BERT as word embeddings	Bidirectional Encoder Representations from Transformers-based Dilated Convolutional Neural Network(BERT-DCNN)	86.3	The text representation model was not improved. BERT embedding was used as usual but the CNN was changed to Dilated CNN
Mutinda et al. (2021)	Positive, negative	BoW with TF-IDF	Support Vector Machines	88.64	Use of Baseline machine learning classifiers. There's a need to evaluate the performance with deep learning classifiers
Mutinda et al. (2021)	Positive, negative	BoW with TF-IDF	Support Vector Machines	89.01	Use of Baseline machine learning classifiers. There's a need to evaluate the
Wang et al. (2021)	Positive, Negative	Refined-Word2vec	CNN	89.2	Embedding of Sentiment words was done separately and later combined with other embeddings thus repetition and redundancy

Research Work	Sentiment Analysis Domain Classes	Text Representation Technique	Classification Technique	Accuracy (%)	Limitations/weakness
Wang et al. (2021)	Positive, Negative	Refined-Glove	CNN	89.7	Embedding of Sentiment words was done separately and later combined with other embeddings thus repetition and redundancy
Yang et al.(2019)	Positive, Negative	XLNet	CNN	96.21	Although XLNet combines the advantages of autoregressive and auto-encoding language models, it does so by considering the entire text. This leads to a high dimensionality of features. Works well with long text sequence

From Table 2.1, it is notable that the use of deep learning classifiers improves sentiment classification accuracy. However, we noted a need to improve the text representation approaches for both base classifiers and deep learning algorithms such as CNNs for sentiment analysis. Most of the research done in SA is on the application of supervised learning approaches in Sentiment analysis and most researchers suggested hybrid methods to advance SA research Wang et al. (2021). Research done on improving text representation mainly concentrates on combining word weighting methodologies such as TF-IDF with embedding techniques such as BERT (Onan 2020), Cheng et al.(2020), Mutinda et al. 2021), Wang et al. (2021).

In conclusion, research has been done on the area but still, there exists some gaps. It's evident from the review that text representation is a critical step in sentiment analysis and a key determinant in the classification accuracy and efficiency of Sentiment Analysis models. A good text representation technique will increase classification accuracy and reduce classification costs. The state-of-the-art text representation techniques such as BERT Jain et al. (2022) suffer several shortcomings. These approaches do not consider relationships between words, they ignore words' characteristics for instance word sentiment orientation and they suffer high feature dimensionality Mutinda et al. (2021), Yang et al. (2019) and Jain et al. (2022).

Most sentence-level sentiment analysis classifiers are based on basic machine learning algorithms thus a need to improve on them. Deep learning algorithms have brought new strengths in classification with much emphasis on image classification. There is

therefore need to investigate how deep learning algorithms can be used in sentence-level sentiment analysis.

These research gaps in sentence-level sentiment analysis have not been adequately addressed in past research works.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Introduction

This chapter describes the research design and methods used in this research. The research design shows the systematic process that was used to link the theory, the empirical methods, and the text data for gaining insights into the problem investigated. The dataset used, the model formulation, and the model performance evaluation metrics used are also discussed in this chapter.

3.2 Research Design and Methods

The study adopted an experimental study design. An experimental research design is a framework of research where two sets of data or variables are used. The first set acts as the constant known as the control group while the second set known as the treatment group is used to evaluate the differences with the first (Curtis et al. 2022). The research process used is summarized inform of a workflow presented in Figure 3.1.

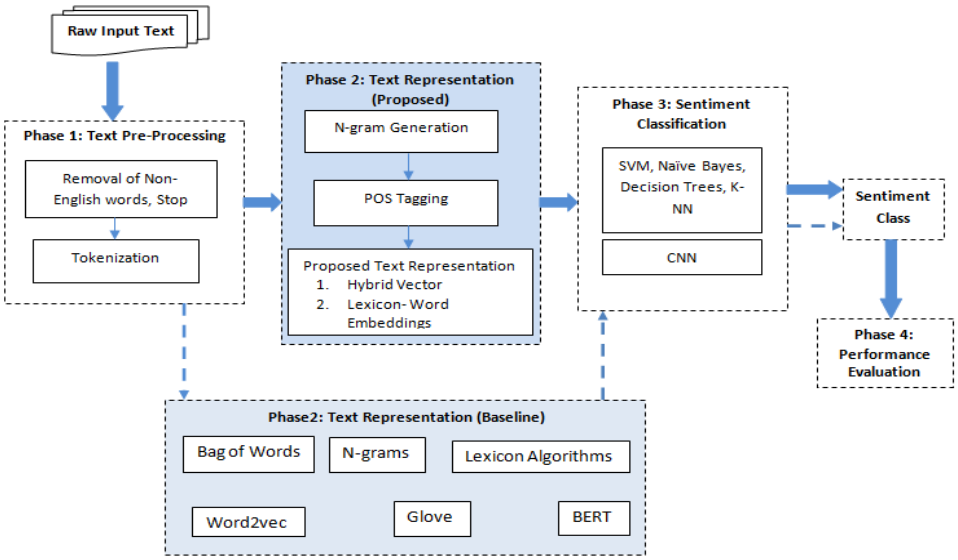


Figure 3.1: Summary of Study Design

The first phase as indicated in Figure 3.1 was text preprocessing but before then we had to obtain the raw text data from the public dataset. The main focus of the study was to investigate existing text representation techniques to develop and validate a novel text representation model that solves the drawbacks of data sparseness and high feature dimensionality of existing models as described in the thesis problem statement. Thus, the experiments done were to validate the performance of the proposed text representation model when applied with base machine learning classifiers and deep learning networks. Therefore, after the formulation of the model, experiments were set up where its performance was tested using machine learning classifiers including deep learning neural networks.

The research procedure employed in the study involved four main phases. Data preprocessing, development of a text representation model, development of a sentence-level sentiment analysis model, and validation of the developed model. The details of the phases are discussed in the subsequent sections. Varied tools were used appropriately in each step of the study. These experiments were done with a personal laptop with a CPU specified as Intel(R) Core(TM) i5-4210U CPU @ 1.70 GHz 2.40 GHz and a memory of 8.00 GB. The capacity of a personal laptop hard drive is 500 GB with Microsoft Windows 8.1 Single Language 64-bit Operating System, x64-based processor. Java libraries were configured in Rapid Miner Studio 9.002, a text-

mining integrated development environment. These libraries provided different modules for the experimentation as required. For word embeddings and Deep Learning experiments, Google's virtual laboratory Collaboratory was used.

3.3 Datasets

The experiments were carried out with a dataset compiled from three public datasets. The three world datasets were the Amazon products' reviews dataset, with 70,000 reviews, the Imbd dataset, with 50,000 movie reviews, and the Yelp dataset, with 300,000 restaurant reviews. In the experiments, we used 3000 reviews, as compiled by Kotzias et al. (2015) and published in a machine learning repository. For each website, Kotzias et al. (2015) randomly sampled 500 positive and 500 negative tweets, which were positive and negative. The details of the datasets used in the study are presented in Table 3.1.

Table 3.1: Summary of Datasets Used in the Experiments

Dataset	Source Website	Number of Tweets(Reviews)	Positive labeled (Kotzias et al.)	Negative labeled (Kotzias et al.)
Product Reviews	Amazon.com	70, 000	500	500
Movie Reviews	Imdb.com	50,000	500	500
Restaurant Reviews	Yelp.com	300,000	500	500

According to Yamane (1967), any population of more than 100,000 can be represented by a sample size of 400 samples or more. In our study, we used a sample size of 500 for each labeled thus representative.

3.3.1 Data Preprocessing

The tweets were cleaned of non-English words or sentences, abbreviations, emoticons and stop words using Java Stanford Core NLP libraries. To achieve this, Net Beans 8.0, a Java IDE was used. Stanford Core NLP libraries were downloaded and added to the Java class. Stanford Core NLP tokenizer was used to tokenize the posts and Part of Speech (POS) tagging. The Sentiment lexicon (Github, 2017) was used in the experiments.

The steps followed in the preprocessing of the data are shown in the Algorithm 1 pseudo code.

Algorithm 1: Pre-processing of data
Input Raw tweets

Output Pre classified data (cleaned data)

START

For each tweet(T)

CHECK for emoticons or images;

IF emoticon or image;

CONVERT to an English word;

ELSE PRINT the text (T1);

ENDIF

For each word in the text(T1)

READ word

If a word is in English

PRINT word (w)

ELSE delete the word

ENDIF

ENDFOR

FOR T1;

READ each word in T1;

If less than two letters or an HTML tag or a symbol delete;

ELSE PRINT the remaining words.

ENDIF

ENDFOR

ENDFOR

END

To implement the Pseudo code in Algorithm 1, Java functions were developed using Netbeans IDE in which Stanford Core NLP libraries were downloaded and added to the functions in Netbeans IDE version 8.2.

3.3.2 Tokenization and POS Tagging

The key features extracted from the texts were the words, POS tags, *N*-grams, and sentiment orientations of words. To achieve the research objectives, several feature matrices were prepared from the datasets for the experiments. One matrix was prepared using the Bag of Words ($N = 1$) of the tweets, the second one compiled using the normal *N*-grams ($N = 3$), and the third using a lexicon to convert the sentences into a bag of words semantic orientations and the fourth using the proposed approach described in section 3.4. The first three matrices were used to generate results as baseline approaches for comparison with the proposed approach. The matrices were used to train and investigate the performance of the proposed model in comparison with baseline approaches.

The steps followed in the annotation of the text data are shown in Algorithm 2.

Algorithm 2: Annotation of data

Input Cleaned text (Tweets)

Output annotated data

START

For each tweet(T3)

READ words one by one

Tokenize, split, POS tag, lemma;

PRINT the annotated text (T4);

ENDFOR

END

A description of the attributes of the data set is presented in Table 3.2.

Table 3.2: Dataset Attributes and Labels

Categories	Attribute	value(s)/ labels
Label	Sentiment Class (c)	1 - Positive 0 - Negative
Attributes	Words	Words
	Words' Polarity (O)	-1 – Negative(N), 0 – Neutral (0), 1 – Positive(P)
	Words POS tag (P)	1 – VBZ 2 – NN 3 – VB 4 – JJ 5 – VBG 6 – VBP 7 – IN 0 – others

From Table 3.2, the Part of Speech (POS) tags used were: Comparative Adjective (VBZ), Noun (NN), Verb in base form (VB), Adjective (JJ), Verb in present participle (VBG), Verb in Present Tense (VBP) and Proposition (IN). Any other POS tag was labeled as others (0).

The data was saved in MS Office Excel for analysis and import to other analysis tools.

3.4 Sentiment Lexicon-Based Text Representation Model

Sentiment classification models consist of three modules as discussed in chapter 2. These modules are the text data preprocessing module, text representation module, and sentiment classification module (Ankit, 2018). Sentence-level sentiment analysis is conceptualized as a sentiment classification problem in which a subjective sentence is categorized into an opinion class. In the proposed sentiment analysis model, the study sought to improve the text representation module of the sentiment analysis model. We first sought to investigate how sentiment lexicon can be used with conventional natural language processing techniques. Later we also sought to investigate how sentiment lexicon could be used with deep learning word embeddings algorithms.

3.5 Text Representation Using Sentiment Lexicon, *N*-grams, and POS Tagging

The study proposed and investigated a text representation model using sentiment lexicon and natural language processing techniques which included *N*-grams and POS tags. A model referred to as Lexicon - pointed hybrid *N*-gram Features Extraction Model (LeNFEM) was formulated and investigated. In the proposed model the text representation is achieved by extracting sentiment term aspects for the sentence text in the form of a hybrid vector which is then optimized using Minimum redundancy maximum relevance (MR^2), a feature dimensionality reduction algorithm. The conceptual framework of the proposed approach is shown in Figure 3.5.

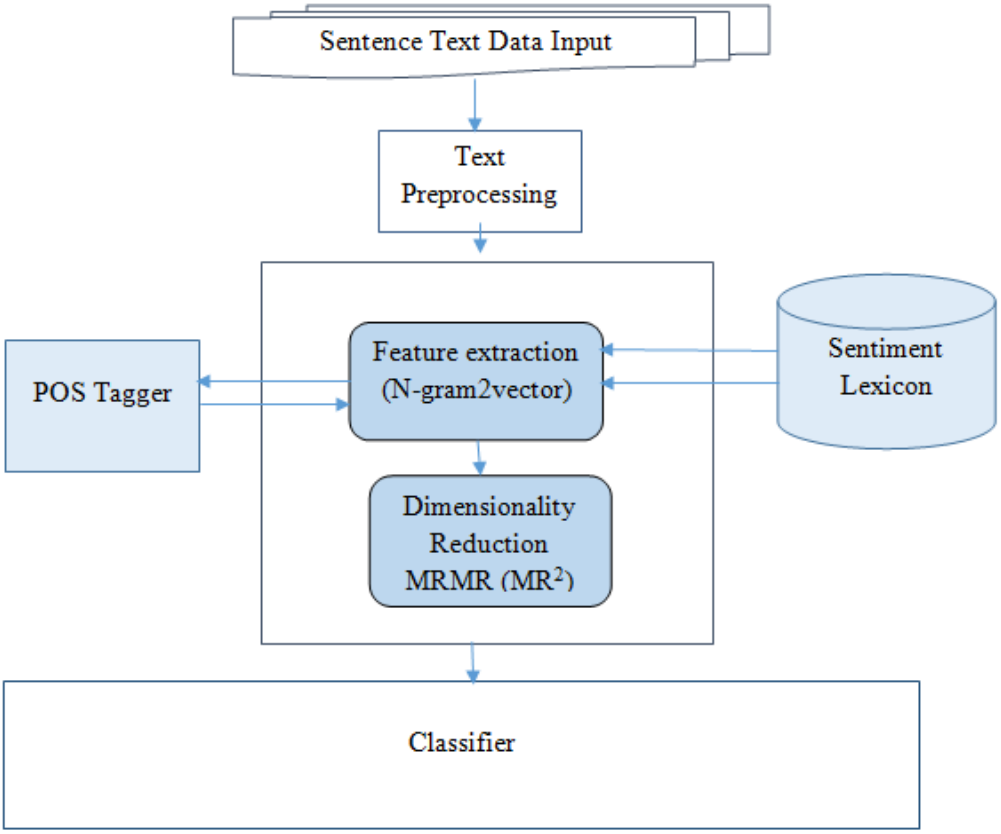


Figure 3.2: Proposed LeNFEM Conceptual Framework

The conceptual model presented in Figure 3.2 shows how the text representation was achieved by combining N-grams, Sentiment Lexicon, and POS tagger to form a text representation vector referred to as N-gram2Vector. The features of the resultant

vector are then selected using the MRMR algorithm. Each module of the proposed model is discussed in detail in the subsequent sections 3.5.1 to 3.5.7.

3.5.1 Sentence Text Data Input and Preprocessing

Social media texts are normally unstructured, noisy, and inconsistent (Noaman H. M., 2018). The tweets were cleaned of non-English words or sentences, abbreviations, emoticons, and stop words after input. Tokenization and transformation of the texts into lowercase was also done to split the sentences into separable words. The preprocessed data was used for text vector construction and representation.

3.5.2 Text Representation in LeNFEM

The novelty of the approach was in the modification of the text representation module. The study proposed the hybrid static *N*-grams2vec algorithm to generate sentence text representation in the form of a vector. The vector attributes were further filtered using the Minimum redundancy Maximum relevance (MRMR) algorithm. An *N*-gram is a statistical language model (LM) in which a document or a sentence is broken down into a sequence of words w_i *(* w_1, w_2, *…..,* w_n*).* Consider a sentence(*S*) $S=w_1, w_2, \ldots, w_n$, and from the chain rule, the sentence can be represented as a probability as shown in equation 3.1

$$P(s) = \prod_{i=1}^{n} P(w_i|w_1, \ldots . . w_{i-1}) \tag{3.1}$$

Since the parameter space of $P\ (w_i|w_1, w_2, \ldots, w_{i-1})$ is too large, the language model puts the context $W_{i-1}=w_1, w_2, \ldots, w_{i-1}$ into an equivalence class determined by a function $\Phi(W_{i-1})$. As a result;

$$P(s) = \prod_{i=1}^{n} P(w_i|\Phi(w_{i-1})) \tag{3.2}$$

N-gram is the most used language model which makes a Markov assumption and defines the context $\Phi\ (W_{i-1})$ as;

$$\Phi\ (W_{i-1}) = w_{i-n+1}, w_{i-n+2}, \ldots, w_{i-1} = h \tag{3.3}$$

For, *N= 1* the context does not exist hence it's the normal bag of words

N=2 the context becomes $\Phi(W_{i-1}) = w_{i-1}, w_i$ Thus two words are considered

N=3 the context becomes $\Phi(W_{i-1}) = w_{i-2}, w_{i-1}, w_{i.}$ Thus, three words are considered.

N-gram models are used in representation of sentences or documents in text classification and sentiment analysis. From the formal definition, an *N*-gram is thus a textual sequence containing *N* adjacent 'textual units' from a particular sentence or document. A 'textual unit' can be identified at byte, character, or word level depending on the context of interest from which a vector representation of the *N*-grams is formed. In this work, *N*-grams were identified the at word level.

To construct an n-gram vector, each sentence is converted into overlapping *N*-grams by running a pre-defined window of size *n*. Each of the *N*-grams is a coordinate in a vector that represents the text under study. The frequency, occurrence, or any other metric of this n-gram in the text becomes the value of this coordinate. The simplest n-gram is the unigram, where $N = 1$, which is the normal "bag-of-words" (BOW) representation. Generating the vector from a huge text dataset can be challenging. *N*-gram models are widely used in NLP tasks since they are simple and effective (Aisopos et al., 2016). However, in n-grams, each sentence is converted into a bag of *N*-grams and represented as a vector of occurrence frequency without taking into consideration the information encapsulated in the *N*-grams of the original text. This leads to so many irrelevant and redundant attributes in the vector. The sliding window of the *N*-gram also makes the variables more and thus some become less relevant.

A novel way of utilizing the *N*-gram model for feature extraction in sentence-level sentiment analysis is proposed in this work. We use the *N*-gram model to select a section of a sentence where sentiment orientation can be easily identified. In the proposal, the *N*-gram window is moved until we get one that contains a sentiment term. This is achieved by invoking a sentiment lexicon that points to an *N*-gram containing the sentiment term after the generation of the word n-grams of the sentence.

The identified *N*-gram is then split into a Bag of Three words. The three words are then used to construct a hybrid vector for the sentence from the words, their POS tag,

and their sentiment orientation. From equation (3.3) and using *N*=3, the context becomes;

$\Phi\ (W_{i-1}) = w_{i-2}, w_{i-1}, w_i$ for words;

$\Phi\ (P_{i-1}) = P_{i-2}, P_{i-1}, P_i$ for POS tags and;

$\Phi\ (O_{i-1}) = O_{i-2}, O_{i-1}, O_i$ for semantic Orientation (3.4)

We combine the three sets of contexts to get a hybrid of words, POS tags, and Semantic orientations called the sentiment aspects (A) given as;

$$\Phi\ (A_{i-1}) = W_{i-2}, W_{i-1}, W_{i,} P_{i-2}, P_{i-1}, P_{i,} O_{i-2}, O_{i-1}, O_i \quad (3.5)$$

Substituting equation (3.5) in equation (3.2) the proposed model becomes;

$$P(s) = \prod_{i=1}^{n} P(A_i|A_1, \ldots\ldots A_{i-1}) \quad (3.6)$$

Where sentence *S* is considered as a sequence of hybrid sentiment aspects including words, POS tags, and Sentiment orientations from *N*=3 *N*-gram as;

$$S = A_1, A_{2,} \ldots, A_n. \quad (3.7)$$

To obtain the vector of the words, POS tags, and sentiment orientations, binary occurrences or *TF-IDF* text vectorization schemes are used. In binary occurrence vectorization, if the word (w_i), the POS tag (Pi), or the sentiment orientation (O_i) is present in a sentence (S_i) it is denoted by one (1) otherwise if it is zero (0). TF-IDF is a text vectorization scheme that is calculated in two steps. First, the term frequency (*TF*) is obtained as the number of times a term (i) appears in a sentence (j) given as;

$$TF_i = tf_{ij} \quad (3.8)$$

Secondly, The Inverse Document Frequency (*IDF*) is given as the ratio of the total number of sentences in the corpus to the total number of sentences that contain at least an occurrence of term (i) given as;

$$IDF = Log\ (N/n_j) + 1 \quad (3.9)$$

Where; N is the total number of sentences and nj is the total number of sentences that contain at least an occurrence of term I then;

$$TF\text{-}\ IDF = TF \times IDF \quad (3.10)$$

Let TF-IDF be the word vector, PF-IDF the POS tag Vector, and SF-IDF the Sentiment Polarity Vector

The Hybrid Vector **V** will be given as;

$$\boldsymbol{V} = (TF - IDF) + (PF - IDF) + (SF - IDF)$$

Thus the Hybrid Vector becomes;

$$\boldsymbol{V} = (TF(t) \times IDF) + (PF(tp) \times IDF) + (SF(ts) \times IDF) \quad (3.11)$$

Using equation (3.10), the word Vector will be given as;

$$TF(t) \times IDF = \frac{N(t_i,d)}{S} \times \log(\frac{N}{N(t_i)}) \quad (3.12)$$

Where;

N is the number of documents (social media reviews) in the corpus;

$N(t_i,d)$ is the frequency of the term t_i in a social media review d;

S is the total number of words in a social media review d is the words in the selected N-gram

$N(t_i)$ is the number of documents that have the term t_i

Similarly, the POS tag vector will be given as;

$$PF(tp) \times IDF = \frac{N(t_p,d)}{S} \times \log(\frac{N}{N(t_p)}) \quad (3.13)$$

Where;

N is the number of documents (social media reviews) in the corpus;

N (t_p,d) is the frequency of POS tag t_p in a social media review d;

S is the total number of POS tags in a social media review d is the words in the selected N-gram

N (t_p) is the number of documents that have the POS tag t_p

Also, the sentiment polarity vector will be given as;

$$SF(ts) \times IDF = \frac{N(t_s,d)}{S} \times \log(\frac{N}{N(t_s)}) \quad (3.14)$$

Where;

N is the number of documents (social media reviews) in the corpus;

N (t_s, d) is the frequency of sentiment polarity t_s in a social media review d;

S is the total number of sentiment polarities in a social media review d is the number of polarities in the selected N-gram

N(t_p) is the number of documents that have the word polarity t_s

Substituting equations (3.12), (3.13) and (3.14) in equation (3.11), the combined Hybrid Vector is given as;

$$\boldsymbol{V} = \frac{N(t_i,d)}{S} \times \log\left(\frac{N}{N(t_i)}\right) + \frac{N(t_p,d)}{S} \times \log\left(\frac{N}{N(t_p)}\right) + \frac{N(t_s,d)}{S} \times \log(\frac{N}{N(t_s)}) \quad (3.15)$$

Where; S is the Size of the N-gram

Figure 3.4 shows how a sentence is converted to N-grams where N=3 and then the vector representation is built using the proposed approach.

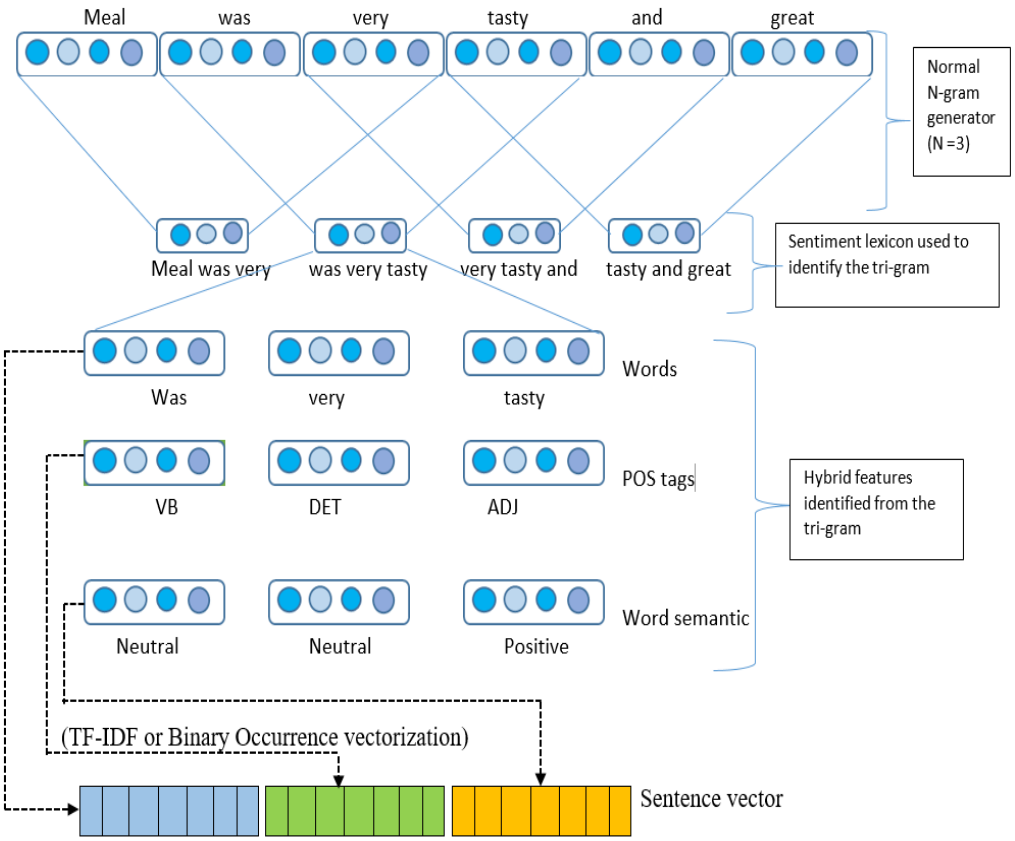


Figure 3.4: The Sentence N-Gram Vector Generator

From the sentence *N*-gram generator presented in Figure 3.4, it is clear that *N*-grams can be used to construct vector representations of documents or sentences. In the state of the art, the vector is constructed from the entire sentence's BOW or *N*-grams. In our approach, we sought to identify a specific part of the sentence where the opinion is presented so that we extract features from there. This is achieved by identifying an *N*-gram that contains a sentiment term or word from a sentiment lexicon. In this case, existing Sentiment Lexicons or customized ones are used. Further, in the approach hybrid feature vectors containing word vectors, POS tag vectors, and Semantic orientation vectors are built from the identified tri-gram.

3.5.3 Assumptions

(i) From the word tri-gram, three scenarios arise, first the sentiment word may be the first word, secondly the sentiment term may be the second word sandwiched by the other words and thirdly, the sentiment word may be the last word among the three

words. It is assumed that the order of the words has an insignificant effect on the model.

(ii) It is assumed that the objective sentence has only one opinion thus one sentiment term in one part of the sentence. Sometimes a sentence may contain more than one opinion and thus more than one sentiment term.

In the next section, we present the algorithms that model and implement the static hybrid *N*-gram2vec representation.

3.5.4 Hybrid *N*-gram2vec Algorithm

The study proposed a formal hybrid *N*-gram2vec algorithm whose characteristics and operation are described below. We define several parameters used to describe the operation of the model. Significant parameters include the size of the *N*-gram which is *N*=3, the sentiment term, the static *N*-gram words, their POS tags, and their Semantic orientations. The algorithm aims at returning the vector representation of the subjective sentence.

Definitions
Let;
D: Sentiment Lexicon
P: Part of Speech tagger
S: Subjective sentence corpus
v: Vector representation of a subjective sentence(S_i).
W_t: Sentiment term
W_1: the first word neighboring the sentiment term
W_2: the second word neighboring the sentiment term
P_t: POS tag of the Sentiment term
P_1: POS tag of the first word neighboring the sentiment term
P_2: POS tag of the second word neighboring the sentiment term
O_t: Semantic orientation of sentiment term
O_1: Semantic orientation of the first word neighboring the sentiment term
O_2: Semantic orientation of the second word neighboring the sentiment term

$\boldsymbol{v}_w$: Vector of words

$\boldsymbol{v}_p$: Vector of POS tags

$\boldsymbol{v}_s$: Vector of Sentiment orientations

We define the vector representation, $\boldsymbol{v}$, of a subjective sentence, S_i, as a concatenation of the three vectors; $\boldsymbol{v}_w$, $\boldsymbol{v}_p$, and $\boldsymbol{v}_s$. as shown below;

$$\boldsymbol{v}: \langle \boldsymbol{v}_w \,\&\, v_p \,\&\, \boldsymbol{v}_s \rangle$$

The three vectors $\boldsymbol{v}_w$, $\boldsymbol{v}_p$, and $\boldsymbol{v}_s$. are values obtained from the occurrences or TF-IDF of the words, POS tags, and Sentiment orientations.

We also define the matrix $\boldsymbol{M}^1$ as the collection of row vectors B_i as presented in equation (3.16).

$$M^1 = \begin{bmatrix} B_1 \\ \dots \\ B_n \end{bmatrix} \tag{3.16}$$

Where; n is the number of sentences in the Corpus

The algorithm listing of the sentence vector representation generation is presented in Algorithm 3.

Algorithm 3: N-gram2Vector Generation

Input: Receives preprocessed Sentence corpus(S), the Sentiment Lexicon(D)and the POS tagger (P)

Output: A Vector representation ($\boldsymbol{v}_i$) representing the subjective Sentence

START

Set the N-gram value to N=3

For each sentence($S_i \epsilon S$)word tokens;

PRINT the tri-grams;

CALL the Sentiment Lexicon(D);

FOR each tri–gram check for a semantic word;

IF a tri-gram contains a semantic word;

PRINT the tri-gram (w)

BREAK

ELSE delete the trigram

ENDIF

ENDFOR

Generate sentence vector(w_i, P, D)

FOR Each word(w_i) in tri – gram(w);

READ (w_i) into Bag of words($\boldsymbol{B_{wi}}$)

DETERMINE the POS tag (p_i)of w_i (using POS tagger P) into Bag of POS tags ($\boldsymbol{B_{pi}}$)

DETERMINE the sentiment orientation(O_i)of (w_i)(from the sentiment Lexicon)into Bag of Sentiment Orientation ($\boldsymbol{B_{oi}}$)

Update $\boldsymbol{B_i}$:‹$\boldsymbol{B_{wi}}$& $\boldsymbol{B_{pi}}$& $\boldsymbol{B_{oi}}$›

RETURN Matrix $\mathbf{M^1}$ as in equation 3.16

ENDFOR

FOR each W_i, P_i, and O_i in B_i

COMPUTE its Binary Occurrence or TF-IDF* value into vector $\boldsymbol{v_i}$:‹$\boldsymbol{v_{wi}}$ & $\boldsymbol{v_{pi}}$ & $\boldsymbol{v_{oi}}$›

ENDFOR

Return Vector $\mathbf{V_i}$.

ENDFOR

3.5.4 Dimensionality Reduction Using Maximum Relevance Minimum Redundancy (MR2)

The N-gram vector generated contains many features hence the need for dimensionality reduction. In this research, we apply and test Maximum Relevance Minimum Redundancy (MR2) as proposed by (Milos, et al., 2017) (Nagamanjula & Pethalakshmi, 2020) (Houda et al., 2014). In MRMR, Mutual Information is used since the features are discrete. First, the Mutual Information (Imin) for each feature is determined as the level of similarity between it and each of the other features calculated as shown in equation 3.17

$$Imin(f_i: f_y) = \sum_{i \in I} \sum_{y \in Y} P(i, y) \log \frac{P(i,y)}{P(i)P(y)} \quad (3.17)$$

Where f_i is the feature and f_y is the other feature being compared. This is calculated for all features in the vector. Secondly, the relevance is determined by calculating the Mutual Information between a feature and the class (Imax) as shown in equation 3.18.

$$Imax(f_i: C_i) = \sum_{fi \in I} \sum_{Ci \in I} P(f_i, C_i) \log \frac{P(f_i,C_i)}{P(f_i)P(C_i)} \quad (3.18)$$

Where f_i is the feature and C_i is the class. This is used to eliminate the irrelevant features since features with high mutual information with the class have a high relationship with the class thus more relevant.

To select the most relevant features the Mutual Information Quotient between I*min* and I*max* for each feature is noted and the features are sorted from the one with the maximum quotient as suggested by Houda et al. (Houda et al., 2014). The feature selection algorithm listing is presented in Algorithm 4.

Algorithm 4: Feature Selection Using MRMR

Input: Receives A matrix **M = (N*F + c**) where; **F** is the features of a collection of subjective Sentence corpus with **N** sentences and **c** is the class column of the sentences.

Output: A matrix (**M'= N*F + c**) containing ordered features in importance representing subjective sentences in the Corpus

START

Sort features()

For each feature $\mathbf{f_i}$ in **M**

CALCULATE its mutual information about

other features as

$$Imin(fi{:}\,fy) = \sum_{i \in I} \sum_{y \in Y} P(i,y) \log \frac{P(i,y)}{P(i)P(y)}$$

CALCULATE its mutual information about the

target opinion class as;

$$Imax(fi{:}\,Ci) = \sum\nolimits_{fi \in I} \sum\nolimits_{Ci \in I} P(fi,Ci) \log \frac{P(fi,Ci)}{P(fi)P(Ci)}$$

CALCULATE MRMR of feature $\mathbf{f_i}$ as

$$MRMR(fi) = \frac{I \max(fi{:}Ci)}{Imin(fi{:}fy)}$$

END FOR

SORT the features(f) in descending order of values of $MRMR(fi)$ and update matrix(**M'**) with sorted features

Return (**M'**)

END

From Algorithm 4, the output is a sorted matrix of features and sentence representations, the optimal features (that provide the best classification performance)

can then be selected from the entire set of features. Combining algorithms 3 and 4 gives the algorithm of the proposed approach as listed in Algorithm 5.

Algorithm 5: Proposed Text Representation Model

Input: Receives a preprocessed Sentence (S_i), the Sentiment Lexicon(D), and the POS tagger (P)

Output: A vector (**V_i'**) containing ordered features in relevance representing subjective sentence S_i

START

FOR each tokenized sentence(S_i)in Corpus(S)

Call function Generate vector representation () from Algorithm 3.

Return V_i:{V_{wi}, V_{pi}, V_{oi}}

Update matrix $\mathbf{M^2 = [V_i] = (n*F)}$; where, **F** is the features of a collection of subjective Sentence corpus with n sentences.

FOR each coordinate in $\mathbf{M^2}$

Call function Sort features () from Algorithm 4.

END FOR

Return vector(**V_i'**)the representation of sentence S_i

END FOR

END

To identify the optimal features, cross-validation is used for various values of k (number of features), and the F- measure performance of each k is noted. The optimal k (number of features) is the maximum number of features that gives the highest model performance(Y). The optimal number of features is dependent on the classifier used. The optimal features are used with a supervised learning machine classifier to classify the posts.

3.5.5 An Illustration of the Approach

Consider five social media posts that are already preprocessed and tokenized, four are already classified (training data) and the fifth is not classified (testing);

S1: wow loved this place (classified as positive)

S2: crust not good (classified as negative)

S3: The selection menu was great and were the prices (classified as positive)

S4: Worst was salmon sashimi (classified as negative)

S5: This place is worth your time let alone Vegas
(…)

The vectors from each of the first four sentences were generated. First, the tri-gram containing the sentiment term is identified from which the bag of words, POS tags, and Orientation are built as presented in Table 3.3.

Table 3.3: Tri-gram Generated from the Tokenized Sentences

S_i	Tri - gram	Bag of words (B_{wi})	Bag of POS Tags (B_{pi})	Bag of Orientation (B_{Oi})
wow **loved** this place	Wow_ **loved**_this	Wow, loved, this	ADV, VB, DT	T, P, T
crust not **good**	Crust_not_ **good**	Crust, not, good	NN, ADV, ADJ	T, N, P
The selection of the menu was **great** and the prices	*Menu_was_* ***great***	*The menu, was, great*	NN, VB, ADJ	T, T, P
The **worst** was the salmon sashimi	*Worst_was* ***_salmon***	*The worst, was, salmon*	ADJ, VB, NN	N, T, T

Where;

T – Neutral; P – Positive; N - Negative sentiment orientation

ADV – Adverb, VB – Verb, DT – Determiner, NN – Noun, ADJ – Adjective

As discussed, the Bag of words, POS tags, and sentiment orientation form the matrix from which the vector representations of the sentences are calculated. The process of

determining vector representation is shown in Table 3.4 using the binary occurrence of the identified features.

Table 3.4: Vectorization Using Binary Occurrences

B_i	wow	loved	this	ADV	VB	DT	…	…	N	P	T
Wow, loved, this, ADV, VB, DT, T, P, T	1	1	1	1	1	1	…	…	0	1	1
Crust, not, good, NN, ADV, ADJ, T, N, P	0	0	0	1	0	0	…	…	1	1	1
The menu, was, great, NN, VB, ADJ, T, T, P	0	0	0	0	1	0	…	…	0	1	1
Worst, was, salmon, ADJ, VB, NN, N, T, T	0	0	0	0	1	0	…	…	1	0	1

The features vector can also be represented using TF-IDF as shown in Table 3.5.

Table 3.5: Vectorization Using tf-idf

B_i	wow	loved	this	ADV	VB	DT	…	…	N	P	T
Wow, loved, this, ADV, VB, DT, T, P, T	1.67	1.67	1.67	1.33	1.12	1.67	…	…	0	1.12	2.00
Crust, not, good, NN, ADV, ADJ, T, N, P	0	0	0	1.33	0	0	…	…	1.33	1.12	1.00
The menu, was, great, NN, VB, ADJ, T, T, P	0	0	0	0	1.12	0	. . .	. . .	0	1.12	2.00
Worst, was, salmon, ADJ, VB, NN, N, T, T	0	0	0	0	1.12	0	…	…	1.33	0	2.00

From Table 3.4, it's clear that the number of features identified can be high leading to high feature dimensionality. The features then need to be reduced to the optimal number. In our approach, this is done by the MRMR feature selection algorithm as discussed.

3.5.6 Supervised Machine Learning Classifiers

The features extracted and selected using LeNFEM were used as input into machine learning classifiers. The classifier was first trained and then tested using the datasets described in section 3.1. In the experiments, we proposed and investigated the use of a Decision Tree, Naïve Bayes, K-nearest neighbor (KNN), or Support Vector Machine (SVM) with the resultant vector obtained as discussed in section 3.5.6.

3.6 Text Representation Using Sentiment Lexicon-Augmented Word Embeddings

In deep learning and as discussed in section 2.5.6, there are several word embeddings used in text representation for sentiment analysis. BERT, unlike other word embedding algorithms, can effectively read a series of words in either direction of the input text, and since it uses the attention mechanism to assign a word, its vector depends on the surrounding words, and it is efficient in word vectorization (Yang, 2022). Although BERT considers the context of a word when assigning the vector, it does so for all the words in the input text, which leads to a resultant vector with high dimensionality. Second, word vectors built from BERT do not contain semantic information, which is critical in sentiment analysis. On the other hand, we realized that sentiment lexicon can be used to identify sentiment words in a text and assign specific sentiment polarity to the words. However, the sentiment lexicon cannot generate representative word vectors, hence leading to high data sparseness. Thus, to improve sentiment classification, we proposed and investigated the Lexicon-enhanced Bidirectional Encoder Representations from the Transformers (LeBERT) model. The model combines sentiment lexicon, *N*-grams, and BERT word embedding algorithm.

The design idea of the LeBERT model is to first use *N*-grams to split the input text into sections, and then use a sentiment lexicon to identify a section or sections that contain a sentiment word. It is worth noting that text reviews, such as social media posts, contain short text, and characteristically, semantic features in short texts are concentrated in a certain part (Yang, 2022). Thus, extracting features from such parts will lead to an efficient and effective text representation. The words of the identified section(s) are then converted into a vector by BERT. The output word vector is then used as the input into a CNN model with a fully connected layer where features from the vector are obtained. The features extracted are then integrated by the dense output layer, and finally, the sentiment class of the text is performed by a SoftMax classifier. The architecture of the proposed LeBERT model is shown in Figure 3.4.

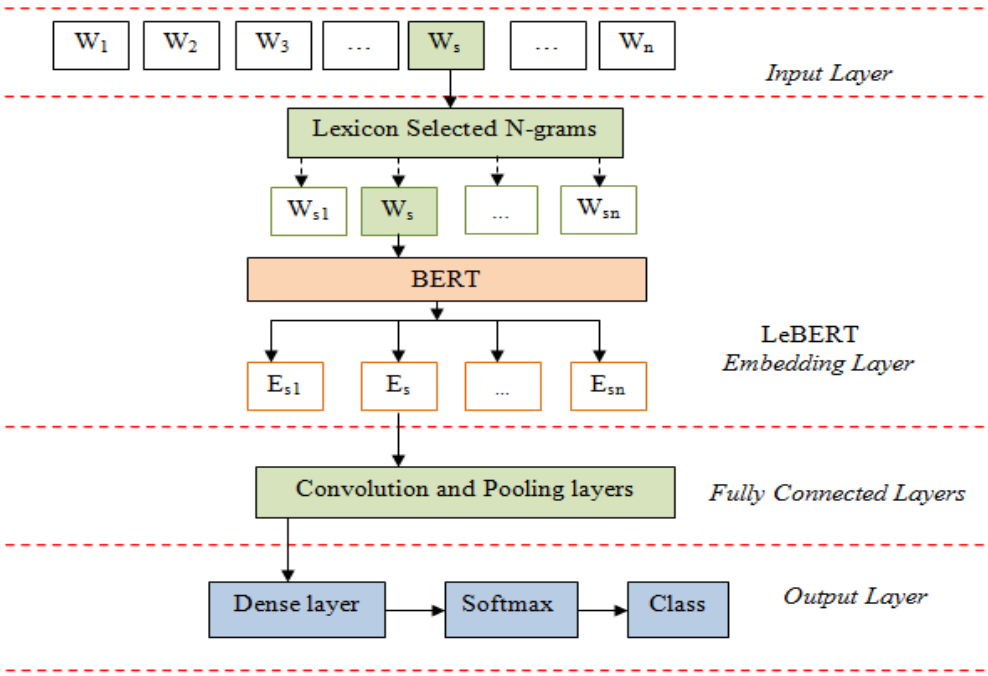


Figure 3.5: Architecture of the Proposed Sentiment Analysis Model

As shown in Figure 3.4, the sentiment lexicon, *N*-grams, and BERT algorithm are used in the embedding layer to build the word vector. The overall sentiment analysis model using the LeBERT model is presented in Figure 3.5.

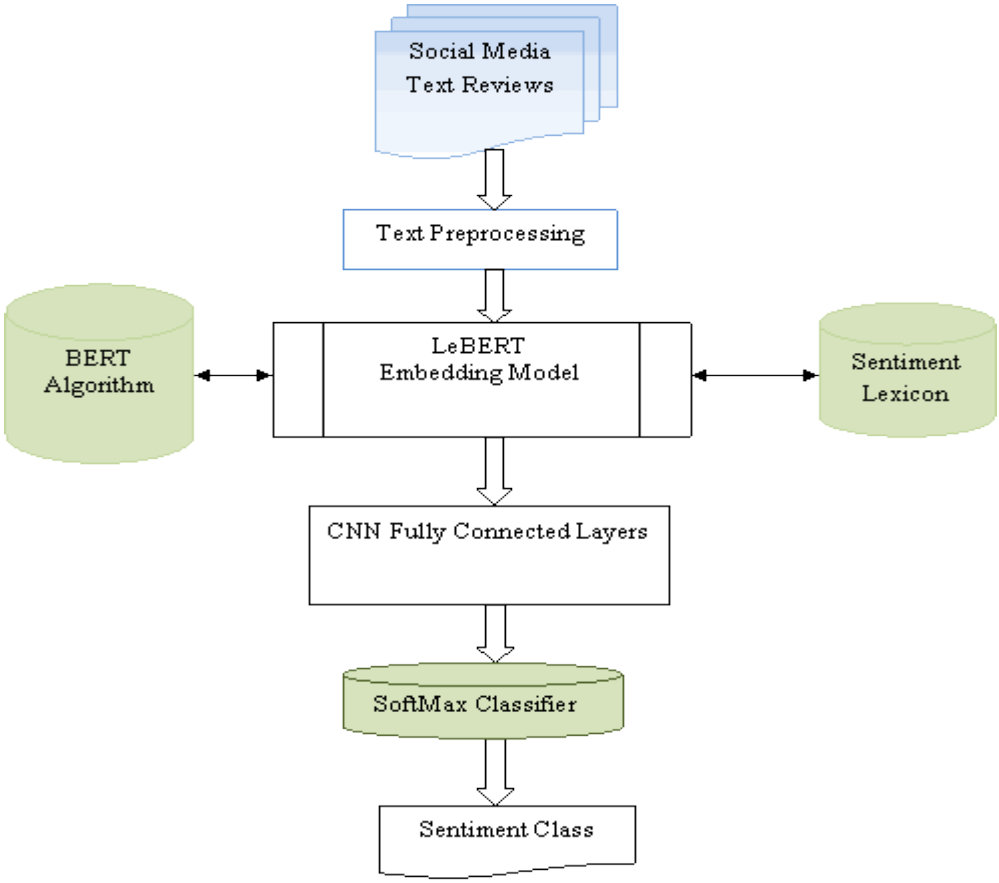


Figure 3.6: Sentiment Analysis Model Using the LeBERT Model

3.5.7 LeBERT Embedding

There are currently two common methods used to build text vectors for sentiment analysis: word-embedding-based methods and lexicon-based methods. In our proposed model, we sought to utilize both methods through *N*-grams. The sentiment lexicon is used to identify the word *N*-grams containing a sentiment word, and then the vector from the *N*-gram words using the BERT word embedding model is used.

To build the vector, we first generate word *N*-grams from the sentences. *N*-gram is a combination of words from a sentence, which forms a Markovian process. Normally, this is used to predict the next word in a sequence of words. Further, the Markovian process also generates the co-occurrence of words, which is a key aspect in influencing sentiment in a text. In this case, we use *N*-gram sequences to partition a sentence into various sections that represent the entire text, such as an online review or a sentence. This is because *N*-grams present the co-occurrence of words in a text in a more

comprehensive manner than a mere bag of words (BoW). The size of the partition depends on the value of N.

For instance, if we consider a sentence S given as:

$$S = \{w1, w2, w3, w4, w5, \dots, \dots, \dots wn\} \quad (3.19)$$

Where w_i are words.

For various values of N, we have;

N = 1, the set of N-grams $N_1 = \{w_1, w_2, w_3, \dots, w_n\}$

N = 2, the set of N-grams $N_2 = \{w_1\_w_2, w_2\_w_3, w_3\_w_4, \dots, w_{n-1}\_w_n\}$

N = 3, the set of N-grams $N_3 = \{w_1\_w_2\_w_3, w_2\_w_3\_w_4, w_3\_w_4\_w_5, \dots, w_{n-2}\_w_{n-1}\_w_n\}$

The fundamental idea is that, with the set of *N*-grams, it is possible to select a section of the entire input text. This ensures that we use the most significant words when building text vectors for sentiment analysis. Once the *N*-gram(s) are identified from the text, it is then reverted to a bag of words. Each word is then converted into a vector using the BERT word-embedding algorithm.

3.5.8 The LeBERT Embedding Algorithm

Let *L*: sentiment lexicon; *C*: a corpus of subjective user reviews (R_i); $\boldsymbol{V_i}$: vector representation of a subjective review (R_i); W_t: sentiment term; W_1: the first word neighboring the sentiment term; and W_2: the second word neighboring the sentiment term.

We define the text vector, $\boldsymbol{v_i}$, of a subjective review, R_i, as the vector originating from a selected section of the review S_i using sentiment lexicon and BERT word embedding model (***Be***). The algorithm listing of the sentence vector representation generation is presented in Table 3.6.

Table 3.6: Algorithm Listing of the Contextualized Text Vector Generation

Algorithm 6: Contextualized Text Vector Generation

Inputs:
$R_i = \{w_1, w_2, \ldots\ldots, w_n\}$, input review containing n words
L = sentiment lexicon
Be = BERT word-embedding model
Output: Contextualized Text Vector ($\boldsymbol{v_i}$), representing the subjective user review
START
Set the N-gram value to N = 3
For each review ($R_i \epsilon C$) with n-word tokens
  PRINT the word trigrams;
  Call the sentiment lexicon (L)
  FOR each trigram check for a sentiment word;
    IF a trigram contains a sentiment word **THEN**
      PRINT the trigram words (w_1, w_t, w_2)
      ELSE delete the trigram
    ENDIF
  END
  Generate section vector( )
  FOR Each word (w_1, w_t, and w_2) in the trigram
    READ (w_i) into gag of words ($\boldsymbol{B_{wi}}$)
    Call the pre-trained word-embedding (***Be***)
    Calculate the word vector ($\boldsymbol{w_{vi}}$)
  END
  Update vector $\boldsymbol{V_i}$**:‹**$\boldsymbol{w_{v1}}$ **and** $\boldsymbol{w_{vt}}$ **and** $\boldsymbol{w_{v2}}$**›**
END
Return Vector $\boldsymbol{V_i}$**.**

3.5.9 The CNN Layer

The CNN deep learning model was used as the classifier, which uses the resultant vector from LeBERT embedding as input and gives the sentiment class as the output. CNNs are specialized types of artificial neural networks, which are capable of outperforming the common machine learning algorithms in supervised learning tasks. CNNs' main function is to identify and learn the information characteristic patterns through the use of convolution layers and thus facilitate the classification of the objects. The CNN model is presented in Figure 3.6. Using the convolution kernels (windows) and the nonlinear function (filter), feature maps are obtained. A pooling operation is then applied to the feature maps to select the optimal features. The dense output layer then classifies the optimal features using the softmax activation function (which uses probability) into a positive or a negative class.

In this phase, the CNN takes the hybrid vector space as input. As discussed the vector space comprises words, POS tags, and Semantic orientations of words identified from a specific part of a sentence using N-grams. We then further seek to select some features from the so many features obtained in the vector space since some may not be important in the classification of the sentences. To achieve this we propose and investigate the use of convolution and pooling operations of deep learning algorithms. Figure 3.9 shows the architecture of the convolutional and pooling layers. For instance, if we have an S sentence and the vector of features is of k features then we will have an S*k matrix as input.

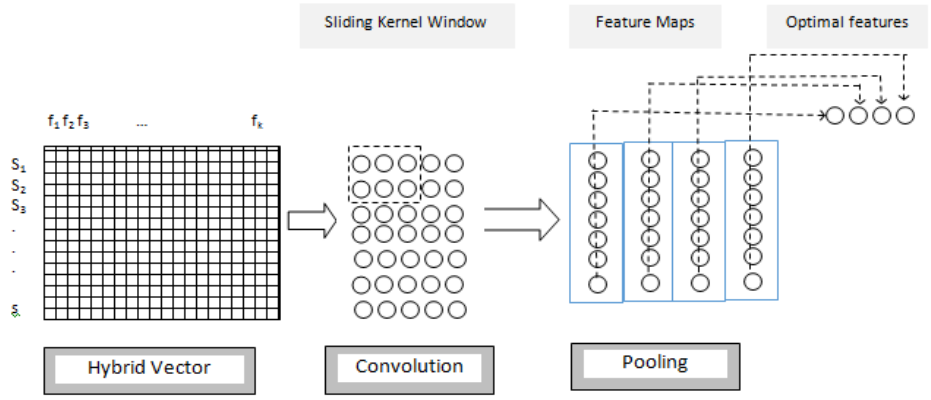


Figure 3.7: Feature Vector Convolution and Pooling

Convolutional Neural Network (CNN) is a class of deep learning models that uses multilayer perceptron to require minimal preprocessing. CNN was initially developed for image classification and computer vision problems but recently it has been applied in text classification in which it is applied on a one-dimensional vector representation of the text as the input. In this research, the convolution operation involves selecting using convolution kernels also referred to as windows. The size of the window is normally $y \times x$. In this case, since the features have been extracted in the previous phase the window considered is of form $\times x$. The convolution kernel of the i-th convolution is thus $w \in R^{y \times x}$. The size of the convolution kernel x controls the number of features in each convolution. The main resolution of using convolution is to apply a nonlinear function to each feature window of size x of the input feature vector. The nonlinear function is referred to as a filter. After the filter has selected a

value from the window the feature window shifts depending on the strides. From the convolution windows, several feature maps with one column each will be obtained depending on the number of features. Formally, the convolution is shown in equation 3.20.

$$c_{i1} = f(xi + b) \tag{3.20}$$

f is the activation function that is used to generate the semantic vector C_i after transforming the input xi of the sentence vector. There are several activation functions but for convolution, we use the Rectified Linear Unit function (RELU) since it is easier to train and leads to better performance compared to other functions. C_{i1} is then the input sent to the pooling layer of the model. To obtain the most important features (optimal features) a pooling operation is done on the feature maps since the convolution layer is connected to the pooling layer. Equation 3.21 shows the mathematical representation of the pooling operation.

$$C_{i2} = max_{i=1}^{n} C_{i1} \tag{3.21}$$

The pooling operation is used to identify the optimal features from the feature maps obtained from the convolution operation. There are two common pooling operations which are maximum and mean pooling. In our model, we use maximum pooling operation since we are interested in obtaining features with the highest amount of information that can be used in the classification of the sentence. Therefore from the feature map, the highest value corresponds to the best feature among those selected. The collection of best features is thus the output of the pooling operation which is the set of sampled features. This also attests to the reason why the pooling operation is also called down sampling.

The output layer consists of a fully connected layer that uses the softmax activation function. It is similar in operation to a neural network and the convolutional layer but its input is a highly refined vector representing a sentence. Thus we represent it as shown in equation 3.22

$$c_{i3} = f(c_{i2} + b) \tag{3.22}$$

Here we use the softmax activation function to optimize the model since it is effective in solving the vanishing gradient problem and speeding up the model's convergence [24] during model training. Softmax activation function consists of multiple neurons and its output is the maximum of the neuron activation. Finally, the sentiment polarities of the text are calculated by the softmax function which uses probabilities as shown in equation 3.23.

$$p_i = \frac{\exp(c_{i3})}{\sum_{k=1}^{n} \exp(c_{k3})} \tag{3.23}$$

The class with the highest probability becomes the sentiment class of the input sentence.

3.6 Model Performance Evaluation

To evaluate the performance of the sentence-level text classification model developed, the performance results of the supervised machine learning algorithms used to simulate the model were plotted in a confusion matrix. A confusion matrix is a square that shows the true classes along the horizontal and the predicted classes along the vertical (Egejuru et al., 2017). The Square matrix is of order ($Ci * C_i$) where C_i is the number of classes in the prediction model. The performance of the model was tested with two classes, the negative and positive classes.

3.7 Conclusion

This chapter described the methodology used to achieve the objectives of the research. The methodology involved descriptions of sentence text features and how experimental matrices were developed. The procedure for developing and validating the proposed sentiment analysis model was also described.

CHAPTER FOUR

RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter presents the research findings and results of the methods that were used for the development of the text representation model for sentiment analysis. The results include data collection, statistical analysis, model formulation, and simulation results from Rapid Miner Studio software which provided the model performance evaluation results.

The results were presented based on the study objectives in tables and graphs.

4.2 Investigation of Text Representation Using Bag of Words, Sentiment Lexicon, N-grams and POS Tagging

The experimental results obtained from the various experiments carried out in the study to investigate the existing text representation techniques and the proposed model was presented in the subsequent sections.

4.2.1 Experiment Results 1: Determination of Number of Variables

The dimension of the text vector built from each technique was determined to establish the number of variables in the vector. The experiment was done using the normal Bag of Words (BoW), normal N-Grams up to N=3, and the words' polarities using the Yelp dataset. The number of variables ranged from 4 to about 9000 variables as shown in table 4.1.

Table 4.1: Number of Variables

Technique	Number of variables
1 N-gram (BoW)	1783
2 N-grams	6524
3 N-grams	8626
Words' Polarities	4

From Table 4.1, N-grams generated a very high (1783 to 8626) number of features. However, the word polarities had only four (4) variables which were positive, neutral, negative, and no label. The number of features obtained using the proposed technique was also investigated.

The optimal number of variables (features) for the proposed hybrid text representation technique with feature selection (LeNFEM) was determined. Since the MRMR algorithm sorts the features in order of importance, the performance varies as the number of selected features increases. We present the results for the first 200 features from the total 973 features since the performance seemed to stabilize for all classifiers used from 50 features and reduced to 200 features. Figures 4.1 and 4.2 show the variation of the features using Binary Occurrence vectorization and TF-IDF vectorization algorithms respectively.

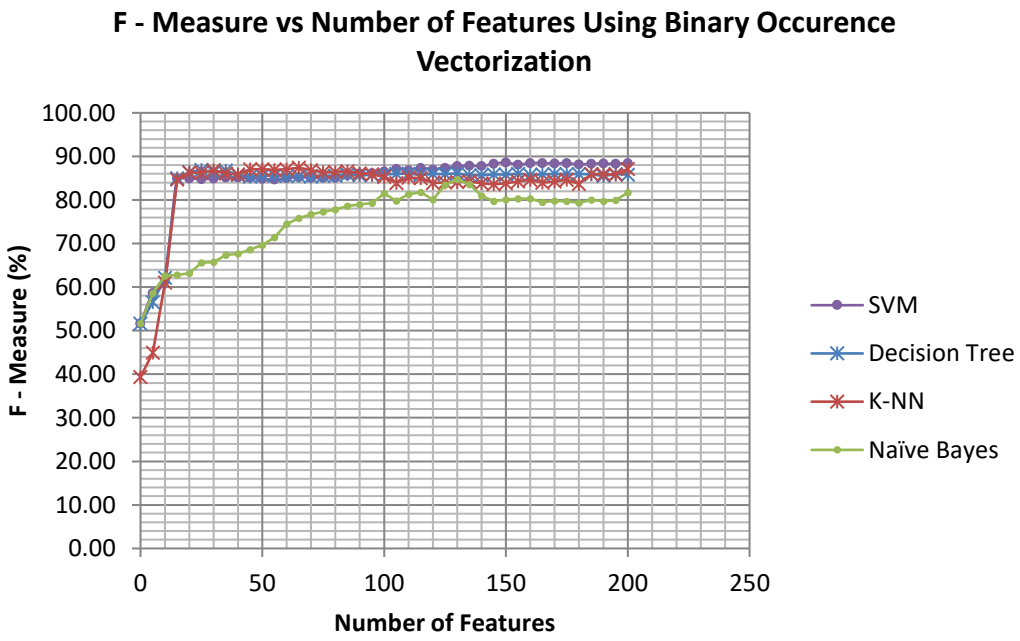


Figure 4.1: Model Performance against Number of Features Using Binary Occurrence

From Figure 4.1, it's clear that as the number of features selected increases, the classification performance increases to a point where it stabilizes. The highest performance for the Decision Tree was achieved with 25 features, K-NN (K=5) with

65 features, Naïve Bayes with 130 features, and for the Support vector machine was 150 features.

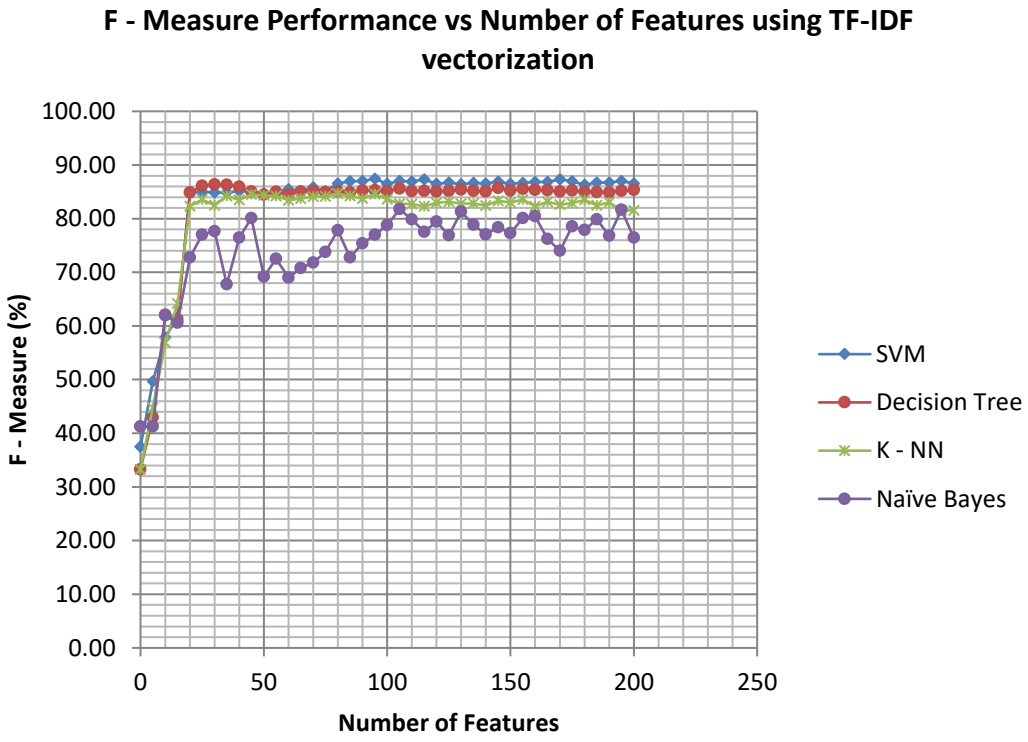


Figure 4.2: Model Performance against the Number of Features Using TF-IDF

From Figure 4.2, as the number of features selected increased, the classification performance increased to a point where it stabilized. The highest performance for the Decision Tree was achieved with 30 features, K-NN (K=5) with 80 features, Naïve Bayes with 105 features, and for the Support vector machine 95 features. The number of features used in the experiments is summarized in Table 4.2 and Table 4.3 for the binary occurrence vector and TF-IDF vector respectively.

Table 4.2: Number of Features in the Binary Occurrence Vector

Technique	Number of Features Used			
	Naïve Bayes	**K-NN**	**Decision Tree**	**SVM**
Bag of Words(BoW)	1783	1783	1783	1783
3 N-grams	8626	8626	8626	8626
Word's Polarity	4	4	4	4
LeNFEM	130	65	25	150

From Table 4.2, it is clear that the number of features obtained using LeNFEM and binary occurrence reduced significantly. For instance, using Naïve Bayes the features were 1783 from Bag of Words but reduced to 130 features using the proposed LeNFEM model.

Table 4.3: Number of Features in the TF-IDF Vector

Technique	Number of Features Used			
	Naïve Bayes	**K-NN**	**Decision Tree**	**SVM**
Bag of Words(BoW)	1682	1682	1682	1682
3 N-grams	8522	8522	8522	8522
Word's Polarity	4	4	4	4
LeNFEM	105	80	30	95

From Table 4.3, it is clear that the number of features obtained using the LeNFEM and TF-IDF vectorization method reduced significantly. For instance, using Naïve Bayes the features were 1682 from Bag of Words but reduced to 105 features using the proposed LeNFEM model. The words' polarity features were four (4) since we had only positive, negative, neutral, and missing polarities.

4.2.2 Experiment Results 2: Proposed Model (LeNFEM) Validation Results

Details of the F-measure performance for the proposed model (LeNFEM) in comparison with the baseline models for each machine learning classifier used were presented in Table 4.4 and Table 4.5.

Table 4.4: F-Measure Results for Binary Occurrence Vector

Text Representation	F-measure (%)			
	Naïve Bayes	**K-NN**	**Decision Tree**	**Support Vector Machines**
Bag of Words	72.24+/-0.520	67.40+/-0.572	70.22+/-0.531	75. 93+/-0.545
3 N-grams	70.72+/-0.551	69.88+/-0.621	70.08+/-0.574	65.81+/-0.560
Bag of words' polarities	82.34+/-0.651	83.55+/-0.542	82.54+/-0.341	83.44+/-0.359
LeNFEM	84. 68+/- 0.671	87.43+/- 0.539	86.91+/- 0.331	**88.64+/- 0.362**

Where; F (%) is the F-measure +/- the standard deviation

From Table 4.4, the proposed LeNFEM approach outperformed the baseline approaches. Using the bag of words' polarities only four features were identified; negative, positive, neutral, or a missing value. It's also worth noting that of the four classifiers used, the support vector machine classifier produced the highest F-measure score of 88.64%.

Table 4.5: F-Measure Results Using TF-IDF Vector

Text Representation	F-measure (%)			
	Naïve Bayes	**K – NN**	**Decision Tree**	**SVM**
Bag of Words	65.64+/- 0.532	59.38+/-0.602	68.85+/- 0.576	66.62+/-0.556
3 N – gram	67.25+/-0.645	64.24+/-0.590	65.92+/- 0.625	62.48+/-0.634
Bag of words' polarities	78.56+/-0.632	82.75+/-0.607	82.36+/-0.334	81.46+/-0.536
LeNFEM	81.75+/- 0.358	84.71+/- 0.350	86.37+/- 0.356	**87.45+/- 0.330**

Where; F (%) is the F-measure +/- the standard deviation

From Table 4.5, the proposed LeNFEM approach outperformed the baseline approaches considered. It's also worth noting that of the four classifiers used, the support vector machine classifier produced the highest F-measure score of 87.45%. These results are similar to the results obtained by Liu et al. (2022) in their study of combining TF-IDF and dictionary-based methods. However, in their study, the highest F-measure was 86.1% which is slightly lower compared to the results in this study. The slight margin could be attributed to the use of the hybrid vector in this study since in their study they used the words only.

We further used a tailed Wilcoxon Sign-Rank test at a significance level of 0.05 to ascertain whether the LeNFEM approach performed better than the baseline approaches. We paired all the F-measure results from the classifiers for the approaches. We first compared the F-measure results of LeNFEM with the Bag of Words approach which gave W-value: 0, Mean Difference: -19.15, Sum of positive ranks: 0, Sum of negative ranks: 36, Z-value: -2.5205 and Sample Size (N): 8. Therefore, since the value of W is 0 which is less than the critical value for $W_{critical}$ at $N = 8$ ($p < .05$) which is 3, the proposed approach performed better than the Bag of Words approach. After comparing the F measure of the proposed LeNFEM approach with 3N-gram, the Wilcoxon Sign Rank test gave; a W-value: 0, Mean Difference: -20.38, Sum of positive ranks: 0, Sum of negative ranks: 36, Z-value: -2.5205, Sample Size (N): 8. Therefore, we use W value since the sample of the results was less than ten(N =8) and because the value of W is 0 which is less than the critical value for $W_{critical}$ at $N = 8$ ($p < .05$) which is 3, the proposed approach produced better performance than the 3N-gram approach.

Similarly, the F measure performance of the proposed LeNFEM approach and the bag of words' polarities approach were compared, the Wilcoxon Sign Rank test gave; a W-value: 0, Mean Difference: -18.58, Sum of positive ranks: 0, Sum of negative ranks: 36, Z-value: -2.5205, Sample Size (N): 8. Therefore, we use W value since the sample of the results was less than ten(N =8) and because the value of W is 0 which is less than the critical value for $W_{critical}$ at $N = 8$ ($p < .05$) which is 3, the proposed approach produced better performance than the bag of words' polarities approach.

4.2.3 Experiment Results 3: Comparison of Binary Occurrence and TF-IDF Vectorization

From section 4.2.2, it was evident that the proposed approach performed better than the baseline approaches considered. We then sought to investigate the performance of the binary occurrence and TF-IDF vectorization algorithms. This was done by comparing their performance in the four machine learning classifiers using the proposed approach. The results are shown in Table 4.6.

Table 4.6: Comparison of F- measure for the Text Vectorization Algorithms Using the Proposed Approach

Vectorization Algorithm	Machine Learning Classifier			
	Naïve Bayes	**K - NN**	**Decision Tree**	**SVM**
Binary Occurrences	84.68+/- 0.671	87.43+/- 0.539	86.91+/- 0.331	**88.64**+/- 0.362
TF-IDF	81.75+/- 0.358	84.71+/- 0.350	86.71+/- 0.356	87.45+/- 0.330

When comparing the performance of the two vectorization algorithms, the binary occurrences algorithm was found to be superior to the TF-IDF algorithm, as it's evident in Table 4.6, with the highest F-Measure (88.64%) using the support vector machine classifier. It is also worth noting that even though the TF-IDF algorithm presented lower performance results than that of the binary occurrences algorithm, it also showed promising results of greater than 80%. Musleh et al. (2023), in their study, proposed an NLP-based model to classify Arabic comments as positive or negative and investigated the effect of N-grams, TF-IDF, and word count vectorizers on Naïve Bayes (NB), SVM, Decision Tree (DT), KNN, Random Forest (RF) and Logistic Regression (LR) and reported high performance in all the classifiers using word count vectorizer without TF-IDF. The difference in their findings with this study is that the NB classifier produced better results with TF-IDF than with the word count vectorizer. In their study, SVM had the highest performance without TF-IDF vectorization which is similar to the findings of this study.

4.2.4 Experiment Results 4: Investigation of Feature Selection on Proposed Model

The performance of the MRMR feature selection algorithm in the proposed approach was investigated in this experiment. Since it was evident that the binary occurrences algorithm performed better than the TF-IDF algorithm, the performance of the feature selection algorithm was evaluated by comparing the F-measure performance of the four machine learning classifiers using the proposed approach with and without feature selection algorithm. The results obtained are shown in Table 4.7.

Table 4.7: Analysis of Feature Selection Algorithm

Machine Learning Classifier	Proposed LeNFEM (with MRMR feature selection) F-measure (%)	LeNFEM (without MRMR feature selection) F-measure (%)
Naïve Bayes	84.68+/- 0.671	82.52 +/- 0.364
K - NN	87.43+/- 0.539	82.65 +/- 0.336
Decision Tree	86.91+/- 0.331	81.11 +/- 0.525
Support Vector Machines	**88.64**+/- 0.362	82.76 +/- 0.351

In Table 4.7, a comparative analysis is made for feature selection performance in terms of F-measure. The LeNFEM approach achieves an F-measure of 88.64 % with the feature selection process. On the other hand, the LeNFEM approach without feature selection achieved only 82.76%. From the results, LeNFEM with feature selection achieved better performance.

4.2.5 Experiment Results 5: Performance Analysis of the Proposed Approach with other Datasets

In this experiment, the performance of the proposed LeNFEM (with MRMR feature selection) approach using the binary occurrences algorithm was investigated with the products and movie datasets. The performance was evaluated by comparing the F-measure performance of the four machine learning classifiers using the proposed approach and the three datasets. The results are shown in Table 4.8.

Table 4.8: F-Measure Performance Analysis of the Proposed Approach with Various Datasets

Machine Learning Classifier F-measure performance (%)	Proposed LeNFEM (with MRMR feature selection) Yelp Restaurant Dataset	Amazon's Products Dataset	IMDb's Movies Dataset
Naïve Bayes	84.68+/- 0.671	83.84+/- 0.604	85.04+/- 0.562
K - NN	87.43+/- 0.539	88.12+/- 0.341	87.82+/- 0.354
Decision Tree	86.91+/- 0.331	86.72+/- 0.323	87.02+/- 0.507
SVM	88.64+/- 0.362	**90.15**+/- 0.512	89.01+/- 0.366

In Table 4.8, a comparative analysis is presented for the F-measure performance of the machine learning classifiers using the proposed approach with various datasets. The LeNFEM approach achieves an F-measure of 88.64 % with Yelp Restaurant Dataset, 90.15% with Amazon's Products Dataset, and 89.01% with Imdb's Movies Dataset.

From the results, the proposed approach has very good performance with all the datasets used with Amazon's Products dataset having the highest F-measure (90.15%) for the support Vector Machine classifier.

4.3 Investigation of Text Representation Using Sentiment Lexicon with Word Embeddings and Deep Learning

This section describes the results obtained from the experiments of determining the effect of using a sentiment lexicon with word embeddings and deep learning models.

4.3.1 Experiment Results 6: N-Grams Ablation Study Results

To verify the effectiveness of using sentiment lexicon with word embedding algorithms to generate word vectors, we first experimented to study the effect of the size of N-grams on the sentiment lexicon-enhanced BERT model with CNN. In the experiment, the restaurant review datasets were used. As discussed in Chapter Three, the input text was first converted into N-grams. A Sentiment Lexicon was then used to select the N-grams containing a sentiment term. The selected N-grams were then used as words for the word embedding layer using the BERT model. The experimental results of N = 1, 2, 3, 4 and all words were as shown in Table 4.9.

Table 4.9: Size of N-grams Analysis Results

Size of N-Grams	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
Sentiment Lexicon Enhanced BERT with CNN				
N = 1	65.02	65.02	65.15	65.08
N = 2	79.45	79.50	80.04	79.77
N = 3	88.20	88.45	89.01	**88.73**
N = 4	87.65	87.65	87.80	87.72
All words	84.00	84.00	84.20	84.10
Sentiment Lexicon Enhanced Glove with CNN				
N = 1	64.82	64.82	65.10	64.96
N = 2	78.05	78.05	79.80	78.92
N = 3	86.42	86.50	87.02	86.76
N = 4	85.85	85.75	86.50	86.12
All words	82.90	82.90	83.02	82.96
Sentiment Lexicon Enhanced Word2Vec with CNN				
N = 1	64.90	64.85	65.15	65.00
N = 2	78.75	78.80	79.78	79.29
N = 3	87.80	87.80	88.85	88.32
N = 4	85.60	85.60	86.82	86.21
All words	83.08	83.08	83.80	83.44

From Table 4.9, we generated N-grams up to N = 4 due to computational resources. For N = 1, it implies that, for each sentence, only one word was used, which was chosen by the sentiment lexicon. The category of 'All words' implies that the sentiment lexicon was not applied to the input text to select some words, hence, this reverts to the original word embedding model. The results indicate a low performance in all the word embedding models when N=1. This is because one word cannot represent the sentiment of the entire text. The highest model performance was obtained at N = 3 across all the three word embedding models with the BERT model producing the highest F-measure of 88.73%. The results thus indicated that N = 3 is an ideal size of N-gram for the proposed model. It is also worth noting that the sentiment lexicon-enhanced BERT model outperformed the other word embedding models.

4.3.2 Experiment Results 7: Effect of Sentiment Lexicon on the Text Data

This experiment sought to test the effect of using Sentiment Lexicon on the input text data. The shape of the Yelp dataset (restaurant reviews) before and after using the sentiment lexicon was analyzed. Table 4.10 presents the details of the text data.

Table 4.10: Details of the Text Data Before and after Using Sentiment Lexicon

Text Data Item	Before Using the sentiment lexicon	After Using the sentiment lexicon
Characters(no spaces)	46,744	14,182
Characters(with spaces)	56,616	19,212
Number of words	10,863	2989
Number of paragraphs	996	996
Average Number of words per Post/paragraph	11	3

From Table 4.10, it was evident that the application of the sentiment lexicon to select a section of the input text significantly reduced the size of the input text. Although the number of posts or paragraphs remained the same, the shape of the input text changed from 11 rows to 3 rows, which, in turn, would reduce the computation time for the model. We then designed and performed experiments with deep learning CNN to evaluate how the Sentiment Lexicon Enhanced embedding models would perform in sentiment analysis. CNN was used since studies such as one done by Elhassan et al. (2023) have shown that CNN performs better in sentiment analysis than other deep learning models such as long short-term memory (LSTM), and a hybrid CNN-LSTM.

4.3.3 Experiment Results 8: Proposed Sentiment Lexicon-Enhanced Word Embedding Model Validation Results

The experiment was carried out to validate the performance of the proposed Sentiment Lexicon-Enhanced Word Embedding model in terms of accuracy, recall, precision, and F-measure of the CNN on the three discussed datasets. The proposed word embedding models were; the Lexicon-Enhanced Glove word embedding model (Le-Glove), Lexicon-Enhanced Word2vec word embedding model (Le-Word2Vec), and Lexicon-Enhanced BERT word embedding model (LeBERT). Original Glove, Word2Vec, and BERT models were used as baseline word embedding models. In this experiment, tri-grams (N = 3) were used. Table 4.11 presents the results obtained from the three datasets.

Table 4.11: Performance Analysis of Sentiment Lexicon Enhanced Models

Dataset	Embedding Model	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
Yelp	Glove	78.50	78.56	78.70	78.63
	Le-Glove	81.50	82.00	83.01	82.50
	Word2Vec	75.50	75.5	75.80	75.65
	Le-Word2Vec	82.40	82.45	83.15	82.80
	BERT	84.00	84.00	84.20	84.10
	LeBERT	88.20	88.45	89.01	**88.73**
Imdb	Glove	79.50	79.50	80.10	79.80
	Le-Glove	82.50	82.70	83.25	82.97
	Word2Vec	77.45	77.46	78.01	77.73
	LeWord2Vec	83.00	83.02	83.42	83.22
	BERT	84.01	84.08	84.63	84.35
	LeBERT	86.10	86.71	87.00	86.85
Amazon	Glove	79.00	79.00	79.65	79.32
	Le-Glove	79.60	80.00	80.45	80.22
	Word2Vec	79.50	79.50	80.25	79.87
	Le-Word2Vec	81.50	81.50	82.05	81.77
	BERT	81.72	81.75	82.04	81.89
	LeBERT	82.40	82.40	82.64	82.52

From Table 4.11, the LeBERT model had the highest F-measure (88.73) in the Yelp Dataset. Generally, the performance of the three sentiment lexicon-enhanced models was better compared to the NLP model's performance. It was observed that BERT had the highest performance in all the datasets. This study confirms the ability of the BERT model to generate better vectors that are rich in context information as hypothesized by other studies such as Delvin et al. (2018). The performance of the models in each of the datasets was also analyzed. Figures 4.3, 4.4, and 4.5 show the performance results of the model on restaurant reviews (Yelp Dataset), movie reviews (Imdb Dataset), and product reviews (Amazon Dataset) respectively.

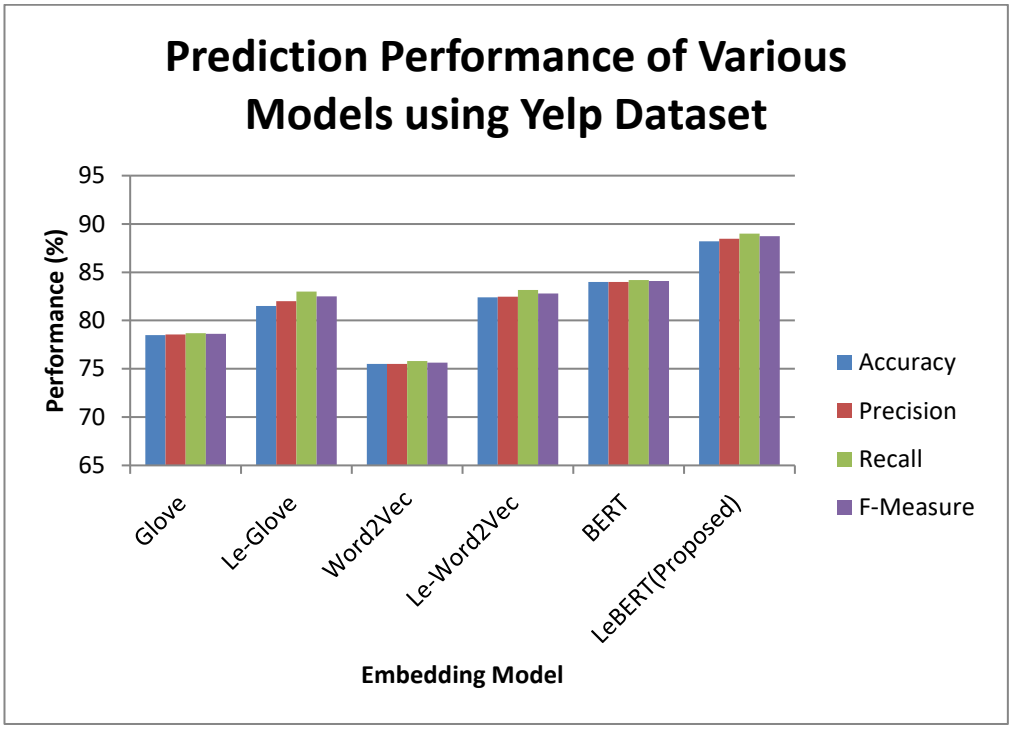


Figure 4.3: Validation Results of Embedding Models Using Yelp Dataset

From Figure 4.3, the proposed LeBERT model had the highest Accuracy (88.20%), Precision (88.45%), recall (89.01%), and F-measure (88.73%) compared to all the other models. It is worth noting that using a lexicon in all the embedding algorithms improved the performance of the model using the Yelp dataset. This is an indication that lexicon-selected sections of the input text had a positive impact on the performance of the sentiment analysis model on the Yelp dataset.

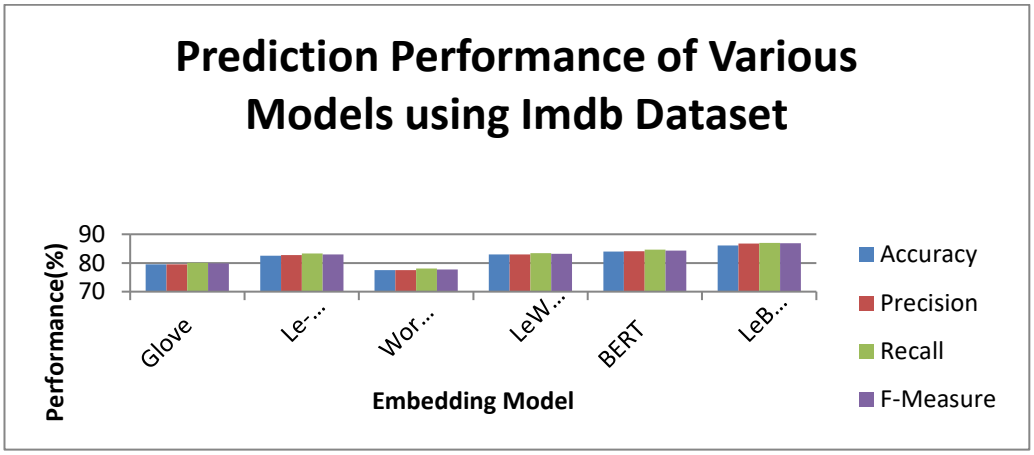


Figure 4.4: Validation Results of Embedding Models Using Imdb Dataset

From Figure 4.4, it was observed that the proposed LeBERT model had the highest Accuracy (86.1%), Precision (86.71%), recall (87%), and F-measure (86.85%) compared to all the other models. It is also worth noting that the use of lexicon in all the embedding algorithms improved the performance of the model. This is an indication that lexicon-selected sections of the input text had a positive impact on the performance of the sentiment analysis models on the Imdb dataset.

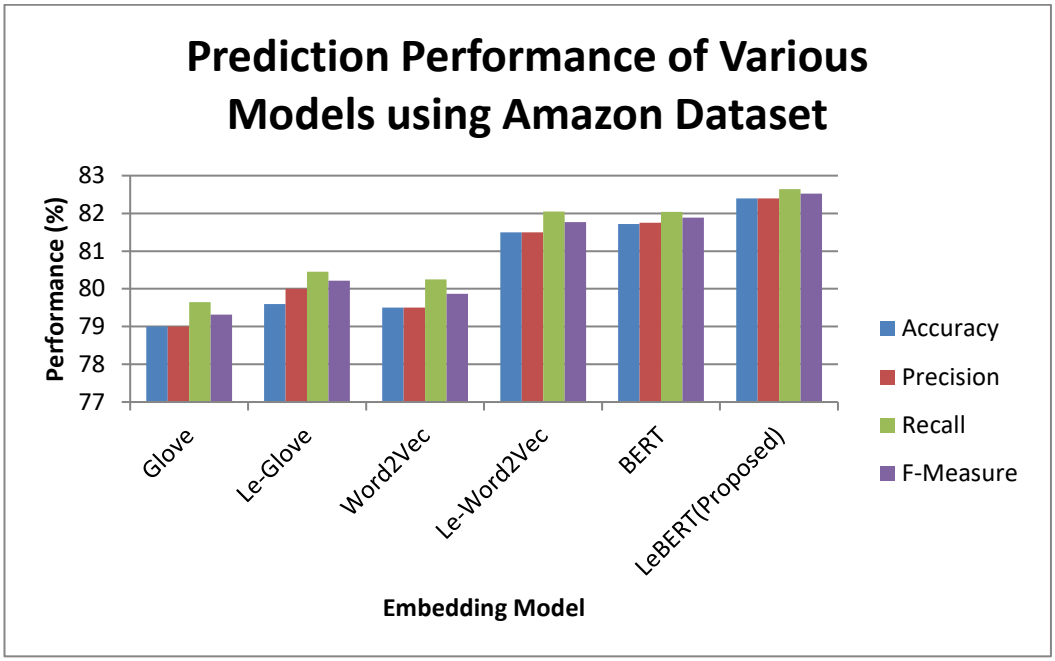


Figure 4.5: Validation Results of Embedding Models Using Amazon Dataset

As shown in Figure 4.5, the proposed LeBERT model had the highest Accuracy (82.4%), Precision (82.4%), recall (82.64%), and F-measure (82.52%) compared to all the other models. It is also worth noting that the use of lexicon in all the embedding algorithms improved the performance of the model. This is an indication that lexicon-selected sections of the input text had a positive impact on the performance of the sentiment analysis models on the Amazon dataset. Generally, it is observed that the use of a lexicon improved the performance of the sentiment analysis models across the three datasets used. This observation indicates the applicability of the proposed model across various datasets.

Accuracy is considered to be a good performance evaluation metric when the classes are balanced (Singh et al. 2022). Since, in our experiments, all three datasets exhibited balanced classes; we compared the accuracy of the proposed model with the various approaches for the three datasets. The results obtained are shown in Figure 4.6.

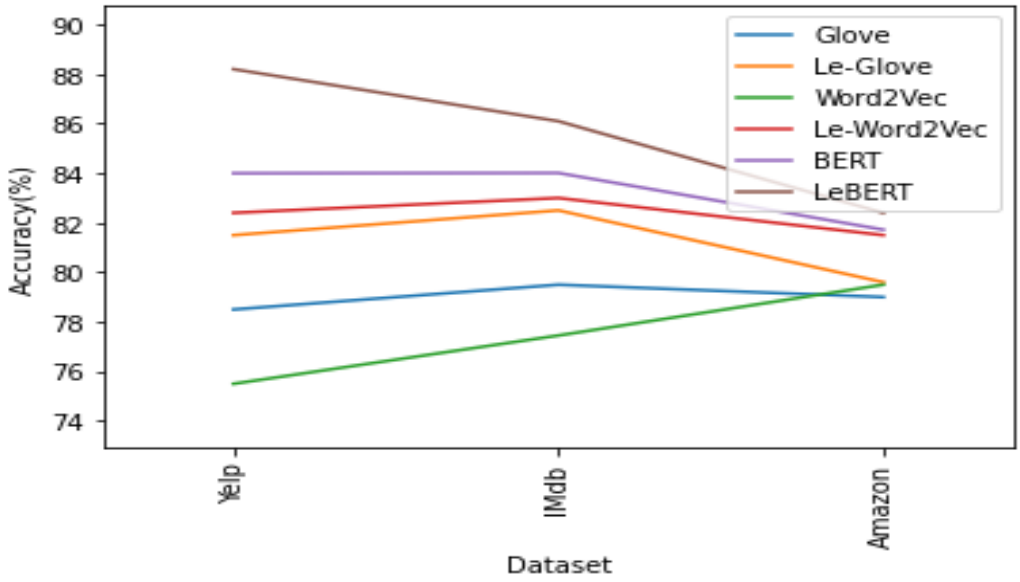


Figure 4.6: Comparison of Accuracy Using Various Embedding Models

From Figure 4.6, the proposed model (LeBERT) had the highest accuracy (89.01%) in all datasets, with relatively lower accuracy on Amazon's product reviews dataset. This could be attributed to the fact that the reviews referred to various products, and thus the sentiment terms varied from one product to another. Elhassan et al. (2023) in their study on Arabic Sentiment Analysis based on Word Embeddings and Deep Learning also found that word embedding features outperformed hand-crafted features. Their results are similar to the results obtained in this study since the text representations obtained using word embeddings yielded higher performance compared to NLP techniques.

4.3.4 Experiment Results 9: Proposed Sentiment Lexicon-Augmented Model with Transformer-Based Models

The experiment was carried out to validate the performance of the proposed Sentiment Lexicon-Augmented Model with transformer-based models in terms of accuracy, recall, precision, and F-measure on the three datasets. The performance of using the Lexicon-Augmented model was evaluated in BERT and XLNet models. In this experiment, tri-grams (N = 3) were used. The results obtained are presented in Table 4.12

Table 4.12: Performance of Proposed Sentiment Lexicon Enhanced Transformer-Based Models

Dataset	Model	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
Yelp	XLNet	83.85	83.85	84.15	83.99
	Le-XLNet	88.21	88.21	88.75	88.48
	BERT	84.00	84.00	84.20	84.10
	LeBERT	88.20	88.45	89.01	**88.73**
Imdb	XLNet	84.00	84.00	84.45	84.22
	Le-XLNet	86.70	86.70	87.20	**86.95**
	BERT	84.01	84.08	84.63	84.35
	LeBERT	86.10	86.71	87.00	86.85
Amazon	XLNet	82.15	82.15	82.25	82.20
	Le-XLNet	81.87	81.87	82.45	82.16
	BERT	81.72	81.75	82.04	81.89
	LeBERT	82.40	82.40	82.64	82.52

From Table 4.12, the LeBERT model had the highest F-measure (88.73%) compared to Le-XLNet (88.48%) in the Yelp Dataset. However, in the IMDB Dataset, Le-XLNet had the highest F-measure (86.95%) compared to LeBERT in the same dataset. Generally, the performance of the transformer-based models (XLNet and BERT) improved after augmentation with the lexicon. Existing studies show that XLNet outperforms better than BERT. From Table 4.12, the results obtained showed a slight variation which is attributed to the fact that XLNet performs better in text with longer sequences (Yang et al., 2019). In this experiment, social media data contains very short sentences which were shortened further by the introduction of the lexicon pointed segment.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1 Introduction

This chapter concludes the thesis with a discussion of the research findings and recommendations. The research objectives are revisited, mainly the improvement of social media sentiment analysis by proposing and evaluating a novel text representation model in the era of short-text data analytics. The approach offers a suitable solution to the high feature dimensionality of existing text representation techniques while improving the quality of social media sentiment analysis models. Later in the chapter, recommendations to advance this research work are presented.

5.2 Summary of Findings

The proliferation of online text data rapidly produced by users through social media applications like Twitter, Facebook, and LinkedIn has made social media sentiment analysis a hot research area. Although these data have been very useful in making decisions in organizations, governments, and other social formations, gaining intelligence from them requires accurate analysis methodologies applied in sentiment analysis, text mining, and information retrieval. This is due to the nature of the social media text data which is mainly unstructured or semi-structured and full of non-linguistic syntax. One of the main challenges to researchers has been the identification of features that represent the text and can be used as input into machine learning algorithms.

Most text representation techniques utilize a bag of words, N-grams, and frequency-based algorithms especially Term Frequency-Inverse document frequency (TF-IDF). Word embedding models are also being used for text representation in this era of deep learning. However, these approaches suffer shortcomings such as; they do not consider relationships between words, they ignore important words' characteristics and they focus on the entire document or sentence thus they suffer high feature dimensionality. Recently attention to feature extraction techniques that focus on a part or a section of

a document or text has increased. This has mainly been done on document-level sentiment analysis where the topic, introduction, or conclusion of the document is considered. In this thesis, a sentiment lexicon-based text representation model for both NLP-based techniques and word embeddings is proposed and investigated. In the model, a sentiment lexicon is used to identify a section of the input text that could be used to generate the resultant word vector representing the entire text. The resultant vector is rich in semantic and contextual information of words and thus utilized in sentiment analysis of the text via machine learning classifiers including deep learning neural networks.

Experiments carried out to evaluate the feasibility of the model were described and the results obtained were presented. Accuracy, Recall, precision, and F-measure were used as evaluation metrics. The results obtained attested to the viability of the approach compared to the baseline approaches chosen. The text representation model was tested with supervised machine learning classifiers; Naïve Bayes, K-Nearest Neighbor, Decision Tree, Support Vector Machines, and deep learning algorithm, the Convolutional Neural Network (CNN). The experiments that were done on publicly available datasets (Yelp Dataset, Imdb Dataset, and Amazon's Dataset) showed that the proposed model improved sentiment analysis significantly.

5.3 Review of the Research Objectives

There are four modules of a sentiment analysis model; text data input module, Text data preprocessing module, text representation module, and sentiment classification module. This study focused on the text representation module. Text representation algorithms have been used in document-level sentiment analysis but are sparsely used in short text classification tasks such as social media data sentiment analysis, hence the need to focus this study on social media.

The first objective of the study was to analyze existing text representation techniques for sentiment analysis to improve them for social media data sentiment analysis. These techniques were found to suffer high feature dimensionality and data sparseness since the features are extracted from the entire document or sentence. Word counts, TF-IDF, N-grams, and word embedding models have been used but there was a need to improve

them to solve the problem of high feature dimensionality and data sparseness as discussed. Word counts vectorizer performed better than TF-IDF models in the experiments however, due to the limited size of social media texts there was a need to improve them. The use of sentiment lexicon via N-grams to identify a section of the input text where opinion is concentrated was found to be a better approach in reducing the feature dimensionality of the resultant vector while enhancing its sentiment content.

For the second objective, a social media text representation model based on a sentiment lexicon-enhanced text representation model was developed. The text representation was achieved by using a sentiment lexicon to point to an N-gram containing a sentiment term from a subjective sentence. To improve the Natural Language Processing-based text representation techniques, the N-gram was expanded by involving the POS tags and the semantic orientations of the words (tokens) in the N-gram. The 3 N-gram was found to perform better than other values of N investigated in the experiments. The vector space was built using binary occurrence and TF-IDF algorithms and further reduced by the minimum redundancy maximum relevance feature selection algorithm. For the word embedding models, the lexicon selected N-grams were converted to a bag of words which became the input text to the embedding layer. Lexicon-enhanced BERT (LeBERT) embedding model outperformed the lexicon-enhanced Glove and Word2Vec models.

The sentiment lexicon-enhanced NLP-based text representation algorithm was implemented in various machine learning classifiers which included the SVMs, NB, K-NN, and Decision Trees. Based on the results, it can be concluded that SVMs were the most suitable classifier since they produced the best classification results when used with the algorithm. The lexicon-enhanced BERT embedding model (LeBERT) proved to be the best model for text representation using deep learning Convolutional Neural Networks.

The findings from this study demonstrate that the sentiment lexicon-based text representation model improved existing text representation techniques for social media sentiment analysis. This was validated using social media data obtained from Twitter.

The performance of the sentiment analysis model developed using the proposed sentiment lexicon-based text representation approach outperformed baseline text representation techniques.

5.4 Recommendations for Future Work

In this study, publicly available social media datasets were used which had reviews having one opinion and two classes of subjectivity that are negative and positive. Further, we would recommend the following. First, further studies are to be carried out to test the applicability of the proposed text representation model in reviews having more than one opinion which can also be classified into finer classes of extent of negativity or positivity. Second, the study has also shown the importance of feature selection and the use of deep learning algorithms in social media sentiment analysis. Further study can be carried out to determine the best combination of feature selection algorithms which was not covered in the study. Thirdly, studies on the application of the proposed text representation model in non-English social media data since this study focused on English reviews only. Fourthly, more experiments on text classification problems other than subjectivity such as misinformation to further investigate and enhance the utilization of the proposed model in the social media industry.

REFERENCES

Aisopos, F., Tzannetos, D., Violos, J., & & Varvarigou, T. (2016). Using n-grams graphs for sentiment analysis: an extended study on Twitter. *IEEE Second International Conference on Big Data Computing Services and Applications.* (pp. 44-51). IEEE.

Aisopos, F., Tzannetos, D., Violos, J., & Varvarigou, T. (2016). Using n-grams graphs for sentiment analysis: an extended study on Twitter. *IEEE Second International Conference on Big Data Computing Services and Applications.* IEEE.

Al-Azani, S., El-Sayed, M., & El-Alfy. (2017). Using Word Embedding and Ensemble Learning for Highly Imbalanced Data Sentiment Analysis in Short Arabic Text. *Procedia Computer Science, 109C* , 359-366.

Alnuaimi, N., Mohammed, M. M., Mohammed, S. A., & Zaki, N. (2019). Streaming Feature Selection Algorithms. *Applied Computing and Informatics*.

Ankit , N. S. (2018). An ensemble classification system for Twitter Sentiment Analysis. *Procedia Computer Science: Elsevier Journal Science Direct*, 937 - 947.

Appel, O., Chiclana, F., & Carter, J. (2015). Main Concepts, State of the Art, and Future Research Questions in Sentiment Analysis. *Acta Polytechnica Hungarica, 12*(3), 87-108.

Araque, O., Corcuera-Platas, I., Sánchez-Rada, J., & Iglesias. (2017). Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications*, 236–246.

Aytug, O., & Serdar, K. (2017). A feature Selection model based on genetic rank aggregation for text sentiment classification. *Journal of Information Science, 43*(1), 25-38.

Baharudin , B., & Khan, A. (2011). Sentiment Classification Using Sentence-Level Semantic Orientation of Opinion Terms from Blogs. *IEEE*

Bermingham, A., & Smeaton, A. F. (2010). Classifying Sentiment in Microblogs: Is brevity an advantage? In *19th ACM International Conference in Information and Knowledge Management*, (pp. 1833-1836).

Bhadane, C., Dalal, H., & Doshi, H. (2015). Sentiment Analysis: Measuring Opinions. *Procedia Computer Science, 45*, 808-814.

Bird, J. J., Ekart, A., & Faria, D. R. (2019). High-resolution Sentiment Analysis by Ensemble Classification. *Research Gate Publication*, 329874584.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics, 5*, 135-146.

Bouazizi , M., & Ohtsuki, T. (2017). A pattern-based approach for Multiclass Sentiment analysis in Twitter. *IEEE, 5*.

Bouras, C., & Tsogkas, V. (2016). Assisting cluster coherency via n-grams and clustering as a tool to deal with the new user problem. *International Journal of Machine Learning and Cybernetics, 7*, 171-184.

Brean, J. (2017). *Opinion Lexicon English.* Retrieved April 15th, 2017, from Github: https:/github.com/Jeffreybrean/Twitter-Sentiment Analysis Tutorial - 201107/data/opinion - lexicon-English

Brill, E. (1994). Some Advances in Transformation-Based Part of Speech Tagging. *AAAI*, 722-727.

Brindha, S., Sukumaran, S., & Prabha, K. (2016). A Survey on Classification Techniques for Text Mining. *3rd International Conference on Advanced Computing and Communication Systems, (ICACCS-2016) Jan 22-23.* Coimbatore, India.

Cambria, E., Schuller, B., Xia, Y., & Havasi, C. (2013). New Avenues in Opinion Mining and Sentiment Analysis. *IEEE Intelligent Systems, 28*(2), 15-21.

Catal, C., & Nangir, M. (2017). A Sentiment Classification Model Based on Multiple Classifiers. *Applied Soft Computing, 50*, 135-141.

Chandrasekaran, G., Nguyen, T., & Hemanth, J. (2021). Multimodal sentimental analysis for social media applications: A comprehensive review. *WIREs Data Min. knowl. Discov.* , e1415.

Cheng, Y., Yao, L., Xiang, G., Zhang, G., Tang, T., & Zhong, L. (2020). Text Sentiment Orientation Analysis Based on Multichannel CNN and Bidirectional GRU with Attention Mechanism. *IEEE Access, 8*, 134964 - 1344975.

Chug, A., Kohli, S., Gupta, S., Ahu, P., & Ahuja, R. (2019). The Impact of Features Extraction on the Sentiment Analysis. *Procedia Computer Science* , 341-348.

Cichosz, P. (2018). A Case Study in Text Mining of Discussion Forum Posts: Classification with Bag of Words and Global Vectors. *International Journal of Applied Mathematics and Computer Science, 28*(4), 787-801.

Curtis, M. J., Alexander, S. P., Cirino, G., George, C. H., Kendall, D. A., Insel, P. A., ... & Ahluwalia, A. (2022). Planning experiments: Updated guidance on experimental design and analysis and their reporting III. *British Journal of Pharmacology, 179*(15), 3907-3913.

D'Silva, J., & Sharma, U. (2022). Automatic text summarization of Konkani texts using pre-trained word embeddings and deep learning. *Int. J. Electr. Comput. Eng. (IJECE)*, 1990-2000.

Dahou, A., & Abd Elaziz Mohamed, J. Z. (2019). Arabic Sentiment Classification using Convolutional Neural Network and Differential Evolution Algorithm. *Computational Intelligence and Neuroscience*.

Dalal, M. K., & Zaveri, M. A. (2013). Semisupervised Learning - based Opinion Summarization and Classification for Online Reviews. *Applied Computer Intelligent and Software Computing*, 1-9.

Dave, K., Lawrence, S., & Pennock, D. M. (2003). Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. *Proceedings of the 12th International Conference on World Wide Web*, 519-528.

Deisy, W. (2010). A novel term weighting scheme MIDF for text categorization. *Journal of Engineering Science and Technology*, *5*(1), 94-107.

Devika, M. D., Sunitha, C., & Ganesh, A. (2016). Sentiment Analysis: A Comparative Study on Different Approaches. *Procedia Computer Science* , 44-49.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Droba, D. D. (1931). Methods used for measuring public opinion. *American Journal of Sociology*, 410-423.

Egejuru, N. C., Balogun, J. A., Mhambe, P. D., Asahiah, F. O., & Idowu, P. A. (2017). Model for prediction of Cataracts using Supervised Machine Learning Algorithms. *Computing, Information Systems, Development Informatics & Allied Research Journal, 8*(3), 47-62.

Elhassan, N., Varone, G., Ahmed, R., Gogate, M., Dashtipour, K., Almoamari, H., El-Affendi, M. A., et al. (2023). Arabic Sentiment Analysis Based on Word Embeddings and Deep Learning. *Computers, 12*(6), 126.

Esuli, A., & Sebastiani, F. (2006). *SentiWordNet: A High-Coverage Lexical for Opinion Mining.* Italy: Institute of Information Science and Technologies of the Italian National Council.

Feldman, R. (2013). Techniques and Applications for Sentiment Analysis. *Communications of the ACM, 56*(4), 82-89.

Gallego, S. R., Lastra, L., Martinez, R. D., Bolon - Canedo, V., Renitez, J., Herrar, F., et al. (2016). Fast-MRMR: Fast Minimum Redundancy Maximum Relevance Algorithm for High-Dimensinality Big Data. *International Journal of Intelligent Systems, 32*(2), 134 - 152.

Garg, S., & Subrahmanyam, V. (2022). Sentiment Analysis: Choosing the Right Word Embedding for Deep Learning Model. In M. P. Bianchini, *Advanced Computing and Intelligent Technologies: Lecture Notes in Networks and Systems* (p. 218). Singapore: Springer.

Ghosh, M., & Sanyal, G. (2018). An Ensemble Approach to Stabilize the features for multi-domain Sentiment Analysis using Supervised Machine Learning. *Journal of Big Data*, 5-44.

Giatsoglou, M., Vozalis, M., Diamantaras, K., Vakali, A., Sarigiannidis, G., & Chatzisavvas, K. (2017). Sentiment analysis leveraging emotions and word embeddings. *Expert Systems With Applications*, 214-224.

Github, P. (2017). *jeffreybrean/twitter-sentiment analysis tutorial-201107/data/opinion-lexicon-English.* Retrieved from Github: Https://github.com

Gosh, S., Roy, S., & Bandyopadhyay, S. K. (2012). A tutorial review on Text Mining Algorithms. *International Journal of Advanced Research in Computer and Communication Engineering, 1*(4), 223-232.

Guo, S., & Yao, N. (2020). Generating word and document matrix representations for document classification. *Neural Computing and Applications, 32*, 10087-10108.

Gu. T, X. &. (2020). Sentiment Analysis via Deep Multichannel Neural Networks With Variational Information Bottleneck. *IEEE Access, 8*, 121014-121021.

Haddi, E., Liu, X., & Shi, Y. (2013). The Role of Text Preprocessing in Sentiment Analysis. *Procedia Computer Science, 17*, 26-32.

Han, Y., & Kwangmi, K. K. (2017). Sentiment Analysis on Social Media Using Morphological Sentence Pattern Model. *IEEE SERA 2017.* London UK.

Hassan, S. U., Ahamed, J., & Ahmad, K. (2022). Analytics of machine learning-based algorithms for text classification. *Sustainable Operations and Computers,* 238-248.

Hatzivassiloglou, V., & W., W. J. (2000). Effects of Adjective Orientation and Granuality on Sentence Subjectivity. *ACL*, 174-181.

Hearst, M. A. (1992). Direction - Based Text Interpretation as an Information Access Refinement. Text-Based *Intelligent Systems,* 1-13.

Houda, O., Omar, N., & Phillippe, B. (2014). Minimum Redundancy and maximum relevance for single and multi-document Arabic text Summarization. *Journal of King Saudi University - Computer and Information Sciences, 26,* 450 - 461.

Hu, Y., Ding, J., Dou, Z., & Chang, H. (2022). Short-Text Classification Detector: A Bert-Based Mental Approach. *Comput. Intell. Neurosci.*

Emotion. (2017, January). *Statistical tools.* Retrieved from Https://Imotions.com/ blog/statistical- tools

Ramos, J. (2003, December). Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning* (Vol. 242, No. 1, pp. 29-48).

Jain, P., Quamer, W., Saravanan, V., & Pamula, R. (2022). Employing BERT-DCNN with a semantic knowledge base for social media sentiment analysis. *J. Ambient. Intell. Humans. Comput.* , 1-13.

Jain, T., Verma, V. K., Sharma, A. K., Saini, B., Purohit, N., Mahdin, H., & Arshad, M. S. (2023). Sentiment Analysis on COVID-19 Vaccine Tweets Using

Machine Learning and Deep Learning Algorithms. *International Journal of Advanced Computer Science and Applications, 14*(5).

Ji, F., Liu, Z., Qiu, X., & Huang, X. (2012). Part of Speech Tagging for Microblog via 2D Sequence Labelling. *Journal of Computational Information Systems, 8*(3), 1149-1156.

Jihson, Z., Meng, Z., & Mao, L. (2014). Big Data Visualization: Parallel Coordinates using Density Approach. *2nd International Conference on Systems and Informatics,(ICSAI).* IEEE.

Kaushik, A., Kaushik, A., & Naithani, S. (2015). A study of Sentiment Analysis: Methods and tools. *International Journal of Science and Research(IJSR), 4*(12).

Kenton, J., & Toutanova, L. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *naacL-HLT*, (pp. 4171-4186). Minneapolis, Minnesota.

Kharde, V., & Sonawane, S. S. (2011). Sentiment Analysis of Twitter Data: A Survey of Techniques. *International Journal of Computer Applications, 139*(11).

Khedkara, S., & Shinde, S. (2020). Deep Learning and ensemble approach for praise and complaint Classification. *Procedia Computer Science* , 449- 458.

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1746-1751). Doha, Qatar: Association for Computational Linguistics.

Koto, F., & Adriani, M. (2015). The use of POS sequence for analyzing sentence patterns in Twitter Sentiment analysis. *29th International Conference on Advanced Information Networking and Applications Workshops* (pp. 547-551). IEEE Computer Society.

Kotsiantis, I. D., Zaharakis, P. E., & Pintela, S. (2006). Machine Learning: A Review of Classification and Combining Techniques. *Artificial Intelligence Review, 26*(3), 159-190.

Kotzias, D., Denil, M., de Freitas, N., & Smyth, P. (2015). From Group to Individual Labels Using Deep Features. *In Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* New York: Association for Computing Machinery.

Kukkar, A., Mohana, R., Sharma, A., Nayyar, A., & Shah, M. A. (2023). Improving Sentiment Analysis in Social Media by Handling Lengthened Words. *IEEE Access*, 9775-9788.

Kumar, A., & Sebastian, T. M. (2012). Sentiment Analysis: A Perspective on Its Past, Present and Future. *International Journal of Intelligent Systems and Applications, 4*(10), 1-14.

Li, G., & Lin, Z. &. (2020). A Discriminative Approach to Sentiment Classification. *Neural Process Letters, 51*, 749 - 758.

Li, G., Lin, Z., & Wang, H. (2020). A Discriminative Approach to Sentiment Classification. *Neural Processing Letters, 51*, 749-758.

Liu, B. (2010). Sentiment Analysis: Multifaced problem. *IEEE Intelligent Systems, 25*(3), 76-80.

Liu, B., & Tsou, K. (2010). Combining a Large Sentiment Lexicon and Machine Learning for Subjectivity Classification. *In Ninth IEEE International Conference on Machine Learning and Cybernetics*, (pp. 3311-3316).

Liu, H., Chen, X., & Liu, X. (2022). A study of the application of weight distributing method combining sentiment dictionary and TF-IDF for text sentiment analysis. *IEEE Access, 10*, 32280-32289.

Liu, J., Zheng, S., Xu, G., & Lin, M. (2021). Cross-domain sentiment-aware word embeddings for review sentiment analysis. *Int. J. Mach. Learn. Cybern.* , 343–354.

Liu, Y., Liu, B., Shan, L., & Wang, X. (2018). Modeling context with neural networks for recommending idioms in essay writing. *Neurocomputing* , 2287-2293.

Loshchilov, l., & Hutter, F. (2016). CMA-ES for Hyperparameter Optimization of Deep Neural Networks. *arXiV.org/abs/1604.07269X1* .

M., H., & B., L. (2014). Mining and Summarizing Customer Reviews. *10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, (pp. 168-177).

Makrehchi, M., & Kamkarhaghighi, M. (2017). Content Tree Word Embedding for document representation. *Expert Systems with Applications, 90*, 241-249.

Mäntylä, V. M., Graziotin, D., & Kuutila, M. (2018). The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. *Computer Science Review*, 16-32.

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment Similarity algorithms and applications: A survey. *Ain Shams Engineering Journal, 5*(4), 1093-1113.

Miao, J., & Niu, L. (2016). A Survey on Feature Selection. *Procedia Computer Science* , 919 - 926.

Mikolov, T. S. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, *26*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *In Proceedings of 26th International Conference on Neural Information Processing Systems* (pp. 3111-3119). Lake Tahoe: CA.

Miller, G. A., Beckwith , R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to WordNet: An On-Line Lexical Database. *International Journal of Lexicography, 3*(4), 235-244.

Milos, R., Mohamed, G., Nenad, F., & Zoran, O. (2017). Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC Bioinformatics*.

Mingyong, L., & Yang, J. (2012). An improvement of TF-IDF weighting in text categorization. *International Conference on Computer Technology and Science (ICCTS).* Singapore: IACSIT.

Mozetic, I., Grcar, M., & Smailovic, J. (2016). Multilingual Twitter Sentiment Analysis: The Role of Human Annotators. *PLoS One, 11*(5) .

Musleh, D. A., Alkhwaja, I., Alkhwaja, A., Alghamdi, M., Abahussain, H., Alfawaz, F., Min-Allah, N. (2023). Arabic Sentiment Analysis of YouTube Comments: NLP-Based Machine Learning Approaches for Content Evaluation. *Big Data and Cognitive Computing, 7*(3), 127. http://dx.doi.org/10.3390/bdcc7030127

Mutinda, J., Mwangi, W., & Okeyo, G. (2021). Lexicon-pointed hybrid N-gram Features Extraction Model (LeNFEM) for sentence-level sentiment analysis. *Engineering Reports*, e12374.

Nagamanjula, R., & Pethalakshmi, A. (2020). A novel framework based on bi-objective optimization and LAN2FIS for Twitter sentiment analysis. *Social Network Analysis and Mining, 34*.

Noaman H. M., S. S. (2018). Enhancing Recurrent Neural Network-based Language Models by Word Tokenization. *Human Centric Computing Information Science, 8*(12), 8-12.

P., B., E., G., A., J., & T., M. (2017). Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguist.* , 135–146.

Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval, 2*(12), 1-135.

Parwez M. A, A. M. (2019). Multi-Label Classification of Microblogging Texts Using Convolution Neural Network. *IEEE Access, 7*, 68678 - 68691.

Passalis, N., & Tefas, A. (2018). Learning Bag - of - Embedded- Words Representation for Textual Information Retrieval. *Pattern Recognition, 81*, 254 - 267.

Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. *16. Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empir.* Doha, Qatar: Association for Computational Linguistics.

Prottasha, N., Sami, A., Kowsher, Murad, S., Bairagi, A., Masud, M., et al. (2022). Transfer Learning for Sentiment Analysis Using BERT-Based Supervised Fine-Tuning. *Sensors*, 4157.

Qader, W. A., Ameen, M. M., & Ahmed, B. I. (2019, June). An overview of bag of words; importance, implementation, applications, and challenges. In 2019 *International Engineering Conference (IEC)* (pp. 200-204). IEEE.

Rezaeinia, S., Rahmani, R., Ghodsi, A., & Veisi, H. (2019). Sentiment analysis based on improved pre-trained word embeddings. *Expert Systems with Applications*, 139-147.

Robertson, S. (2004). Understanding inverse document frequency: on Theoretical Arguments for IDF. *Journal of Document, 60*, 503-520.

Rudkowsky , E., Haselmayer, M., Wastian, M., Jenny, M., Stefan, E., & Sedlmair, M. (2018). More than Bags of Words: Sentiment Analysis with Word Embeddings. *Journal of Communication Methods and Measures*, 140-157.

Sakor, A., Mulang, I. O. ', Kuldeep Singh, Saeedeh Shekarpour, Maria Esther Vidal, Jens Lehmann, and Sören Auer. (2019). Old is gold: Linguistic driven approach

for entity and relation linking of short text. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 2336-2346).

Salvetti, F., Lewis, S., & Reichenbach, C. (2014). Impact of Lexical filtering on overall Opinion Polarity Identification. *In Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications.* Stanford, US.

Samira, P., Saad, S., Yilin, Y., Haiman, T., Yudong, T., Maria, P., et al. (2018). A survey on Deep Learning: Algorithms, Techniques, and Applications. *ACM Computing Surveys*,, *51*(5), DOI:https://doi.org/10.1145/3234150.

Santur, Y. (2019, September). Sentiment analysis based on gated recurrent unit. *In 2019 International Artificial Intelligence and Data Processing Symposium* (IDAP) (pp. 1-5). IEEE.

Saraswathi, K., & Tamilarasi, A. (2014). Investigation of Support Vector Machine Classifier for Opinion Mining. *Journal of Theoretical and Applied Information Technology.*, *59*(2), 291-296.

Sebastian, F. (2002). Machine Learning in automated text categorization. *ACM Computing Surveys 34*(1), 1-47.

Shrestha, A., & Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE Access, 7,* 53040-53065.

Simha, A., & Iyengar. (2007). Medical Datamining with a New Algorithm for Feature Selection and Naive Bayesian Classifier in ICIT. *IEEE Computer Society*, 44-49.

Singh, K.N., Devi, S.D., Devi, H.M., & Mahanta, A.K. (2022). A novel approach for dimension reduction using word embedding: An enhanced text classification approach. *Int. J. Inf. Manag. Data Insights, 2,* 100061.

Srivastava, B. B. (2018). Sentiment Classification of Online Consumer Reviews Using Word Vector Representations. *Procedia Computer Science 132* , 1147 - 1153.

Suanmali, L., Salim, N., & Binwahlan, M. S. (2009). Fuzzy Logic-Based Method for Improving Text Summarization. *International Journal of Computer Science and Information Security, 2*(1).

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics 37*(2), 267-307.

Talaat, A. S. (2023). Sentiment analysis classification system using hybrid BERT models. *Journal of Big Data, 10*(1), 1-18.

Tian, X., & Yanmei, C. (2011). An Improvement of TF-IDF: Term Based Term Weighting Algorithm. *Journal of Software, 6*(3).

Turney, P. D. (2002). Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification Reviews. *40th Annual Meeting of the Association for Computational Linguistics*, (pp. 417-424).

Velikovich, L., Blair-Goldensohn, S., Hannan, K., & McDonald, R. (2010). The Viability of Web-derived Polarity Lexicons. *Tech. Rep.*

Wang, G., Sun, J., Ma, J., Xu, K., & Gu, J. (2014). Sentiment Classification: The Contribution of Ensemble Learning. *Decision Support Systems, 57*, 77-93.

Wang, Y., Huang, G., Li, J., Li, H., Zhou, Y., & Jiang, H. (2021). Refined global word embeddings based on sentiment concept for sentiment analysis. *IEEE Access, 9*, 37075-37085.

Wang, Y., Huang, M., Zhu, X., Zhao, L. (2016). Attention-based LSTM for Aspect-level Sentiment Classification. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (pp. 606-615). Texas: Austin.

Wen, Y., Liang, Y., & Zhu, X. (2023). Sentiment analysis of hotel online reviews using the BERT model and ERNIE model—Data from China. *PLoS ONE, 18*(3), https://doi.org/10.1371/journal.pone.0275382.

Wiebe, J. M., Bruce, R. F., & O'Hara, T. P. (1999). Development and use of a gold-standard data set for subjectivity classifications. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, 246-253.

Wiebe, J. (1994). Tracking Point of View in Narrative. *Computational Linguistics. 35*(3), 399-433.

Wilson, T., Wiebe, J., & Hoffmann, P. (2009). Recognizing Contextual Polarity: An Exploration of Features for Phrase-Level Sentiment Analysis. *Computational Linguistics 35*(3), 399-433.

Yang, H. (2022). Network Public Opinion Risk Prediction and Judgment Based on Deep Learning: A Model of Text Sentiment Analysis. *Comput. Intell. Neurosci.*

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pre-training for language understanding. *Advances in neural information processing systems*, *32*.

Young, R. S., Rose, D. C., Karnowski, T. P., & S. H. Lim, R. M. (2015). Optimizing Deep Learning Hyper Parameters Through an Evolutionary Algorithm. *Workshop on Machine Learning in High-Performance Computing Environments*, (p. 4). New York, NY, USA.

Zaidi, S. F. A., Awan, F. M., Lee, M., Woo, H., & Lee, C. G. (2020). Applying convolutional neural networks with different word representation techniques to recommend bug fixers. *IEEE Access, 8*, 213729-213747.

Zhenyu, L., Haiwei, H., Chaohong, L., & Shengfei, L. (2020). Multichannel CNN with Attention for Text Classification. *arXiv:2006.16174v1* .

Zhou, H., Wang, X., & Zhang, Y. (2020). Feature Selection based on weighted conditional mutual information. *Applied computing and informatics*.

APPENDICES

Appendix I: Author's Publications during PhD Study

Mutinda, J., Mwangi, W., & Okeyo, G. (2021). Lexicon-pointed hybrid N-gram Features Extraction Model (LeNFEM) for sentence-level sentiment analysis. *Engineering Reports*, *3*(8), e12374.

Mutinda, J., Mwangi, W., & Okeyo, G. (2023). Sentiment analysis of text reviews using lexicon-enhanced Bert embedding (LeBERT) model with convolutional neural network. *Applied Sciences*, *13*(3), 1445.