SENTIMENT ANALYSIS OF TOXIC COMMENTS ON SOCIAL MEDIA USING DEEP LEARNING

By

Huba Noor

NATIONAL UNIVERSITY OF MODERN LANGUAGES

ISLAMABAD

May 2024

Sentiment Analysis of Toxic Comments on Social Media Using Deep Learning

By

Huba Noor

BSSE, IIUI, Islamabad, 2019

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

In Software Engineering

To

FACULTY OF ENGINEERING & COMPUTING

NATIONAL UNIVERSITY OF MODERN LANGUAGES ISLAMABAD

THESIS AND DEFENSE APPROVAL FORM

The undersigned certify that they have read the following thesis, examined the defense, are satisfied with overall exam performance, and recommend the thesis to the Faculty of Engineering and Computer Sciences for acceptance.

Thesis Title: Sentiment Analysis of Toxic Comments on Social Media Using Deep Learning.

Submitted by: Huba Noor **Registration #:**39/MS/SEF20

Master of Science in Software Engineering

Software Engineering

Name of Discipline

Dr. Muzafar Khan

Name of Research Supervisor

Signature of Research Supervisor

Dr. Raheel Zafar

Name of Research Co-Supervisor

Signature of Research Co-Supervisor

Dr. Sumaira Nazir

Name of Head of Department SE

Signature of Head of Department SE

Dr. Noman Malik

Name Dean (FEC)

Signature of Dean (FEC)

May 2024

AUTHOR'S DECLARATION

I Huba Noor

Daughter of Ezaz Muhammad Khan

Registration # 39/MS/SE/F20

Discipline Software Engineering

Candidate of **Master of Science in Software Engineering (MSSE)** at the National University of Modern Languages do hereby declare that the thesis **Sentiment Analysis of Toxic Comments on Social Media Using Deep Learning** submitted by me in partial fulfillment of MSSE degree, is my original work, and has not been submitted or published earlier. I also solemnly declare that it shall not, in the future, be submitted by me for obtaining any other degree from this or any other university or institution. I also understand that if evidence of plagiarism is found in my thesis/dissertation at any stage, even after the award of a degree, the work may be canceled, and the degree revoked.

_____

Signature of Candidate

Huba Noor

Name of Candidate

May 2024

Date

ABSTRACT

SENTIMENT ANALYSIS OF TOXIC COMMENTS ON SOCIAL MEDIA USING DEEP LEARNING

In the rapidly evolving field of natural language processing, accurately predicting sentiments in text remains a critical challenge. This thesis addresses the problem by developing a novel multi-head model combining transformer-based architectures, DistilBERT and RoBERTa, with Bi-LSTM layers. Leveraging their complementary strengths, the model captures both global context and sequential dependencies in textual data. The research methodology involves extensive data preprocessing, model training, and evaluation using accuracy and F1-scores. Results demonstrate that the multi-head model outperforms traditional approaches, achieving a notable accuracy of 90.02%. This advancement offers significant benefits, including improved sentiment-driven decision-making and valuable insights across various industries, such as social media monitoring, customer feedback analysis, and market research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENT

First and foremost, I would like to express my heartfelt gratitude and deep appreciation to Almighty Allah, whose blessings made this study possible and successful. Without divine support, this achievement would not have been possible.

I am immensely thankful to my mother Zahida Ezaz my father Ezaz Muhammad Khan and my Husband S. Huzaifa Hussain and all the individuals and sources whose unwavering support and encouragement played a pivotal role in the completion of this study. Their honest espousal has been invaluable, and I am sincerely grateful for their contributions. I owe a special debt of gratitude to my research supervisor Dr. Muzafar Khan and Co supervisor Dr. Raheel Zafar, whose dedication and guidance were instrumental in shaping my research journey. Their commitment and relentless efforts left no stone unturned, and I am truly grateful for their mentorship.

To every person who has contributed to my success in ways both seen and unseen, I extend my heartfelt thanks. Your support has been an indispensable part of this endeavor, and I am deeply appreciative for everything you have done.

DEDICATION

I wholeheartedly dedicate this thesis work to the pillars of my life: my beloved parents, my husband, the remarkable teachers, and my cherished friends, all of whom have been unwavering in their support throughout my entire educational journey. Their boundless love, encouragement, and belief in my potential have been the driving force behind my pursuit of knowledge and academic excellence.

To my parents, whose unwavering love and sacrifices have been a constant source of strength, I owe my deepest gratitude. They have stood by me during the challenging and triumphant moments, providing unwavering support and guiding me with their wisdom.

I am equally indebted to my extraordinary teachers, who have not only imparted knowledge but also instilled in me the virtues of discipline, perseverance, and integrity. Their passions for teaching and dedication to their students have inspired me to strive for greatness in every endeavor I undertake.

CHAPTER 1

INTRODUCTION

The Chapter provides an in-depth exploration of sentiment analysis within the realm of social media, focusing particularly on the detection and management of toxic comments. It outlines the significance of sentiment analysis in understanding public opinion and decision-making processes, while also discussing the challenges posed by nuanced language and evolving user behaviors. Through research questions and objectives, it aims to enhance the accuracy of sentiment analysis through the integration of advanced deep learning techniques. The overview highlights the importance of continuous research to address the complexities of online discourse and foster healthier online communities.

1.1 Context

Social media platforms have integrated themselves into our daily routines, providing individuals with a space to express their thoughts, viewpoints, and emotions. These platforms encourage communication and connection, but they also face challenges in managing detrimental and harmful content. Struggle to control harmful and toxic information. Online communities may become hostile and toxic when users use insulting, abusive, or harassing language in their comments. Sentiment analysis, which falls under the umbrella of Natural Language Processing (NLP), presents a valuable means to tackle the problem of toxic comments on social media. The primary goal of sentiment analysis is to identify and classify the emotional context or sentiment conveyed within a written text. In the context of toxic comments, sentiment analysis can help identify whether a comment carries negative sentiment or not.

The sharing of data is influenced by social media. People share their thoughts on many issues via internet to enlighten other users, attract attention, distribute knowledge, or usually just to express their thoughts, whether they are good or bad. On daily basis a huge volume of text data is disseminated on the internet. Because of this, it is quite helpful to estimate user interest in specific topics by looking at user opinions. Online websites are common forms of communication. In the midst of all this, sifting through vast amounts of text data to discern valuable information becomes a challenging endeavor. As helpful as it is, it is also essentially impossible to filter through numbers of comments manually and systematically in internet to find perspectives. Data mining and machine learning can be used in this situation. "Toxic" refers to something that is harmful, poisonous, or detrimental, often in a figurative sense. In the context of social media or online communication, "toxic comments" typically refer to messages or interactions that are aggressive, offensive, harassing, or otherwise harmful to individuals or communities. These comments can include hate speech, personal attacks, threats, bullying, or other forms of harmful behavior [1].

Individuals communicate their viewpoints and ideas on social media. The most well-known social networking sites are Facebook, Twitter, and YouTube, where users express their opinions through comments and likes [1]. At this time, the internet has become a major part of our life. Most people share their thoughts on many topics through social networking or blogging websites. Additionally, they use these websites to research other people's perspectives. As a result, sentiment analysis from data mining has grown to be a significant area of study [2]. Opinion mining, also known as Sentiment Analysis, is the process of analyzing words and sentences to discern the public's perspective [3]. Sentiment analysis, often known as opinion mining, is a branch of natural language processing (NLP) that focuses on to understanding and extracting sentiments and views from textual material. It entails evaluating the subjective content within a text to determine if the opinion expressed is a positive or negative one.

With the explosion of digital content and the widespread use of social media, sentiment analysis has gained significant importance. It provides valuable insights into public opinion, customer feedback, and brand perception. By automatically analyzing large volumes of text data, sentiment analysis enables organizations to understand the sentiment of their customers, track public sentiment towards their products or services, and make data-driven decisions. Sentiment analysis is being used to examine comments posted on different social media platforms. A wide range of applications have made use of sentiment analysis, such as the examination of product reviews and movie trailers. Sentiment analysis methods from deep learning and machine learning are used to create public reviews [4].

In this study, we aim to enhance the accuracy and effectiveness of sentiment analysis through the integration of advanced deep learning techniques. Specifically, we propose to leverage Bidirectional Encoder Representations from Transformers (BERT) in conjunction with Bidirectional Long Short-Term Memory (BiLSTM) layers. Our goal is to capitalize on BERT's ability to capture intricate linguistic patterns and semantic representations, while simultaneously leveraging BiLSTM layers to capture sequential dependencies within text data. By combining these methodologies, we aim to develop a more robust sentiment analysis model capable of accurately discerning sentiments expressed in diverse textual contexts.

Traditional approaches to sentiment analysis often rely on handcrafted features and shallow machine learning models such as logistic regression or support vector machines. While these methods can achieve decent performance, they may struggle to capture the intricate nuances of human language and context-dependent sentiments. Recent advancements in deep learning, particularly transformer-based models like BERT (Bidirectional Encoder Representations from Transformers), have revolutionized the field of NLP. These models leverage large-scale pre-training on vast text corpora to learn rich, contextualized representations of language. However,

while BERT excels at capturing global context and semantics, it may struggle with capturing sequential dependencies within text data.

1.2 Level of Sentiment Analysis

The Sentiment analysis is distinguished into three sub-levels Sentence level, Document level and Aspect level [5]. Shows in fig.1. Sentence-level sentiment analysis may not be well-suited for complex sentences, as its primary purpose is to identify opinions within individual sentences and categorize them as either positive or negative. Sentiment analysis is document level is simplest way to study public sentiment. It examines entire textual content and then classified with one tag either negative or positive. The last type is Sentiment analysis is Aspect level can accomplish simple and short sentences but weak in long and complex sentences. In contrast to the other two levels, Aspect does not take language structure, such as articles, sentences, and clauses into consideration. By separating sentiments, it is possible to identify a sentence [6].

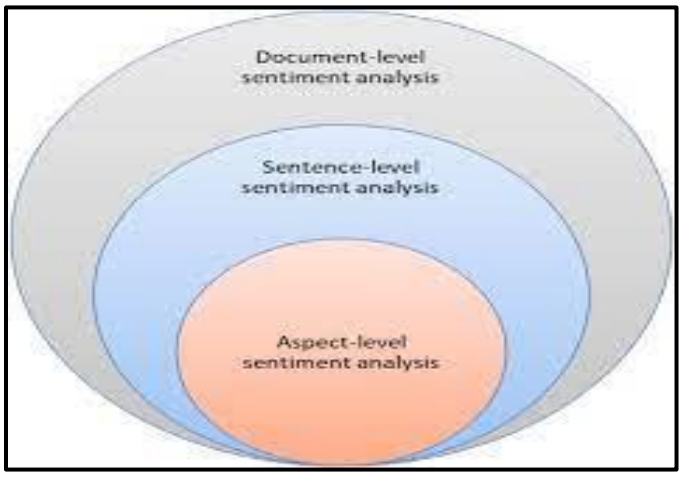

Figure 1.1: Level Of Sentiment Analysis

When a decision must be made, the views of others can be important. Considering those choices involves significant resources, people consider their companions' previous encounters. Social media today offers new means for easily sharing ideas with those connected to the Internet. Polarity detection (positive or negative) is the main focus of sentiment analysis. Twitter is a micro blogging website with several brief applications for social media marketing. Political parties, for example, could be interested to see whether people concur with their educational philosophy. The demand to acquire opinions from social networking sites and ascertain what people like or detest has been the most important point of view in the current circumstances. The focus of the study is primarily on document-level sentiment analysis. This involves analyzing entire texts or documents to classify them with a single sentiment label, such as positive or negative.

1.3 Impact of Sentiment Analysis

Making decisions affects many aspects of our existence. Using the opinions gained from the reviews in determining a number of decisions, including "which books to buy," "which hotel to stay at," "which movie to watch," etc. Every business seeks to satisfy client needs in today's competitive marketplace by developing fresh, cutting-edge items. To gather feedback from customers and implement the necessary product enhancements, a business needs today focus on individual reputation as a key factor. Using sentiment analysis, you may foresee developments in the markets through tracking public opinion. It is also helpful in elections as candidates want to know what public opinion expectations from the outcome [7]. Tracking the context of tweets associated with particular goods, services, or market trends helps with market research. This useful data help to utilized to spot new trends, comprehend customer preferences, and create marketing plans. Tweets on political occasions, campaigns, or policy decisions to provide or analyzed to learn important information about the public's emotions. This data may be used by political parties and candidate’s judge voter moods, improve their messaging, and adjust their strategy as needed [2].

1.4 Application of Sentiment Analysis

Online applications of sentiment analysis systems encompass online marketing, assessing the perspectives of bloggers, and analyzing people's attitudes toward products, among others [8]. Following applications are explained below.

1. Most people provide reviews of products on the retailer's website. Therefore, it is effective to create a system that suggests higher-quality and better-rated items to new website visitors in accordance with the sentiment analysis of user comments collected on shop websites.

2. A significant portion of many websites are allotted to internet advertising. Analyzing the content of advertising websites allows an internet marketer to select a website with great content for promoting their items.

3. Through data analysis, business intelligence aims to boost an organization's profitability and improve customer relations. A business intelligence specialist may offer a better technique to record positive remarks and, as a consequence, better benefit from product sales by assessing the feelings of user comments. Text comments recorded on a website are a set of unstructured text data [9].

1.5 Classification Issues in Sentiment Analysis

The study of sentiment first appeared to be just a classification issue, but as digging deeper, it appears to be more difficult. Language-based obstacles (Linguistic) and accuracy-based challenges are two different categories of difficulties.

1.5.1 Language-Based obstacles

These difficulties are primarily related to the English language. Texts can contain sarcastic comments or unspoken emotions, which pose a challenge in their detection and may result in inaccurate conclusions [10]. For instance, a statement like "This phone boasts an impressive battery life of 38 hours" is not sarcastic, whereas another statement like "This phone has an outstanding battery life of 2 hours" is clearly a sarcastic remark. The polarity of the sentence changes depending on the circumstances. "Long" can, for instance, indicate a variety of meanings depending on the context. For instance, "This phone has a long battery life," conveys a positive opinion on the battery life, but it can also convey a negative opinion in another context, such as "The car takes too long to start." [11]. Over time, people's opinions on an issue may evolve. Therefore, the timing of the analysis should also be considered [12]. Users don't always utilize proper language and spelling due to impatience, the informal nature of the internet, and length limitations. This results in mistakes in spelling, abbreviations, strong uppercase letters, category extension, and the usage of jargon [10]. When attempting to use natural language processing systems, tokens like URLs and emoticons might get complex [11]. In the same post, some individuals express their opinions in many languages. This makes the mining procedure more difficult [12]. The lack of clarity in word meaning is the largest difficulty in sentiment analysis. A term might have a positive meaning in one situation and a negative meaning in another. Use the term "short" as an example. Customer feedback that the PC started in a short time would be

positive. A customer's complaint about the computer's limited battery life would be negative. It is not possible to use a system that has been trained to gather feedback on specific sorts of items for other types of products [13]. Words like "OMG," "TC," or "LOL" may be used by people to indicate their response. To train our network to accurately recognize the sentiment, extra work is needed to identify these terms. Irony statements, like sarcasm, lead to an inaccurate understanding of the data and provide issues for natural language processing. With the widespread usage of graphical emoticons (also known as emojis), it can be challenging to understand how people are expressing themselves [7].

1.5.2 Accuracy-based Challenges

This includes challenges that have an impact on the classification models' accuracy. These are classified into two categories: theoretical problems and technical problems [14]. This is a theoretical problem that calls for a massive lexical dataset in order to obtain highly accurate findings. With the daily introduction of new word, managing vocabulary has grown more challenging. This is a technical problem since, depending on the context; words can be both positive and negative at the same time [7]. This raises a theoretical problem. While most individuals have the liberty to openly and anonymously express themselves on the internet, there are cases where certain individuals or groups exploit this freedom by posting spam reviews with the intention of either endorsing or disparaging other products [12].

1.6 Challenges in Sentiment Analysis

The main purpose of sentiment analysis is to identify words that indicate emotion from unstructured text inputs and categorize them as positive or negative. First, must use hypothesis testing techniques to extract the sentiment topics from the input data [10] . The independent words are to be removed. Phases and Adverb can present the sentiment of the text. However, determining

the polarity words is still a challenging procedure because people don't always express their opinions in the same way and the same opinion word may be viewed as positive in one context and negative in another [15]. People occasionally use emoticons (emojis) to express how they feel about a product. Now understand the significance of sentiment analysis. Identifying and extracting sentiment from data is the basic concept of sentiment analysis.

The terms "subjective" and "objective" can be used similarly in two separate contexts. Movies are quite noisy, and noisy videos are less accurate. The words "noisy" are used to denote both subjective and objective terms in the sentences above. Numerous sentences strongly convey intense emotions, yet pinpointing the origin of these feelings can be challenging. As a result, establishing connections to specific keywords or phrases becomes problematic. For instance, “When I read the book ‘I too had a love story’, I really want my to-be husband like him”. Here, the review's content should be relating to the story's hero. Certain words within the text capture the general feeling of the document. Good performances, an intriguing story, and stronger editing. But the movie is poor. Despite the initial positive statements, it's the last sentence that shifts the overall review into a negative tone. Depending on the domain, the same sentiment term may be used in a positive or negative context. The movie's climax is unexpected. "The car is moving at an unpredictable speed." In the first evaluation, unpredictability is seen positively but negatively in the second.

Sarcastic sentences use positive opinion terms to convey negative opinions. It expresses the opinion on things indirectly. Nice cologne. It must be used to marinate. Despite being a pleasant statement, it unintentionally criticizes perfume in negative way. In normal text categorization, the sentence structure can sometimes highly indicate the view about the product. "The Android phone has a much longer battery life than the IPhone phone." This statement provides a positive review of Android and a negative review for IPhone. Everyone has a unique viewpoint on the product. It could be favorable to one person and unfavorable to another. Samsung phones cost less and are

easier to operate than Apple iPhones. The review might be provided by any common individual. Despite the negative reviews, only high class individuals prefer to use Apple iPhones. The second review is similarly favorable (positive) to the poor but negative to the rich. [16].

1.7 Problem Statement

People share their viewpoints on different social media platforms like Facebook, X (Formally known as Twitter). Online communication becomes toxic if it contains hateful comments. Toxicity has become a prominent concern, especially in the context of social media. [17-18]. Traditional methods for detecting toxic comments often struggle to accurately identify and classify such content due to the nuanced nature of language and the evolving tactics used by malicious users. Therefore, there is a pressing need to develop more effective and robust approaches for automatically detecting toxic comments in online discussions [19-20].

1.8 Research Questions

Following are the research questions:

RQ1: How can the integration of advanced deep learning techniques improve the detection of toxic comments in online social media platform?

RQ2: How does the model perform in term of accuracy compared with existing approaches for toxic comment detection?

1.9 Research Objectives

The research key objectives are:

OBJ1: To identify the model, perform in term of accuracy compared with existing approaches for toxic comment detection

OBJ2: To identify the integration of advanced deep learning techniques can improve the detection of toxic comments in online social media platform.

1.10 Scope of Study

This research addresses the main issue of online toxicity within the context of the internet and social media platforms, emphasizing the need to understand and combat harmful behaviors in online interactions. It introduces a novel approach to toxicity detection, focusing initially on distinguishing between toxic and non-toxic comments. However, it acknowledges the potential for future research to expand upon this foundation. Such future investigations could involve categorizing various types of online toxicity beyond a binary classification, incorporating contextual factors to enhance detection accuracy, and exploring strategies to mitigate the impact of toxic content, including content moderation and user interventions.

Additionally, the scope could extend to addressing other forms of harmful content beyond comments, such as images or videos. By pursuing these avenues, the research aims to contribute to a more comprehensive understanding of online toxicity and promote the development of

strategies to foster healthier and more inclusive online communities. This research is contributed to Deep learning and NLP.

1.11 Summary

The exchange of information on social media has seamlessly integrated into daily existence, enabling individuals to voice their ideas and viewpoints on a wide array of subjects. Nevertheless, within the extensive volume of textual data exchanged, distinguishing valuable insights can prove to be quite a challenge. Sentiment analysis, or opinion mining, presents an automated solution by analyzing text to identify the expressed sentiment, be it positive or negative. The research objectives of this study involve identifying significant features in detecting toxic comments and developing a system for accurately identifying such comments. By exploring these aspects, the study aims to contribute to the domain of data mining and machine learning. Overall, sentiment analysis of toxic comments on social media has far-reaching implications in promoting a respectful online discourse and improving decision-making processes for businesses and organizations. The challenges in sentiment analysis require continuous research and advancements in the field to address the complexities of language and user behavior on social media platforms.

CHAPTER 2

LITERATURE REVIEW

In this chapter, the focus is on presenting the relevant background research for the study. This chapter delves into the existing studies pertaining to sentiment analysis. Furthermore, it provides insights into the data pre-processing techniques employed for data cleansing, and it also delves into the discussion of feature extraction and selection methods.

2.1 Sentiment Analysis Using Machine Learning Methods

Huq M et al. [20] used machine learning algorithm SVM, K-NN and Sentiment Classification algorithm to calculate the accuracy. Both techniques used twitter data set and same feature extraction techniqueswere used on dataset. Accuracy achieved with four different feature is 60% after adding key word-basedfeature, they achieved 70% accuracy.

Aufar et al. [21] examined two sentiment analysis algorithms: the Random Forest Algorithm and the Decision Tree Algorithm, using a dataset gathered from the Nokia Mobiles YouTube channel. TF-IDF (Term Frequency-Inverse Document Frequency) is utilized for feature extraction and gives superior accuracy.

Zhen Zuo et al. [22] did sentiment analysis and compared two supervised machine learning algorithms of large-scale stream review dataset using Decision Tree and Naïve Bayes. TF-IDF is used for feature selection. The result shows that for stream review dataset, Decision Tree achieved 75.05% accuracy and outperformed Naïve Bayer. The Naive Bayes technique was used by Ramdhani et al. [23] to perform Sentiment Analysis on comments made about the KFC Salted Egg

on Twitter and YouTube with accuracy of 84.50%. Muhammad et al. [24] applied Naïve Bayes algorithm with Support vector machine on the data collected on the comments obtained from YouTube educational videos the combination of both methods has given better accuracy level.

Using supervised learning techniques M. Shaheen et al. [25] performed sentiment analysis on reviews of mobile phone products using the Random Forest Classifier. Thw dataset was generated from amazon.com for customerreview about the unlocked mobile phones. In this study total 7 classifiers are compared one of which was Radom Forest Classifier. With 83% accuracy, the result shows that the Random Forest classifier outperformed every other classifier for the given mobile dataset. Long Short-Term Memory and NB-SVM performed equally well and displayed comparable accuracy of 73%. Conventional Neural Network displayed accuracy of 77%.

Goel et al. [26] applied Naïve Bayes and Recurrent neural network for Sentiment Analysis, which is the method for determining opinion in texts. Data gathered from tweets on Twitter were applied to algorithms. The Navies Bayes algorithm achieved 78.12% accuracy.

Sentiment analysis was carried out by Bhuiyan et al. [27] using YouTube comments as the dataset and the SentiStrength3 thesaurus. These videos cover under a variety of areas, including those related to science,technology, entertainment, and education. SentiStrength3 thesaurus study of the sentiment lexicon achieved a 75.45% accuracy rate. Sharma et al. [28] used machine learning based algorithm unsupervised lexicon-based approach to perform sentiment analysis on tweets. The result shows that positive words in tweets are large in count than negative words in tweets. It also indicates that suggested algorithm gave better accuracy than existing method.

Following the feature elimination process in tweets, Jung et al. [29] employed a multinomial naive Bayes approach for sentiment analysis. The researchers analyzed a dataset consisting of 1.6 million tweets sourced from the Sentiment 140 dataset, which were sorted into

positive, negative, or neutral sentiment categories. In their research, they find that the multinomial naive Bayes model attained 85% accuracy when they split the dataset into training and testing parts at a ratio of 9:1.

Athindran et al. [30] applied a naive Bayes approach to examine feedback from customers on Twitter. Tokenization and stemming were used in the preprocessing stages, and 77% accuracy was achieved by the naïve bayes model. These studies highlight the effectiveness of machine learning techniques, particularly the naive Bayes classifier, in the field of sentiment analysis.

Vanaja et al. [31] compared two well-known ML methods, support vector machine and naive Bayes, in the context of sentiment analysis for customer reviews on Amazon. The text data was represented by the researchers using the a priori approach, and stop words were eliminated during the preprocessing phase. The results showed that, with an accuracy of 84.42% compared to 81.42%, naive Bayes outperformed support vector machines.

Maximum entropy, support vector machines, and naive bayes were employed in a sentiment analysis experiment by Iqbal et al. [32] The two datasets utilized in the experiment were Sentiment 140 and IMDb. Preprocessing methods such as tokenization, lemmatization, and text cleaning were applied to these two datasets in the experiments. With 84% accuracy on the IMDb dataset and 82% accuracy on the Sentiment140 dataset, the maximum entropy strategy proved to be the most accurate when combined with bigram and unigram features.

Three machine learning algorithms—AdaBoost, SVM, and Decision Tree—were evaluated for performance by Rathi et al. [33] The Sentiment 140 dataset, the University of Michigan dataset, and the polarity dataset were all subjected to these algorithms' application. The Term Frequency-Inverse Document Frequency (TF-IDF) function was used to preprocess the text input. AdaBoost obtained an accuracy rate of 67%, decision trees acquired an accuracy rate of

84%, and support vector machines obtained an accuracy rate of 82%, according to the study's findings.

The multinomial naive Bayes model for sentiment analysis and the Support Vector Classifier (SVC) with linear kernels were the two machine learning techniques that Rahat et al. [34] Assessed for performance. They made use of a dataset that included 10,000 Tweets that were classified as neutral, positive, or negative. To prepare the text, a number of preprocessing methods were used, such as stop-word removal, URL removal, and stemming. The results showed that the multinomial naive Bayes model was surpassed by the Support vector classifier, which obtained an accuracy of 82.48%, while the naive Bayes model earned an accuracy of 76.56%.

In an experiment, Wongkar and Angdresey et al. [35] tested Support Vector Machine, K-Nearest Neighbor (KNN), and Naïve Bayes machine learning techniques for sentiment analysis. Twitter posts pertaining to the Indonesian presidential contenders for 2019 made up the dataset they used, which was split into an 80% training set and a 20% testing set. Included in the data pretreatment procedures are text mining, tokenization, and text parsing. The most accurate technique was Naïve Bayes, which obtained 75.58% accuracy according to the results of their research. Following at 73.34% accuracy was K-Nearest Neighbor, with the lowest accuracy being 63.99% for Support Vector Machine.

Gupta et al. [36] conducted a comprehensive study in which they evaluated the effectiveness of sentiment analysis using a variety of techniques such as decision trees, logistic regression, support vector machines, and neural networks. They used the Sentiment 140 dataset, which was preprocessed and transformed into TF-IDF features. The neural network model outperformed all other methods in their study, achieving the highest accuracy of 80%.

Harjule et al. [37] conducted a thorough comparison of SA methodologies, including deep

learning and machine learning methods. An ensemble strategy that integrated multinomial naive Bayes, logistic regression, and SVM with majority voting and LSTM was among the strategies they investigated. Their research made use of two datasets: Twitter US airline and Sentiment 140. Tokenization, converting text to lowercase, deleting stop words, removing URLs and hashtags, and removing punctuation were all performed on the datasets. LSTM achieved the highest accuracy on the Sentiment 140 dataset, achieving 82%. Support Vector Machine (SVM) performed the best on the Twitter US Airline Sentiment dataset, with an accuracy of 68.9%.

For the purpose to compare the efficacy of four distinct machine learning models for sentiment analysis— Support Vector Machine (SVM), Long Short-Term Memory (LSTM), BERT, and Multinomial Naive Bayes—Dhola and Saradva et al.[48]carried out a research. They made use of the 1.6 million tweets in the Sentiment 140 dataset. They used a number of methods, such as tokenization, stemming, stop-word removal, lemmatization and punctuation removal, during the data preparation stage. Their study's findings showed that the BERT model performed better than the other models in sentiment analysis, with an amazing accuracy rate of 85.4%.

A research was undertaken by Varshney et al. [38]to evaluate the efficacy of several machine learning approaches for sentiment analysis. Naive Bayes, logistic regression, stochastic gradient descent, and an ensemble model (Voting Classifier) that made use of majority voting were some of the methods they looked at. On the Sentiment140 dataset, they performed data preparation by eliminating null values, usernames, hyperlinks, and superfluous columns. After that, they used TF-IDF feature extraction and a lowercase text conversion to find the most significant characteristics. In their sentiment analysis, they found that the ensemble model (Voting Classifier) fared better than the other models, obtaining an astounding 80% recall rate for the positive class on the Sentiment140 dataset.

Table 2.1: Existing Literature Related To Machine Learning Methods

Sr.	Author	Dataset	Features	Methods	Result	Remarks
1	Huqmet et al [2017]	twitter data set		SVM, KNN, (SCA) Sentiment classification Algorithm	70%	SCA gives higher accuracy
2	Zhen zuo [2018]	stream review dataset	TF-IDF	Decision Tree, Naïve Bayes	75%	Decision Tree gives higher accuracy
3	M. shaheen et al. [2019]	Amazon dataset for customer review about unlocked mobile phones		Random Forest Classifier, CNN, NB-SVM, LSTM	83%	Random Forest Classifier gives higher accuracy
4	Goel et al. [2018]	Twitter dataset		RNN, Naïve Bayes	78.12%	Naïve Bayes gives higher accuracy
5	Jung et al. [2016]	Sentiment 140	Stemming	Naïve Bayes	85%	Naïve Bayes gives higher accuracy

6	Athindran [2018]	Twitter dataset	Tokenization, stemming	Naïve Bayes	77%	Naïve Bayes gives higher accuracy
7	Vanaia et al [2018]	Customer reviews on dataset	Stop word	Naïve Bayes, SVM	84.42	Naïve Bayes gives higher accuracy
8	Iqba et al. [2018]	Sentiment 140 and IMDb	Tokenization, lemmatization, text cleaning	Naïve Bayes, SVM, maximum entropy	84%	Naïve Bayes gives higher accuracy
9	Rathi et al [2018]	Sentiment 140, University of Michigan, and the polarity dataset	TF-IDF	Decision tree, AdaBoost, SVM	84%	Decision tree gives higher accuracy
10	Rahat et al [2020]	10,000 Tweets	Stemming, URL removal, stop word removal	Support Vector Classifier, Naïve Bayes	82.48	Support vector machine gives higher accuracy
11	Wongkar et al. [2019]	Twitter dataset	Tokenization, text prasing, text mining	SVM, KNN, Naïve Bayes	75.58%	Naïve Bayes gives higher accuracy

12	Gupta et al. [2019]	Sentiment 140 dataset	TF-IDF	Decision Tree, Logistic regression, SVM, neural network	80%	Neural Network gives higher accuracy
13	Harjule et al. [2020]	Twitter US airline and Sentiment 140 dataset	Stop word, case folding, tokenization	LSTM, logistic regression, Naïve Bayes, SVM	68.9%	SVM gives higher accuracy
14	Dhola and Saradva et al. [2021]	Sentiment 140 dataset	Lemmatization, stop-word and punctuation removal, tokenization, stemming	SVM, LSTM, multinomial naïve Bayes and BERT	85.4%	BERT model gives higher accuracy
15	Varshney et al. [2020]	Sentiment 140 dataset	Hyoerlink, null values, lower casing, TF-IDF	Logistic regression, ensemble modeling, and Naive Bayes (voting classifier)	80%	Ensemble model gives higher accuracy

2.2 Sentiment Analysis Using Deep Learning Methods

M. rani et al. [39] used LSTM, Bidirectional GRU and Transfer learning algorithms on comment dataset extracted from top rated Pakistani news channels. Preprocessing step was performed on comments dataset. First Stop words and punctuations were removed. Lemmatization is applied by using the "Word net" dataset. LSTM has an accuracy of 81%, bidirectional GRU has an accuracy of 76.46%, and model train with transfer learning has an accuracy of 84.89%.

Li et al. [40] research on customer comments used LSTM for detecting emotions to improve product quality and sales. They use both English and Chinese texts. English sentences have naturally participles to split into word. However, Chinese words do not have naturally participles English words have so they used word segmentation technique to transform Chinese word into participle.

Transfer learning was used by Tao et al. [41] on three datasets collected from social media including reviews of restaurants, different types of wines, and films reviews. Each dataset has accuracy of more than 81%. Transfer learning was used by Sadr et al. [42] to develop an attention-based convolution neuralnetwork and they achieved very good results. The proposed model use attention layer to extract the important features and suppressing the effect of unimportant feature. Nguyen L et al. [43] used transferlearning model and create a dataset for constructive and toxic speech detection with 10,000 human annotated comments. They achieved 78.59% accuracy for constructive data and 82.42% accuracy for toxic data.

Bidirectional GRU was applied to multi-label sentiment classification on consumer evaluations of mobile phones by Zhang et al. [44] To further optimize the use of lexical and emotional semantic information, the author also used SoftMax activation to the lexical layer, which may be viewed as an attention mechanism for word characteristics. Recurrent neural networks

(RNNs) were train on a huge dataset of product reviews from several categories for multilingual Sentiment Analysis by Can et al. [45] The model was also trained using English reviews that were domain-specific. The author used the RNN model. The RNN model's average accuracy versus various reviews is greater than 80%.

Adyan M. R [46] use deep neural network (RNNand CNN). Twitter dataset was used results shows that the accuracy of train set was 77.45% and test setwas 75.03%. Six different classifiers have been used to the Sentiment Strength Twitter Dataset, according to Ahuja et al. [47] Two features—TF-IDF and N-gram—are manually annotated the dataset. After doing sentiment analysis on these two, it is evident that TF-IDF feature gives better results than N-gram.

An inventive sentiment analysis model built on the transformer architecture was presented by Jing & Yang et al. [48] to capture significant sentence aspects, their model combined positional embedding with word-vector representation. They reduced the amount of layers and parameters in the model by simplifying it to simply use the encoder module of the transformer. Compared to conventional techniques like LSTM and CNN, their method showed a significant boost in classification accuracy, with improvements ranging from 0.3% to 1.0%. Their method also drastically lowered the overall number of model parameters. Their technique got a remarkable 76.40% accuracy rate on the NLPCC2014 Task2 dataset.

In a research, Maghsoudi et al. [49] used publicly accessible information to analyze Twitter data about insomnia throughout a range of time periods. They classified emotions as negative, positive or neutral by using pre-trained transformer models and the Dempster-Shafer theory (DST) for sentiment analysis. They tested their approach on a collection of 300 annotated Tweets in order to validate it, and they reached an outstanding 84% accuracy rate. Compared to the pre-pandemic time, there was a greater chance of people sending unfavorable Tweets regarding insomnia during the peri-pandemic period, according to their research using

logistic regression.

Table 2.2: Existing Literature Related To Deep Learning Methods

Sr.	Author	Dataset	Features	Methods	Result	Remarks
1	M. Rani et al. [2021]	Pakistani news channels	Stop words, lemmatization	LSTM, Bi-GRU and Transfer learning	86%	Transfer learning gives higher accuracy
2	Tao et al. [2020]	reviews of restaurants, different types of wines, and films reviews.		Transfer Learning	81%	Transfer Learning gives higher accuracy
3	Nguyen et al. [2021]	10,000 human annotated comments		Transfer Learning	78.59	Transfer Learning gives higher accuracy
4	Tyagi et al. [2020]	Sentiment 140 dataset	Removing stop word, punctuation, numerals, urls, case	Hybride model CNN, Bi-LSTM	81.20 %	Hybride model CNN, Bi-LSTM

			folding, stemming			
5	Jing and Yang et al. [2022]		Light-Transformer	Light-Transformer	76.40 %	Light-Transformer gives higher accuracy

2.4 Sentiment Analysis Using Hybrid Methods

Abd El-Jawad M [50] conducted a study where they assessed the effectiveness of various machine learning and deep learning algorithms, alongside a hybrid approach. They worked with a dataset comprising one million tweets. The results indicated an impressive accuracy rate of 83.7% for the hybrid system, underscoring its efficiency in sentiment analysis.

A hybrid model for sentiment analysis that combined the advantages of a Conventional Neural Network (CNN) and Long Short-Term Memory (LSTM) was presented by Goularas and Kamis et al. [51] 32,000 tweets from the International Workshop on Semantic Evaluation competition used as the dataset for their investigation. They changed the content to lowercase and eliminated emoticons, URLs, and special characters during the data preparation step. For text representation, their hybrid model used Word2Vec and GloVe, two distinct pre-trained word embeddings. Their findings showed that using GloVe embedding resulted in a 59% accuracy rate for the hybrid CNN and LSTM model.

A hybrid model for sentiment analysis was presented by Hossain, Bhuiyan, et al. [52] and merged

LSTM with a traditional neural network. They used a dataset of 100 restaurant ratings that they gathered from the Food Panda and Shohoz apps to carry out their investigation. They carried out preprocessing procedures to prepare the text data by removing unnecessary words and symbols. They then changed the text by embedding words using the Word2Vec technique. A Word2Vec pre-trained model embedding layer, a convolutional layer, a max-pooling layer, an LSTM layer, a dropout layer, and a classification layer were among the layers that made up the suggested technique. Their self-collected dataset yielded an accuracy rating of 75.01% for their hybrid model.

For sentiment analysis, Tyagi et al. [53] presented a hybrid model that paired a Bi-LSTM with a CNN. They made use of the Sentiment140 dataset, which comprised 1.6 million tweets expressing a range of emotions. The dataset was preprocessed in a variety of ways, including converting the text to lowercase and eliminating stop words, punctuation, numerals, URLs, and stop signs. Additionally, stemming was used. An embedding layer that made use of the GloVe pretrained model, a one-dimensional convolutional layer, a Bi-LSTM layer, fully connected layers, dropout layers, and a classification layer were among the layers that made up the suggested hybrid model. Their results show that the model performed with an accuracy of 81.20% on the Sentiment140 dataset.

In sentiment analysis, Tesfagergish et al. [54] proposed a two-step approach to emotion identification in sentiment analysis. In the first stage, an unsupervised zero-shot learning model based on a sentence transformer was employed to provide probabilities for 34 different emotions. In the second stage, a machine learning classifier was built using ensemble learning based on the sentiment labels. This unique hybrid semi-supervised approach got the highest accuracy of 87.3% when evaluated on the English SemEval 2017 dataset, confirming its usefulness. On three benchmark datasets, the technique's performance was investigated using a range of classifiers, including machine learning, neural networks, and ensemble learning. Notably, the strategy utilizing sets of 10 and 6 emotions obtained the greatest accuracy of 0.87 for binary sentiment

classification and 0.63 for three-class sentiment classification, respectively.

Table 2.3: Existing Literature Related To Hybrid Methods

Sr.	Author	Dataset	Features	Methods	Result	Remarks
1	Goularas et al. [2019]	Twitter dataset	Removing urls, emoticons, lower casing, word embedding, word2vec, glove	LSTM, CNN	59%	CNN and LSTM with Glove gives higher accuracy
2	Hossain et al. [2020]	100 restaurant ratings	Removing unimportant words, symbols, embedding technique word2vec	Hybrid model CNN, LSTM	75.01 %	Hybrid model CNN, LSTM
3	Maghsoudi et al. [2022]	300 annotated Tweets		Transformer, Dempster-shafer theory	84%	Transformer, Dempster-shafer theory gives higher accuracy
4	Tesfagergish et al. [2022]		Zero-shot transformer	Ensemble learning	87.3%	Ensemble learning gives higher accuracy

Machine learning techniques such as Support Vector Machine, Naïve Bayes, Random Forest and K-means are known for their fast execution but can struggle with word interpretation and word dependency [6]. However, deep learning techniques such as Gated Recurrent Units, Recurrent Neural Networks, and Conventional Neural Networks tend to provide higher accuracy when processing independent text characteristics, but they may have trouble accurately collecting syntactic information, which is the context and syntax of sentences [54]. The extensive literature review emphasizes how different machine learning and deep learning techniques are applied to sentiment analysis. However there are some problems that need to be fixed for better analysis [6].

2.5 Summary

The literature review presented a thorough examination of various machine learning (ML) and deep learning (DL) techniques and Hybrid approaches are applied to sentiment analysis across diverse datasets. Researchers explored methods ranging from traditional ML algorithms like Support Vector Machines (SVM) and Naïve Bayes to sophisticated DL models such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNN). Overall, the review underscores the ongoing evolution of sentiment analysis techniques and the importance of exploring innovative approaches to enhance accuracy and address inherent challenges in interpreting textual data.

CHAPTER 3

RESEARCH METHODOLOGY

This chapter explains how the study was carries out and why these selected methods are suitable. This chapter explains how experiment was carried out from where the data was collected, what data pre-processing, feature extraction and feature selection techniques were used for experiment. Model's implementation procedure and evaluation are describing in this chapter. In this study two advanced transformer-based model Distilbert and Roberta have been used along with Bi-LSTM layers. By merging these models, the aim to improve the performance of sentiment analysis.In recent years, deep learning models have been highly successful in sentiment analysis. These models can learn complex patterns and representations from text data, allowing them to capture the nuances and context of sentiment expressions.

3.1 A Key Component in NLP

In the Natural Processing Language (NLP) Sentiment analysis has become one of many important fields of computational studies. Text is divided into Positive and Negative sentiments through the process of sentiment analysis, or it involves the task of categorizing text as either positive or negative based on its inherent emotional tone. It deals with the mining and extracting of important information related to opinion or sentiment from a group for specific topics. Sentiment analysis has gained so much popularity among various fields such as politics, marketing, e-comer's, social media, business etc. [56].

3.2 Text Classification Steps

This section describes the brief of all activities and techniques employed in the classification of text of machine learning algorithms and hence the development of the system.

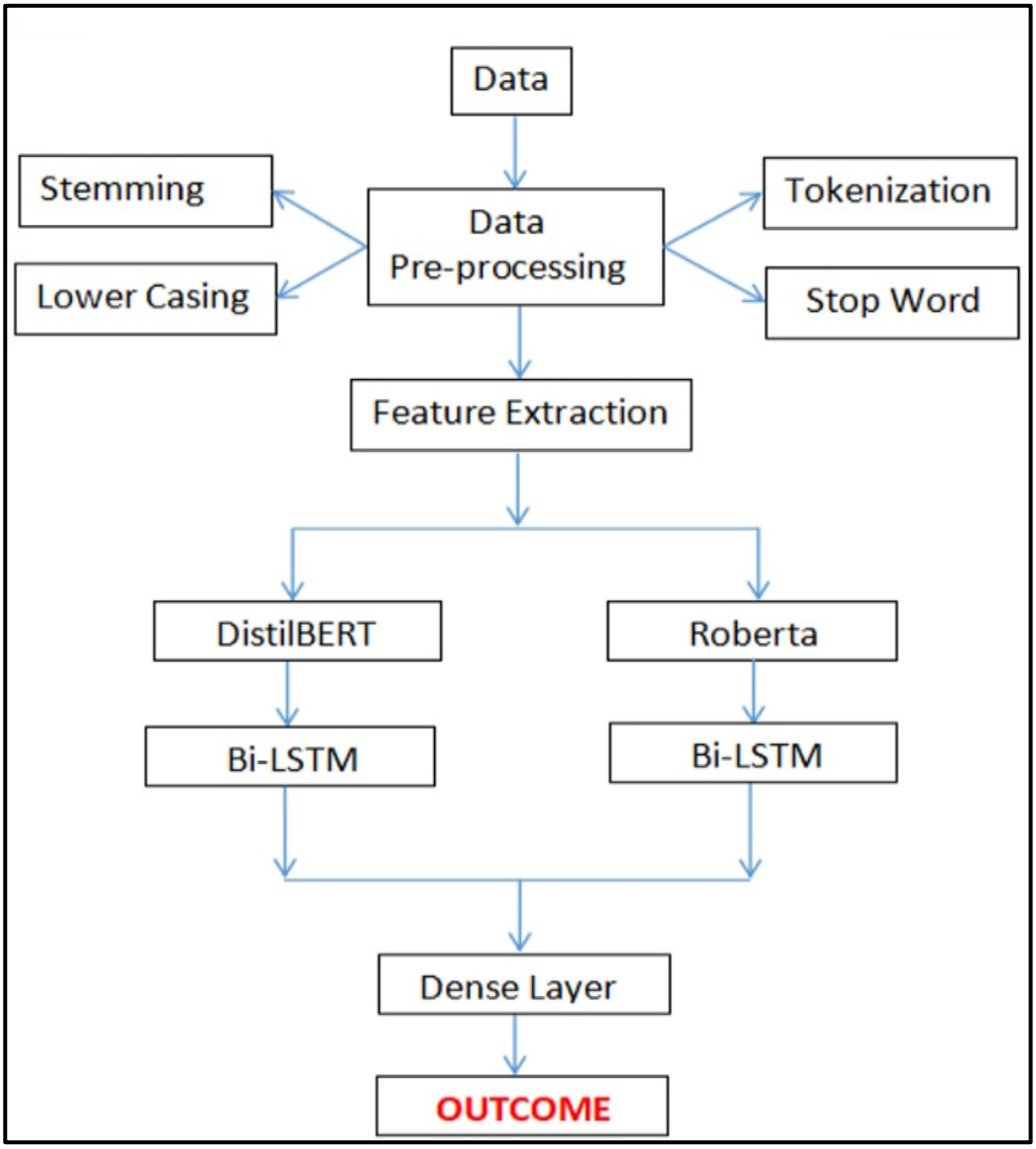

Figure 3.1: Text Classification Steps

3.3 Dataset Sentiment 140

A large amount of consumer sentiment data that was taken directly from Twitter is available in the Sentiment140 dataset, which is made available by Stanford University. It includes 1.6 million tweets with a variety of positive and negative remarks that were gathered from the platform.

Analyzing sentiments in this dataset can be challenging due to the concise and informal nature of tweets. Tweets often adopt a brief and casual writing style, which can potentially lead to ambiguity in determining the sentiment's positive or negative nature. The brevity of tweets, coupled with their informal style, might result in sentiment analysis methods struggling to precisely identify the sentiment due to the potential omission of crucial sentiment-bearing words or contextual cues [61].

3.4 Testing and Training Dataset

The Sentiment140 dataset can be used by consumers or companies to automatically analyze the sentiment of their brands, products, or topics on Twitter. The dataset consists of 1.6 million tweets in English, half of which have positive sentiment and the other half have negative sentiment. This dataset can be used to train sentiment analysis models, which can then be used to classify the sentiment of new tweets. An experiment uses a ratio of 8:2 in which 80% of data is used for training and 20% of data is used for testing purpose.

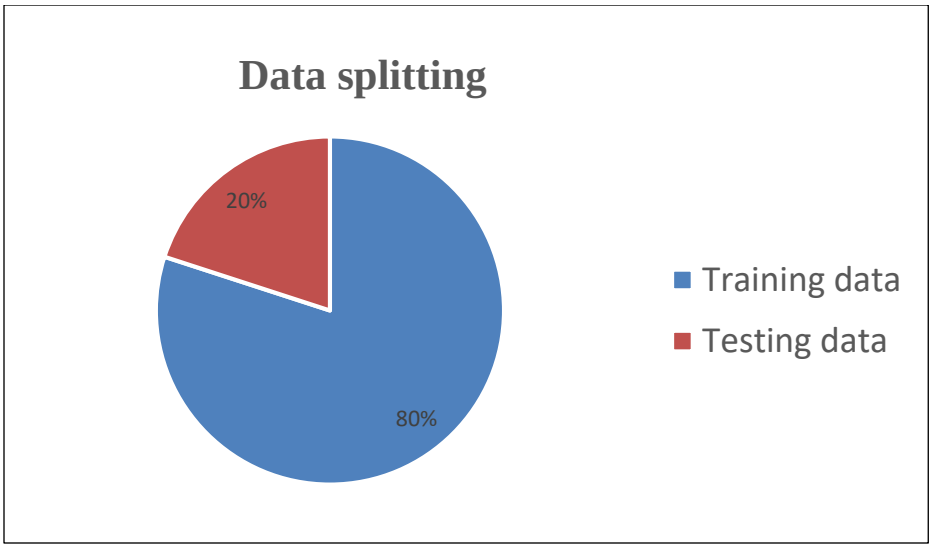

Figure 3.2: Data Splitting

3.5 Data Pre-processing

Data Pre-processing is a very important technique in the development of any machine learning system either it is related to image processing or text classification. The aim of data preprocessing is to convert raw data into a clean and appropriate format for analysis. When dealing with text data, this raw data may include elements like hashtags, punctuation, URLs within the text, or numerical values. Therefore, it is important to clean and refine data to ensure it gives us more efficient and better results.

Data preparation involves the transformation of data to a format suitable for input into a machine learning algorithm. It is essential to preprocess the dataset initially to enhance the performance of the model.

```
def clean_text(text):

    text = re.sub(r'#\w*','',text) #removing hashtags
    text = re.sub(r'@\w*','',text) #removing minshions
    text = re.sub(r'https?:\S*','',text) #removing http/s links
    text = re.sub(r'\d*','',text) #removing numbers
    text = re.sub(r'\W+',' ',text) #removing non words like (- , :)
    text = re.sub(r'_','',text) #removing underscore symbol
    text = re.sub(r'\s\S$','',text) #removing single characters at the end of the sentenses
    text = re.sub(r'^\s','',text) #removing spaces at the begain of the sentences
    text = re.sub(r'\s$','',text) #removing spaces at the end of the sentences
```

Figure 3.3: Data Pre-Processing

	Target	ID	Date	Query_flag	User	Tweet_text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

Figure 3.4: Dataset Before The Pre-Processing

In order to standardize the dataset following techniques helped us to achieve the relevant machine learning results.

3.5.1 Lowercasing

Lowercasing is a technique to transform words into lowercase. This is done in order to reduce varieties within same words that are actually same but due to having mixed (both lower and uppercase) words, they are considered as different. It is converting all the uppercase and lowercase text into lowercase to treat all words as same [57].

Now change all the uppercase text (capital words) into lower case text. It is used to ensure that all the text data are in same casing to make NLP task easier.

```
# Sample text for demonstration
text = "This Platform is straight forward and very user friendly"

# Tokenize the text into words
words = word_tokenize(text)

# Lowercase all words
lowercased_words = [word.lower() for word in words]

# Print the original text and the lowercased version
print("Original Text:", text)
print("Lowercased Text:", ' '.join(lowercased_words))
```

Figure 3.5: Lowercasing

The **lower ()** method is used on the **text** variable to convert all characters to lowercase, resulting in the **lowercased_text**. Now, all words in the text are in lowercase, regardless of how they were initially written.

```
Original Text: THIS PLATFORM IS STRAIGHT FORWARD AND VERY USER FRIENDLY
Lowercased Text: this platform is straight forward and very user friendly
```

Figure 3.6: Lowercasing Example

3.5.2 Tokenization

Tokenization is a technique of separating text into smaller tokens or words. After tokenizing data, removed the punctuations and numbers from data as they were of no need for us. It’s a widely used method when dealing with text classification. Simplified certain terms for understanding by returning them to their simplest forms [55].

3.5.3 Stop words Removal

Stopped using some terms since they are frequently found in articles, conjunctions, and prepositions such as “the”, “a”, “is” and “it” [59]. This is the process of removing those words from text which do not have any significant contribution in the classification of text and are used frequently within the text data. By minimizing noise and enhancing the extraction of sentiment-bearing words, the text data that results, free of stop words, improves the quality of sentiment analysis [60].

Stop words are typical words that are excluded from text. They are often removed from textual data because they do not carry relevant information. Stop words are common words that are frequently eliminated from text data because they contain no relevant information. These words are used often in sentences and are common in all languages, but they do not contribute significantly to the understanding or interpretation of the text's main content. During NLP processing stop words are removed.

Common examples of stop words in English include "a," "an," "the," "and," "in," "on," "is," "are," "for," "to," "of," "with," and etc.

```
# Stop Words
stop_words = stopwords.words('english')
```

Figure 3.7: Stop Word

This will output a list of frequently used stop words in English, which you may use to remove them from text data while performing other NLP tasks.

3.5.4 Stemming

Stemming is the process of reducing words to their base or root form by removing prefixes, suffixes, and inflectional endings. It aims to group related words together and normalize variations of the same word, which can help improve computational efficiency and reduce vocabulary size in natural language processing tasks.

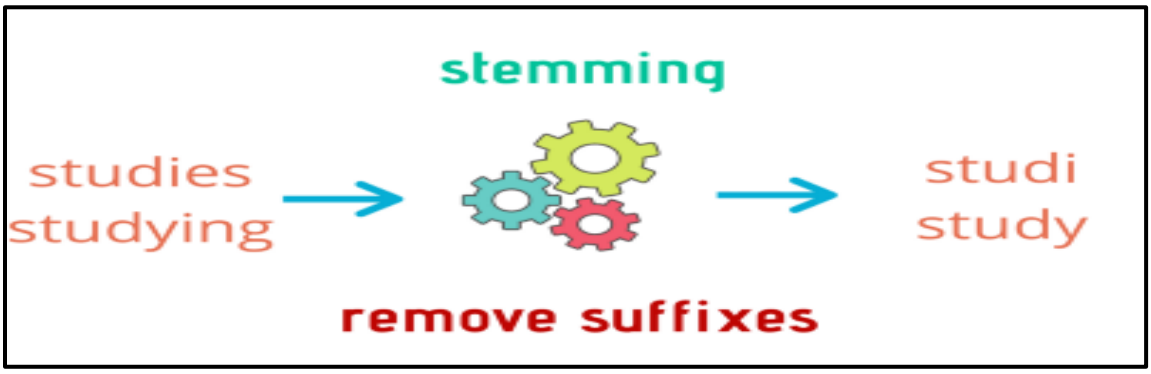

Figure 3.8: Stemming

Stemming is a natural language processing (NLP) method that breaks down words into their "stem," or root, form. The main objective of stemming is to normalize words such that several spellings of the same word are considered as one, which is useful for applications like sentiment analysis, text mining, and information retrieval. It improves the text efficiency and reduce the vocabulary size. For example, consider the following words

Creating -> Stem: Create
Creative -> Stem: Create
Created -> Stem: Create
Creates -> Stem: Create
Programming -> Stem: Program
Programmer -> Stem: Program
Programs -> Stem: Program
Runner -> Stem: Run
Running -> Stem: Run

```
#stemming

Stemmed_words = list()

snow_stemmer = SnowballStemmer(language='english')
for i in range(len(WORDS)):
    Stemmed_words.append(snow_stemmer.stem(WORDS[i]))

text = ' '.join(WORDS)

return text
```

Figure 3.9: Stemming Code

1. The code initializes an empty list **Stemmed_words** to store the stemmed words.
2. It creates a SnowballStemmer object with the language set to 'english'. SnowballStemmer is another stemming algorithm available in NLTK.
3. The code iterates through each word in the **WORDS** list and applies stemming using the SnowballStemmer. The stemmed word is then appended to the **Stemmed_words** list.
4. After the stemming process, the code joins the stemmed words back into a single string separated by spaces.
5. Finally, the function returns the resulting string.

After performing all the data preprocessing techniques all the noise from the data is cleaned.

	Target	ID	Date	Query_flag	User	Tweet_text	Cleaned_text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...	awww bummer shoulda got david carr third day
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...	upset update facebook texting might cry result...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...	dived many times ball managed save rest go bounds
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire	whole body feels itchy like fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....	behaving mad see

Figure 3.10: Dataset After Pre-Processing

3.6 Feature Extraction

As the study for detecting offensive communication is getting an increased attention so there is also an increase in the combination of algorithms and features explored. This section lay down the common feature extraction techniques are used in this system. Also call them as word embedding techniques.

In this study, utilized the DistilBERT and RoBERTa tokenizers to extract features from the textual data for sentiment analysis. The feature extraction process involves converting the raw text into numerical representations, often referred to as word embeddings or features, which can be processed by the respective models.

The feature extraction process begins by tokenizing the input text using the DistilBERT and RoBERTa tokenizers. These tokenizers split the text into subword units and assign numerical representations (token IDs) to each subword. Additionally, they generate token type IDs and attention masks. Token type IDs indicate whether a token belongs to the first or second sentence (in the case of pairwise tasks), while attention masks indicate the positions of actual tokens versus padding tokens.

3.7 Tokenization Process

The feature extraction process starts by splitting the input text into individual sentences or documents, depending on the desired granularity for analysis. Each sentence is then tokenized using the corresponding tokenizer, either DistilBERT or RoBERTa.

3.7.1 DistilBERT Tokenization

The DistilBERT tokenizer employs a Byte-Pair Encoding (BPE) algorithm to tokenize the text into sub word units known as Word Pieces. This tokenizer splits words into sub word units based on their frequency and occurrence patterns in the training corpus. The DistilBERT tokenizer also includes special tokens such as [CLS] (classification token) and [SEP] (separator token) to mark the beginning and separation of sentences.

3.7.2 RoBERTa Tokenization

Similarly, the RoBERTa tokenizer uses a variant of BPE called Sentence Piece for tokenization. This tokenizer enables a more fine-grained sub word tokenization approach. It incorporates special tokens like [CLS], [SEP], and [PAD] (padding token) to mark sentence boundaries and handle variable-length sequences.

3.7.3 Encoding

After tokenization, the tokenizer assigns a unique numerical identifier called a token ID to each token. Additionally, it assigns a token type ID to each token, indicating whether it belongs to the first sentence or the second sentence (in the case of pairwise tasks). The tokenizer also generates attention masks, which are binary masks indicating the positions of actual tokens versus padding tokens.

3.7.4 Padding

To enable efficient batch processing, the input sequences are padded or truncated to a fixed length. Padding involves adding special [PAD] tokens to ensure that all sequences have the same length. The padding tokens have no impact on the model's attention mechanism or predictions. Truncation is applied when the input sequence exceeds the maximum allowed length, and tokens beyond this limit are discarded.

3.7.5 Obtaining Features

Once the text has been tokenized, encoded, and padded, obtain the corresponding token IDs, token type IDs, and attention masks as the input features for the respective models

(DistilBERT and RoBERTa). These features capture the semantic information and contextual relationships within the text data.

3.7.6 Model Execution

The token IDs, token type IDs, and attention masks are passed through the DistilBERT and RoBERTa components of proposed model architecture separately. Each component processes the features through its transformer layers, capturing contextual information and generating output representations. These representations encode high-level contextual information and semantic meaning of the input text.

3.7.7 Bi-LSTM Processing

After the DistilBERT and RoBERTa components, the output representations are passed through Bi-LSTM layers. The Bi-LSTM layers process the merged representations in a sequential manner, capturing the sequential dependencies and context of words in the text. This sequential processing enhances the model's understanding of the order and context of words, allowing it to capture longer-range dependencies that are relevant for sentiment analysis.

3.7.8 Fusion and Concatenation

After the Bi-LSTM layers, the outputs of the DistilBERT and RoBERTa components are merged by concatenating them. This fusion of representations combines the high-level contextual information learned by the transformer models with the sequential processing capabilities of the Bi-LSTM layers. The merged representation becomes the input for subsequent layers or for generating sentiment predictions.

3.7.9 Feature Concatenation

In the multi-head model architecture, the features obtained from DistilBERT and RoBERTa is concatenated before being passed to the subsequent layers. This merging of features enables the combination of knowledge and representations learned by both models, potentially enhancing the overall sentiment analysis performance.

By utilizing the DistilBERT and RoBERTa tokenizers, effectively convert the raw text into numerical representations that capture semantic and contextual information for sentiment analysis. The resulting features, comprising token IDs, token type IDs, and attention masks, serve as input to the respective models. This feature extraction approach forms the foundation for subsequent processing and prediction of sentiment labels, leveraging the power of pre-trained language models and sequential processing capabilities of the Bi-LSTM layers in the multi-head architecture.

3.7.10 Sentiment Prediction

The merged representation is then passed through fully connected layers or other classification layers to generate sentiment predictions. These layers learn to map the merged representation to sentiment labels, such as positive, negative, through a training process that minimizes a suitable loss function, such as categorical-cross-entropy loss. The model is trained using labeled data, where the sentiment labels are provided for the training examples.

3.8 Classification

As a branch of artificial intelligence (AI), machine learning focuses on building models and algorithms that can learn from data and then use that information to anticipate or make choices. The core objective of machine learning is to empower computers to autonomously discern patterns

and connections within a provided dataset. A subfield of machine learning known as "deep learning" focuses on building artificial neural networks that closely resemble the composition and operations of the human brain. Deep learning algorithms, also known as deep neural networks, use numerous layers of linked nodes, also known as artificial neurons or units, to learn hierarchical data representations. Their proficiency is in handling complicated tasks such as speech and image recognition, natural language processing, and other uses.

The primary benefit of deep learning is that it can automatically extract complex features from raw data and learn them on its own, eliminating the need for human feature engineering. This makes it especially effective for high-dimensional applications like speech or image recognition that need big datasets. In implementation, propose a multi-head model for sentiment analysis that use pre-trained language models like DistilBERT [63] and RoBERTa [64], Transformer-based model with sequential processing capabilities of Bi-LSTM layers [65]. By merging these models, the aim is to leverage their complementary strengths and improve the performance of sentiment analysis.

The decision to incorporate a transformer model into the proposed architecture is based on its proven effectiveness in capturing broader context and semantic meanings in natural language processing (NLP) tasks. Transformer-based models like DistilBERT and RoBERTa have shown impressive performance across various NLP tasks by comprehending intricate linguistic nuances and extracting significant features from text data. By utilizing pre-trained transformer models, the sentiment analysis model can benefit from their extensive linguistic representations and profound understanding of context, laying a strong foundation for precise sentiment analysis.

However, while transformers excel at grasping long-range dependencies and semantic information, they may not fully capture the sequential relationships within the text. This is where Bi-LSTM layers complement the architecture. Bi-LSTM layers are well-suited for modeling

sequential data and capturing the temporal dependencies between words in a sentence. By integrating Bi-LSTM layers following the transformer model, the sentiment analysis architecture can effectively capture both overarching context and localized sequential information, leading to a more comprehensive understanding of the text and enhanced sentiment analysis performance.

In summary, the hybrid approach of combining a transformer model with Bi-LSTM layers leverages the strengths of both techniques: the transformer's ability to capture broader context and semantic representations, and the Bi-LSTM's proficiency in modeling sequential dependencies. This combined strategy enables the sentiment analysis model to achieve superior performance by effectively capturing various facets of language comprehension, resulting in more accurate sentiment analysis outcomes.

By combining transformer-based models with BiLSTM layers, the proposed architecture aims to harness the complementary strengths of both approaches. While transformer-based models handle the broader context and semantic understanding, BiLSTM layers capture the finer sequential nuances within the text. This synergistic combination allows the model to achieve more robust and accurate sentiment analysis, outperforming traditional approaches that rely solely on one type of architecture. Additionally, incorporating BiLSTM layers can also help mitigate some of the limitations of transformer-based models, such as handling rare expressions, contextually complex samples, and sarcasm. Overall, the integration of BiLSTM layers into the model architecture enhances its ability to capture both global context and sequential dependencies, thereby improving its performance in sentiment analysis tasks

3.9 Transformer Model

A transformer model is a neural network architecture capable of automatically converting a given input into a corresponding output. It is designed to process the sequence of data like

sentences in natural language, and has evolved into a fundamental component for a range of natural language processing (NLP) applications.

A transformer model consists of two main components: an encoder and a decoder. The encoder takes an input sequence, such as a sentence, and transforms it into a sequence of hidden representations. The decoder then takes these hidden representations and generates an output sequence, such as a translation or a summary. Both the encoder and decoder are composed of multiple layers of identical units called encoder-decoder blocks. Each encoder-decoder block consists of two main components: a self-attention mechanism and a feed-forward network.

The self-attention mechanism is the key innovation of the transformer model. It allows the model to learn long-range dependencies between words in a sentence. The self-attention mechanism works by calculating the attention score for each word in the input sequence. The attention score for a word is a measure of how important that word is for understanding the meaning of the sentence. The attention scores are then used to weight the hidden representations of the words, so that the most important words have the most influence on the output sequence.

The feed-forward network is a simple neural network that is applied to each word in the hidden representation sequence. It consists of two fully connected layers with a ReLU activation function in between. The feed-forward network helps the model to learn more complex relationships between words.

A transformer model is like a clever language expert that can understand the relationships between words and sentences. It uses a special mechanism called "attention" to focus on the most important parts of the input text, just like you would when reading a book. This allows the transformer model to capture the overall meaning of the text and perform tasks like machine translation, text summarization, and question answering.

The Transformer architecture has demonstrated significant effectiveness across diverse NLP tasks, encompassing machine translation, text summarization, and sentiment analysis. Additionally, it has been modified and expanded for applications outside NLP, including image processing and reinforcement learning. A notable strength of the Transformer architecture lies in its capacity to apprehend long-range dependencies within data, facilitated by the self-attention mechanism. This attribute has played a pivotal role in its extensive adoption and contribution to the advancement of state-of-the-art NLP models like BERT and similar innovation.

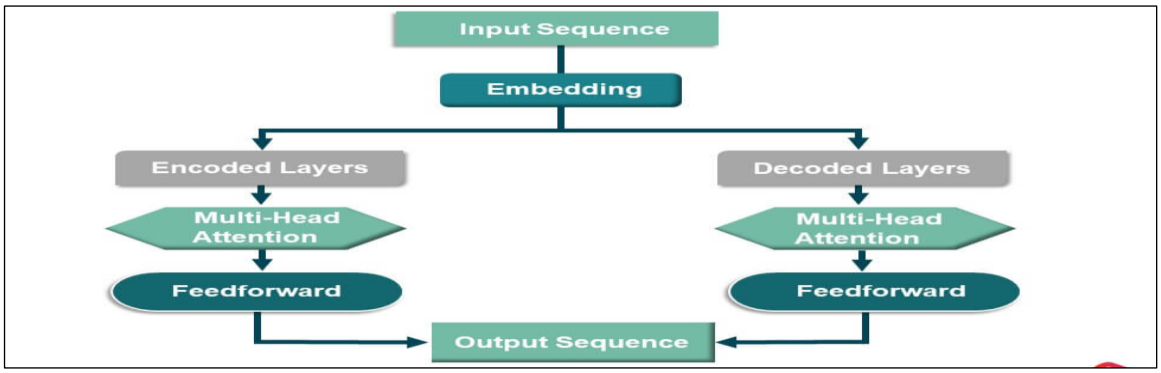

Figure 3.11: Transformer Model

3.10 BERT

BERT Bidirectional Encoder Representations from Transformers is a powerful natural language processing (NLP) model developed by Google. It's like a language understanding superhero for computers. It stands out as an intelligent language model that acquires knowledge from extensive text, comprehends words bi-directionally, focuses on context, and excels at tasks such as answering questions and comprehending sentence meanings. Its significance lies in its profound impact on advancing computers' ability to understand human language.

3.11 RoBERTa

RoBERTa is another powerful transformer-based model that builds upon the BERT architecture. It incorporates additional training techniques and modifications to further improve its performance. RoBERTa captures contextual information through self-attention mechanisms, allowing it to understand the nuances and context of sentiment expressions. Like DistilBERT, RoBERTa is pre-trained on large corpora and fine-tuned for the sentiment analysis task.

3.12 DistilBERT

DistilBERT serves as the initial feature extractor in model. This version is a condensed variant of the BERT (Bidirectional Encoder Representations from Transformers) model, maintaining most of its effectiveness while requiring less computational resources. DistilBERT is a transformer-based model that utilizes self-attention mechanisms to capture contextual information and semantic representations from text. It has been pre-trained on large corpora, enabling it to learn the relationships and representations of words and sentences.

3.13 Bi-directional long short-term memory (Bi-LSTM)

Long short-term memory is a special kind of RNN model. LSTM has ability to captured long term dependencies than RNN. LSTM can store backward information it only stores forward information The concept of Bi-directional Long Short-Term Memory (Bi-LSTM) was introduced to enable access to both backward and forward features. While traditional LSTM only allows input in a single forward direction, Bi-LSTM permits input in both directions simultaneously – forward and backward. This enhances the model's ability to capture contextual information from both past and future contexts.

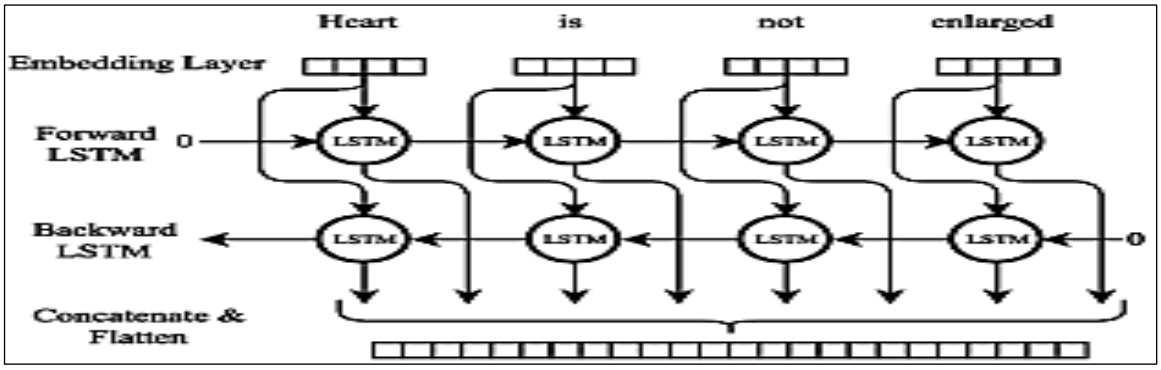

Figure 3.12: Architecture Of A Bi-LSTM **[66]**

3.14 Training and Fine-Tuning

The model is trained in an end-to-end manner, with the objective of minimizing the loss between the predicted sentiment labels and the ground truth labels. The training process involves forward and backward propagation, adjusting the weights and parameters of the model based on the computed gradients. The model is fine-tuned using the sentiment analysis task-specific data to adapt the pre-trained representations for sentiment analysis.

3.15 Experimental Setup

3.15.1 Software and Libraries Sklearn (Scikit-learn)

Python-based open-source library with a wide range of tools and machine learning methods for a range of machine learning tasks. This consists of tools for tasks like feature extraction and text categorization.

3.15.2 Spacy

A library for Python that includes functions like dependency parsing, part-of-speech tagging, and named entity recognition and offers advanced functionality for natural language processing.

3.15.3 FastText

A Python library designed for word representation and the categorization of text.

3.15.4 Pandas

A library in python. It is powerful open-source data manipulation and analysis library. For effective manipulation and analysis of structured data, it offers data structures and functions.

3.15.5 NLTK

It is used to import natural processing toolkit (NLTK) library in python. It provides wide range of tools and resources. It is a powerful library due to its large coverage it supports multiple language.

3.15.6 Google Collab

Google offers a cloud-based integrated development environment (IDE) which allows user to write, run and collaborate online Python code.

3.15.7 Jupyter notebook

Jupyter Notebook provides an interactive and flexible environment for data analysis, visualization, and experimentation with code. It is widely used in data science and machine learning projects for its ease of use and ability to combine code, visualizations, and narrative in a single document.

3.15.8 Transformer

A python library which provides many pre-trained transformer models. It performs various natural processing (NLP) task.

3.16 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is an essential step in any data analysis project, including sentiment analysis. It involves thoroughly examining the dataset to gain a deep understanding of its characteristics, patterns, and potential obstacles. EDA enables us to make informed decisions about data preprocessing and feature engineering. In this section, the EDA conducted for sentiment analysis project using a multi-head model with DistilBERT, RoBERTa, and Bi-LSTM layers is discussed.

To begin the EDA, it's important to provide an overview of the sentiment analysis dataset. Describe the dataset's size in terms of the number of samples it contains and the distribution of sentiment labels (e.g., positive, negative). Additionally, highlight any specific characteristics of the dataset, such as its source, language, or domain.

Next, analyze the text data present in the dataset to gain insights into its composition and structure. Consider several aspects.

3.16.1 Text Length

Investigate the distribution of text lengths, whether measured in words or characters, to understand the variability of the data. This information can be helpful in determining appropriate input sequences for the models.

3.16.2 Word Frequency

Identify the most common and least common words in the dataset. This analysis can reveal important keywords or domain-specific terms that might significantly impact sentiment classification.

3.16.3 Stop words

Determine the presence of stop words (commonly used words like "the," "is," "and") and evaluate their impact on sentiment analysis. Consider whether removing or retaining stop words would be beneficial for the task at hand.

3.17 Multi-Head Model for Sentiment Analysis

In this research, a multi-head model architecture has been utilized, which combines DistilBERT, RoBERTa, and Bi-LSTM layers for the task of sentiment analysis. This approach harnesses the strengths of these models to capture semantic information from the text, while also incorporating sequential processing capabilities to gain a deeper understanding of the order and

context of words. This enables the model to process sequences of words sequentially, thereby enhancing its ability to comprehend word order and context effectively.

In the development of Sentiment Analysis prediction project, it was aimed to create a powerful and sophisticated model architecture that could leverage the full potential of transformer-based models and deliver superior performance. Transformer-based models have revolutionized natural language processing tasks by effectively capturing contextual information and long-range dependencies in text.

To join the strengths of multiple transformer models, a unique approach is employed by incorporating two separate branches in proposed architecture. The first branch utilized the DistilBert model, which is a lightweight and efficient variant of the original BERT (Bidirectional Encoder Representations from Transformers) model. DistilBert is known for its ability to provide accurate embeddings and semantic understanding of text while requiring significantly fewer computational resources. This allowed proposed model to quickly process large volumes of data without compromising on performance.

The second branch of proposed architecture integrated the powerful Roberta model. Roberta is an optimized version of BERT that employs a larger number of training steps and significantly more data, leading to even better language understanding and representation capabilities. By combining DistilBert and Roberta, it was aimed to capitalize on their complementary strengths and extract a diverse range of linguistic patterns and nuances from the input text.

However, the importance of sequential information in understanding the context of a sentence effectively was recognized. To capture this sequential understanding, Bi-LSTM (Bidirectional Long Short-Term Memory) layers following each branch of the transformer models

were introduced. Bi-LSTMs are a type of recurrent neural network that can process the text in both forward and backward directions, enabling the model to gain a deeper comprehension of the tokenized text and the relationships between words. This helped the model to effectively remember important information and understand the sequence's order and context more profoundly.

The true magic of proposed architecture came into play during the merging process of the DistilBert and Roberta branches. The outputs of both branches were combined to create a consolidated vector. This merging process was thoughtfully designed to allow the model to synergistically leverage the knowledge extracted from both transformer models. By blending the insights from DistilBert's efficiency and Roberta's powerful language understanding, the model achieved a comprehensive representation of the input text, which was then passed through a fully connected layer for sentiment classification.

The fusion of these cutting-edge techniques, namely transformer-based models, Bi-LSTM layers, and a carefully designed merging process, enabled proposed Sentiment Analysis model to deliver state-of-the-art results. The model exhibited a remarkable ability to comprehend the complexities of natural language, capture subtle sentiments, and make accurate predictions across various domains and contexts.

3.18 DistilBERT

DistilBERT is a transformer-based model that utilizes self-attention mechanisms to capture contextual information and semantic representations from text. It has been pre-trained on large corpora, enabling it to learn the relationships and representations of words and sentences. This function effectively provides a way to load a pre-trained DistilBERT model fine-tuned for sentiment analysis, create a modified version of the model for downstream sentiment analysis, and return the input layer, tokenizer, and model outputs for further use in sentiment analysis tasks.

```
def loadDistilBert():

  # Load the pre-trained model
  model_name = 'distilbert-base-uncased-finetuned-sst-2-english'
  config = DistilBertConfig.from_pretrained(model_name)
  model = TFDistilBertModel.from_pretrained(model_name, config=config)
  # Tokenize the input text
  tokenizer = DistilBertTokenizer.from_pretrained('distilbert-base-uncased')

  # Remove the classifier layers
  model.layers.pop()

  # Build a new model without the classifier
  inputs = tf.keras.Input(shape=(512,), dtype=tf.int32, name="Input_Distilbert")

  outputs = model(inputs)[0]
  outputs = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(128, return_sequences=True))(outputs)
  outputs = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(64))(outputs)
  outputs = tf.keras.layers.Dense(64, activation='relu')(outputs)

  return inputs, tokenizer, outputs
```

Figure 3.13: Def Loaddistilbert()

3.19 RoBERTa

RoBERTa captures contextual information through self-attention mechanisms, allowing it to understand the nuances and context of sentiment expressions. Like DistilBERT, RoBERTa is pre-trained on large corpora and fine-tuned for the sentiment analysis task. This function provides a way to load a pre-trained RoBERTa model fine-tuned for Twitter sentiment analysis, and it sets up a multi-layer architecture that includes bidirectional LSTMs for processing input sequences and extracting sentiment-related features. This function is used to obtain the input layer, tokenizer, and model outputs for sentiment analysis tasks.

```
def loadRoberta():

  model_name = "cardiffnlp/twitter-roberta-base-sentiment-latest"
  tokenizer = RobertaTokenizer.from_pretrained(model_name)
  base_model = TFRobertaModel.from_pretrained(model_name)

  # Create a new input layer
  input_ids = tf.keras.Input(shape=(512,), dtype=tf.int32, name="Input_Roberta")

  # Connect the input layer to the desired layer of the base model
  outputs = base_model(input_ids)[0]  # Using [0] to access the last hidden state
  outputs = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(128, return_sequences=True))(outputs)
  outputs = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(64))(outputs)
  outputs = tf.keras.layers.Dense(64, activation='relu')(outputs)

  return input_ids, tokenizer, outputs
```

Figure 3.14: Def Loadroberta()

3.20 Bi-LSTM Layers

After the DistilBERT and RoBERTa components, Bi-LSTM (Bidirectional Long Short-Term Memory) layers has been introduced into the model architecture. Bi-LSTMs are recurrent neural networks that process the input text in both forward and backward directions. This bidirectional processing enables the model to capture dependencies and context from both preceding and succeeding words. The Bi-LSTM layers aim to enhance the model's ability to understand the sequential nature of sentences and capture longer-range dependencies that might influence sentiment expressions.

```
def defineModelArchitecture():

  tokenizer_roberta = list()
  inp_roberta, tokenizer_roberta, roberta = loadRoberta()
  inp_distilbert, tokenizer_distilbert, distilbert = loadDistilBert()

  #Merging model A and B
  my_model = tf.keras.layers.concatenate([roberta, distilbert],name="concatenated_layer")

  my_model = tf.keras.layers.Dense(64, activation='relu')(my_model)
  my_model = tf.keras.layers.Dense(32, activation='relu')(my_model)

  output = tf.keras.layers.Dense(2, activation='softmax')(my_model)

  #Model Definition
  combined_model = tf.keras.Model(inputs=[(inp_roberta,inp_distilbert)], outputs=[output], name = "Combined_Model")

  return combined_model, tokenizer_roberta, tokenizer_distilbert
```

Figure 3.15: Def Definemodelarchitecture_A ()

defineModelArchitecture_A and B that builds a combined model by merging the output representations of two pre-trained models: RoBERTa and DistilBERT. This combined model is then further processed to make sentiment predictions. This function defines a combined model architecture that leverages features learned by both RoBERTa and DistilBERT. The models are concatenated, and the combined features are processed through dense layers before making sentiment predictions using a softmax output layer. This approach can help improve sentiment analysis by exploiting the strengths of both models.

```
ms=tf.distribute.MirroredStrategy(devices = ['GPU:0', 'GPU:1'])
with ms.scope():
    model, tokenizer_roberta, tokenizer_distilbert = defineModelArchitecture()

for layer in model.layers[:-10]:
  layer.trainable = False

for layer in model.layers:
  print("{}: {}".format(layer, layer.trainable))

with ms.scope():
    model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

model.summary()
```

Figure 3.16: Def Definemodelarchitecture_B ()

It defines a MirroredStrategy for distributed training on multiple GPUs, creating a combined model, setting certain layers in the model to be non-trainable, compiling the model, and displaying a summary of the model. In summary, this code sets up distributed training using the MirroredStrategy on two GPUs, creates a combined model with tokenizers, freezes most of the model's layers for fine- tuning, compiles the model with optimization settings, and displays a summary of the model's architecture. This is a common procedure for distributed training and fine-tuning pre-trained models for specific tasks. It reads text data and labels from a CSV file, preprocesses the text data by tokenizing it for both RoBERTa and DistilBERT models, and one-hot encodes the labels. The preprocessed text data is now ready to be used as inputs for the RoBERTa and DistilBERT models. This is a common data preparation step for natural language processing (NLP) tasks where text data is used as input to deep learning models.

```
import os
checkpoint_path = "Distil_Roberta_BiLSTM_epoch_{epoch:02d}.h5"

cp_callback = tf.keras.callbacks.ModelCheckpoint(
   checkpoint_path, verbose=1, save_weights_only=True,
   # Save weights, every epoch.
   save_freq='epoch')


with ms.scope():
    model.fit(x=[inputs_roberta['input_ids'],inputs_distilbert['input_ids']], y=np.array(lables), epochs=5, batch_size=128, callbacks = [cp_callback],)


model.save("Distil_Roberta_BiLSTM.h5")
```

Figure 3.17: Distill_Roberta_Bilstm.H5

It is used to train a deep learning model (presumably the combined model created earlier) using a MirroredStrategy for distributed training across multiple GPUs. It also includes model check pointing to save the model's weights during training and ultimately saves the entire model. In summary, this code trains a deep learning model with distributed training across multiple GPUs using a MirroredStrategy. It saves model checkpoints during training and, after completing training

for the specified number of epochs, it saves the entire trained model to a file named "Distil_Roberta_BiLSTM.h5." This allows you to use the trained model for tasks like sentiment analysis on toxic comments as mentioned in your thesis topic.

Python libraries and their components are used in code. These libraries include TensorFlow for deep learning, the Hugging Face Transformers library for working with transformer models, NumPy for numerical operations, and Pandas for data manipulation and analysis. Tokenizer_roberta object is an instance of the RobertaTokenizer class, which is used for tokenizing text data specifically for RoBERTa models. The from_pretrained method is used to load a pre-trained tokenizer. The path to the pre-trained tokenizer is constructed using os.path.join(os.getcwd(), 'Roberta_Tokenizer'), which concatenates the current working directory with the folder name 'Roberta_Tokenizer' to form the path to the pre-trained tokenizer. Tokenizer_distilbert object is an instance of the DistilBertTokenizer class, used for tokenizing text data for DistilBERT models. Once again, the from_pretrained method is used to load a pre-trained tokenizer, and the path is constructed in a similar manner to the previous line, this time for the 'Distilbert_Tokenizer' folder. Model = load_model(...): This line initializes a variable named model and assigns it the result of loading a Keras model.

The load_model function is used to load a previously trained model from a file. It takes two main arguments: os.path.join(os.getcwd(),'DistilBert_Roberta_BiLSTM.h5'): This part constructs the full file path to the HDF5 file containing the pre-trained model. os.getcwd() gets the current working directory, and os.path.join() combines it with the filename 'DistilBert_Roberta_BiLSTM.h5' to create the complete path to the model file. custom_objects={"TFRobertaModel":transformers.TFRobertaModel,'TFDistilBertModel':transformers.TFDistilBertModel}: This part specifies custom objects needed during the loading process. The load_model function uses these custom objects to correctly handle any layers or components that may not be standard Keras layers. In this case, it maps the custom layer names used in the

model file to their corresponding Hugging Face Transformers model classes. It tells Keras how to handle these custom objects when loading the model. model.summary(): Once the model is loaded, you call the summary() method on the model object. This method prints a summary of the model's architecture, including the types of layers, their output shapes, and the number of parameters in each layer. It provides valuable information for understanding the structure and complexity of the loaded model.

pd.read_csv('/content/gdrive/MyDrive/MS Sentiment/Testing/testing_dataset.csv'): This line of code uses the Pandas library (pd) to read a CSV file. The file being read is located at the specified path, which is '/content/gdrive/MyDrive/MS Sentiment/Testing/testing_dataset.csv'. This path points to a file in the Google Drive directory. read_csv is a Pandas function for reading CSV files, and it returns the data as a DataFrame, which is a two-dimensional tabular data structure. df.head(): Once the CSV file has been read and stored in a DataFrame named df, this line is used to display the first few rows of the DataFrame. The head () method is called on the DataFrame, and it defaults to displaying the first 5 rows. This provides an initial view of the data, including the column names and the values in those columns.

Tokenizing the text data in the 'Cleaned_text' column of the DataFrame df using the RoBERTa tokenizer. The parameters used are as follows: tokenizer_roberta: This is the RoBERTa tokenizer that you previously defined. list(df['Cleaned_text']): This line takes the 'Cleaned_text' column from the DataFrame df and converts it into a Python list. The RoBERTa tokenizer expects text data in list format. add_special_tokens=True: This parameter specifies that special tokens (such as [CLS] and [SEP]) should be added to the input text. return_tensors='tf': This parameter indicates that you want the output tensors to be in TensorFlow format. padding='max_length': It specifies that the text sequences should be padded to the maximum length defined by max_length. truncation=True: This parameter ensures that sequences longer than max_length are truncated. max_length=512: The maximum length for each sequence is set to 512 tokens. The result of this

operation is stored in the embeddings_roberta variable, which likely contains the tokenized and preprocessed text data ready for input into a RoBERTa model.

Similarly, you perform a similar operation for DistilBERT: In this case, DistilBERT tokenizer (tokenizer_distilbert) is used to tokenize the 'Cleaned_text' data from the DataFrame df. The parameters are the same as in the previous example. Both embeddings_roberta and embeddings_distilbert likely store the tokenized and preprocessed text data, which can be used as input to RoBERTa and DistilBERT models for various natural language processing tasks. These embeddings contain the encoded representations of the text that the models can work with.

This code is making predictions using a machine learning model and then extracting the predicted classes. The code takes the preprocessed input data, embeddings_roberta and embeddings_distilbert, feeds them into the machine learning model (model) for prediction, and then extracts the predicted class labels (preds) by choosing the class with the highest probability score for each sample. This is a common process in supervised machine learning for making predictions on input data. New DataFrame, df_pred, is created to store the true target values and the predicted values, and then it displays the first few rows of this DataFrame. This code is used to create a DataFrame (df_pred) that contains both the true target values ('y_test') and the predicted values ('y_pred') from your machine learning model. It provides a convenient way to compare and analyze the model's performance by examining how well the predictions align with the actual target values. The head() method is used to display the initial rows of this DataFrame to get an initial view of the data. After executing this line of code, the df_pred DataFrame will be saved as 'df_pred.csv' in the specified directory, and you can access the file at that location for further analysis or sharing the results.

Importing various functions and classes from the scikit-learn library (sklearn) for evaluating and reporting the performance of machine learning models. It is used for evaluating

the performance of classification models in machine learning and are essential for assessing how well a model performs in various scenarios. Creating and displaying a confusion matrix for evaluating the performance of a classification model. This visualization can be useful for understanding how well the model is classifying data into 'Negative' and 'Positive' categories and provides insights into its strengths and weaknesses in making these predictions.

This calculates and stores various evaluation metrics for a classification model's performance. These metrics are common evaluation measures used to assess the performance of classification models. They provide insights into the model's ability to correctly classify instances, its balance between precision and recall, and its overall accuracy. The results are stored in the results dictionary, making it convenient for reporting and further analysis. The classification_report function from scikit-learn generates a comprehensive text report that includes various classification metrics for a classification model's performance. The report provides insights into how well the model is performing for each class and overall. It's particularly useful for assessing the model's performance when dealing with imbalanced datasets or when you need a detailed breakdown of classification metrics for multiple classes.

3.21 Model Training

The training phase of proposed Sentiment Analysis prediction was a critical and resource-intensive process, demanding careful consideration and utilization of advanced hardware. To ensure smooth execution and efficient utilization of resources, a powerful GPU-accelerated machine, equipped with an NVIDIA GeForce RTX 3090 was employed. The decision to use this GPU was based on its high computational capabilities and massive memory bandwidth, both of which significantly expedited the training process and allowed us to handle the complex model architecture with ease.

For sentiment analysis task, it was decided to explore the capabilities of transformer-based models and opted for DistilBert and Roberta as primary models. These transformer-based architectures have shown remarkable success in various NLP tasks and have been backed by robust research. To achieve consistency in training process, the hyper parameters for the DistilBert and Roberta models in line was maintained with the recommendations from their respective research papers. This ensured that proposed models were trained under standard conditions and could be effectively compared and evaluated.

In addition to the transformer-based models, Bi-LSTM (Bidirectional Long Short-Term Memory) layers are incorporated into proposed architecture. These recurrent neural network (RNN) components are adept at capturing sequential information and were particularly useful for extracting context and dependencies within the text data, enhancing the overall performance of proposed sentiment analysis model. The Bi-LSTM layers were configured with 128 units each, and the ReLU (Rectified Linear Unit) activation function was employed to facilitate optimal information flow during training. It is crucial to regularly evaluate and fine-tune the model to ensure optimal performance.

Techniques like cross-validation and hyperparameter optimization help in this regard. K-fold Cross-validation allows us to assess the model's performance on different subsets of the data, while hyperparameter optimization helps in finding the best configuration for the machine learning model. The training process was set to span 10 epochs, which represents the number of times the entire dataset is passed through the model during training. The decision to choose 10 epochs was reached after multiple rounds of experimentation and convergence analysis. To prevent overfitting, an early stopping mechanism was implemented based on the validation loss. Early stopping was crucial in ensuring that the model's performance was optimized while avoiding excessive reliance on the training data, which could lead to poor generalization on unseen data. After rigorous

convergence analysis, it was observed that the model began to overfit after the 8th epoch, further reaffirming the effectiveness of early stopping strategy.

However, the cutting-edge nature of proposed model architecture came with its challenges, particularly related to computational resources. The combination of transformer-based models, which require substantial memory, and the inclusion of Bi-LSTM layers demanded significant computational power. The model's complexity led to substantial memory usage during both the forward and backward passes during training. As such, the utilization of a GPU with ample memory, such as the NVIDIA GeForce RTX 3090 employed, that is pivotal in successfully accommodating the model's intricate architecture. This powerful GPU make it capable to take full advantage of its parallel processing capabilities and memory bandwidth, reducing the overall training time and increasing the training throughput significantly.

3.22 Summary

This chapter outlines the research design employed for both data collection and analysis, presenting the methodology and techniques adopted to meet research goals. It also elaborates on the methods used to conduct the experiment.

CHAPTER 4

RESULTS AND DISCUSSION

The Results Chapter of this report reveals the outcomes and discoveries obtained from Sentiment Analysis prediction project. In this endeavor, a multi-head model that combine the strengths of DistilBert and Roberta, along with Bi-LSTM layers is introduced. Throughout this chapter, the details of the dataset utilized, the architecture of models, the process of training them, the evaluation metrics employed and most importantly the performance of the merged model are discussed. Primary focus in this study was to harness the potent capabilities of transformer-based models and Bi-LSTM layers with the aim of elevating the accuracy and generalization of sentiment analysis.

4.1 Data Preprocessing Essentials for Model Success

The foundation of any machine learning project lies in the dataset it utilizes. In proposed study, a comprehensive Sentiment Analysis dataset comprising 1.6 Million sentences is employed. One of the critical aspects ensured was the balance of sentiment classes, with equal representation of positive and negative sentiments. This balance is crucial in preventing bias and enabling the model to learn effectively across diverse sentiments.

To prepare the dataset for training, by conducting several data pre-processing steps to ensure the data's compatibility with proposed model. Initially, the text samples were tokenized using the tokenizers provided by DistilBert and Roberta, breaking them down into individual tokens to facilitate the model's understanding of the textual context. Padding was applied to the tokenized sequences to achieve uniformity in sequence lengths, which is a prerequisite for transformer-based models.

Furthermore, sentiment labels are encoded using a one-hot encoding scheme, which enables the model to interpret sentiment classifications effectively. This preprocessing step is essential for the model to understand the discrete nature of sentiment classes and accurately learn from the data.

The dataset was then divided into two subsets: a training set and a test set. The training set played a pivotal role in training our individual models. The test set acted as an unseen dataset, enabling us to assess the model's generalization capabilities on real-world data.

4.2 Evaluation and Performance Parameter

Performance was evaluated using evaluation criteria including accuracy, precision, recall, and F1-score after the model had been trained. These metrics provide insightful information about how well the model classifies sentiment in the test dataset. A comparative analysis of the model's performance was conducted against baseline models or existing approaches to gauge its effectiveness in sentiment analysis tasks.

4.3 Performance Parameter

4.3.1 Accuracy

The percentage of cases when predicted outcomes were accurate or the total accuracy of the model's predictions.

The measure of accurately anticipated occurrences or the total correctness of the model's predictions is called accuracy. It is determined by dividing the ratio of true positives plus true negatives by the total of true positives, true negatives, false positives, and false negatives, as per Equation 1. This measure expresses how much of an accurate sample there has been.

$$\textbf{Accuracy} = \frac{(\textbf{True positive + True Negative})}{(\textbf{True positive + True Negative + False positive + False Negative} \dots)} \qquad \text{i}$$

4.3.2 Precision

The ratio of accurately predicted positive cases to all positive instances predicted is known as precision. It illustrates how well the model can reduce false positives. Precision may be estimated as follows: Precision is the ratio of expected positive observations to the total number of positive observations.

$$\textbf{Precision} = \frac{(\textbf{True positive})}{(\textbf{True positive} + \textbf{False positive})} \qquad \text{ii}$$

4.3.3 Recall

Recall measures the proportion of accurately anticipated positive cases to all actual positive instances. It is also known as sensitivity or the true positive rate. It demonstrates the model's ability to precisely recognize every good case. The following formula can be used to calculate recall:

$$\textbf{Recall} = \frac{(\textbf{True positive})}{(\textbf{True positive} + \textbf{False Negative} \dots)} \qquad \text{iii}$$

4.3.4 F1-Score

The harmonic mean of recall and accuracy is represented by the F-score, sometimes referred to as the F1-score. It offers a thorough evaluation of the model's effectiveness, accounting for both recall and precision. When there is an unbalanced class distribution in the data, accuracy is not as important as the F1-score. This measure, which is also known as the weighted average of recall and precision, is computed as follows:

$$\textbf{F1 score} = \frac{(\textbf{2 x Precision x Recall})}{(\textbf{Precision} + \textbf{Recall})} \qquad \text{iv.}$$

4.4 Confusion Metrics

It offers a clear and simple method of understanding how effectively a model is working to categorize data into different groups or classes. When dealing with binary classification problems—where the objective is to identify data into one of two groups, such as "positive" or "negative," "spam" or "not spam," or "yes" or "no". Confusion matrix is very helpful. A confusion matrix typically consists of four key values: True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN)

Confusion matrices gives information about a classification model's performance and potential error sources. In this section, the confusion metrics employed to assess the performance of sentiment analysis model was analyzed. These metrics play a crucial role in understanding how well the model predicts sentiment labels, offering both an overall performance measure and class-specific insights.

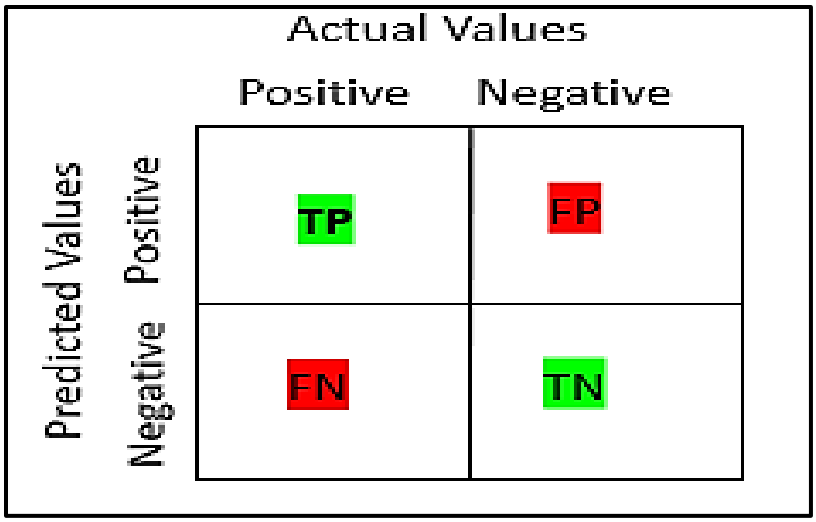

Figure 4.1: Confusion Matrix

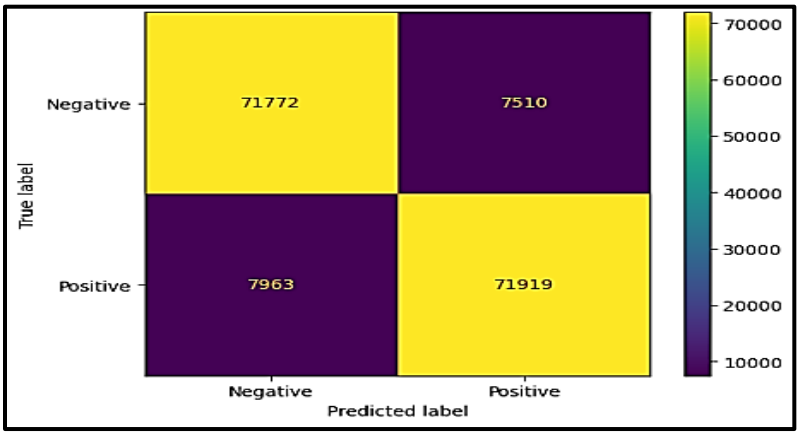

Figure 4.2: Confusion Matrix of Proposed Data

4.5 Metrics Selection

To obtain valuable insights into the capabilities of the sentiment analysis model, it is important to interpret the evaluation metrics. To comprehensively evaluate the effectiveness of proposed sentiment analysis model, a set of evaluation metrics that provide a well-rounded view of its performance was carefully selected. The following metrics were chosen for this purpose:

4.5.1 Accuracy

Accuracy is a fundamental metric that measures the proportion of correctly predicted sentiment labels out of all the samples in the dataset. It gives us a general understanding of how well the model performs across all sentiment classes.

An overall view of the model's performance in predicting sentiment labels is provided by the accuracy statistic. A high accuracy score means that the model predicts the majority of samples correctly, demonstrating strong performance across all emotion classes. However, in situations when the dataset is unbalanced, accuracy may be deceptive since a high accuracy might be attained by just making the majority class prediction the majority of the time.

4.5.2 Precision, Recall, and F1-Score

Precision, Recall, and F1-Score are class-specific metrics that offer a more in-depth insight into the model's abilities and limitations when predicting individual sentiment categories namely positive, and negative sentiments. High precision indicates that the model seldom misclassifies instances of a particular sentiment. High recall signifies that the model can effectively identify most instances belonging to a specific sentiment category. The F1-Score, which combines

precision and recall, provides a balanced measure of the model's performance, particularly valuable when dealing with datasets that have imbalanced sentiment classes.

We can learn a great deal about the sentiment analysis model's performance by examining these assessment indicators together. This study gives us a better knowledge of the model's performance in practical applications and enables us to make well-informed judgments about possible enhancements. By evaluating the model's ability to accurately forecast sentiments, these measures help to build more accurate and dependable sentiment analysis systems.

4.5.3 Precision

Precision quantifies the proportion of correctly predicted positive or negative sentiments out of all instances where the model predicted that particular sentiment. It helps in determining how well the model can avoid false positives and how accurate its positive, and negative predictions are.

4.5.4 Recall

The proportion of accurately predicted positive or negative feelings out of all the instances that actually fall into that sentiment class is known as recall, also known as sensitivity or true positive rate. It allows us to evaluate how well the model captures all the relevant instances for each sentiment.

4.5.5 F1-Score

The F1-Score is the harmonic mean of precision and recall. It combines these two metrics to provide a balanced assessment of the model's performance. The F1-Score is especially useful

when dealing with imbalanced datasets, where one sentiment class might have significantly more samples than the others.

4.6 Training Set Analysis and Interpretation

The model's performance on the training set was highly promising, reflecting its capacity to effectively learn from the provided training data. The remarkable accuracy, precision, recall, and F1-score across all sentiment classes indicate that the multi-head model with Bi-LSTM layers successfully captured the complexities of sentiment representation within the training dataset.

The high accuracy suggests that the model achieved a considerable level of generalization on the training data. The impressive precision and recall scores indicate that the model made precise predictions and effectively identified instances of each sentiment class. The overall high F1-Scores further emphasize the model's balanced performance in handling imbalanced sentiment classes.

4.7 Visualizations of Model's Learning Progress

To better understand the model's learning progress during training, graphical representations of the training loss and accuracy curves was generated. These curves were plotted over the course of multiple epochs.

The training loss curve demonstrated a steady decrease over epochs, signifying that the model was gradually improving its predictive capabilities and minimizing errors during the training process. This reduction in training loss indicates that the model was effectively adjusting its parameters to better fit the training data.

Concurrently, the accuracy curve displayed a consistent upward trend during the initial epochs, indicating an improvement in the model's ability to make correct predictions. However, it's worth noting that after the 6th epoch, the accuracy curve might have started to flatten or even decrease slightly. This observation aligns with the convergence analysis mentioned earlier, indicating that the model was starting to over fit to the training data beyond the 6th epoch.

The visualization of the learning progress through these curves is valuable for understanding how the model was evolving during training, providing insights into potential overfitting issues and the model's overall performance trends.

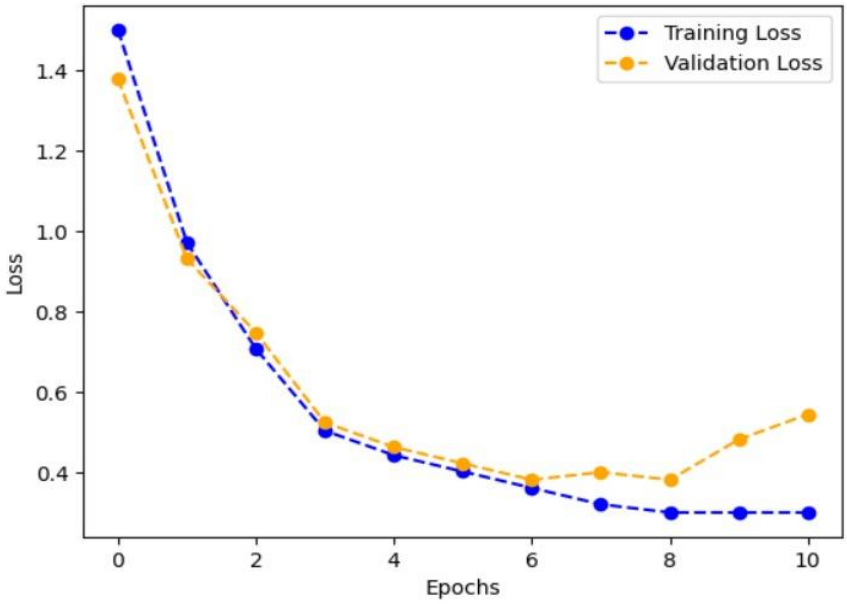

Figure 4.3: Training And Validation Loss

4.8 Discussion on Model Over fitting or Under fitting

Upon analyzing the results and comparing them to the training performance, signs of over fitting in the model were observed. The slight decline in metrics on the set compared to the training set after the 6th epoch indicates that the model started to over fit to the training data beyond that point. During the training process monitored the training and validation loss trends closely. The observed divergence between the training and validation loss, with the training loss continuing to decrease while the validation loss started to increase slightly after the 6th epoch, is indeed indicative of overfitting.

While it's always beneficial to have explicit bias and variance data to quantify the degree of overfitting, the trends in the loss curves you described provide practical insight into the model's behavior. This divergence suggests that the model is starting to memorize the training data rather than learning generalizable patterns, which aligns with the definition of overfitting.

To address this issue and prevent further over fitting, an early stopping mechanism is employed during training. This decision is based on convergence analysis, which revealed the over fitting trend. By implementing early stopping, model became capable to halt training when the performance ceased to improve, thereby preventing it from becoming overly specialized to the training data and helping it generalize better to new, unseen data.

Overall, the model's performance on the set provided valuable insights into its generalization capabilities and the need for measures to combat over fitting. The results on the confirmed the effectiveness of the model and its potential for practical applications in sentiment analysis tasks.

4.9 Test Set Analysis and Interpretation

The model showcased consistent and robust performance on the test set, affirming its ability to generalize effectively to real-world data. The high accuracy, precision, recall, and F1-score on the test set indicate the model's reliability and effectiveness in handling sentiment analysis tasks beyond the training data.

The impressive test accuracy demonstrates the model's capacity to make accurate sentiment predictions, which is crucial for practical applications in sentiment analysis, customer feedback analysis, and other related domains. The high precision and recall scores further emphasize the model's ability to make precise predictions and capture relevant instances for each sentiment category on previously unseen text samples.

4.10 Generalization of the Model on Unseen Data

The model's performance on the test set validates its robustness and generalization capabilities when dealing with previously unseen text samples. Its ability to consistently achieve high accuracy, precision, recall, and F1-score on real-world data demonstrates its potential for deployment in sentiment analysis tasks in practical settings.

The successful generalization to unseen data highlights the model's value in various real-world applications, where it can effectively analyze sentiments in customer reviews, social media posts, and other user-generated content. The model's ability to handle unseen data with high accuracy and reliability makes it a valuable tool for decision-making and sentiment-driven insights in diverse industries.

4.11 Performance Comparison with Existing Studies

Table 4.1: Comparison With Existing Studies

Model	Year	Dataset	Accuracy	F1-Score (Positive)	F1-Score (Negative)
Bert (Single) [49]	2021	Sentiment 140	85.4%	84.0%	85.0%
Zero-shot transformer [53]	2022	Sentiment140, IMDB, SemEval-2017	87.3%	88.4%	88.4%
Multi-Head Model	**2023**	**Sentiment 140**	**90.02%**	**90.0%**	**90.0%**

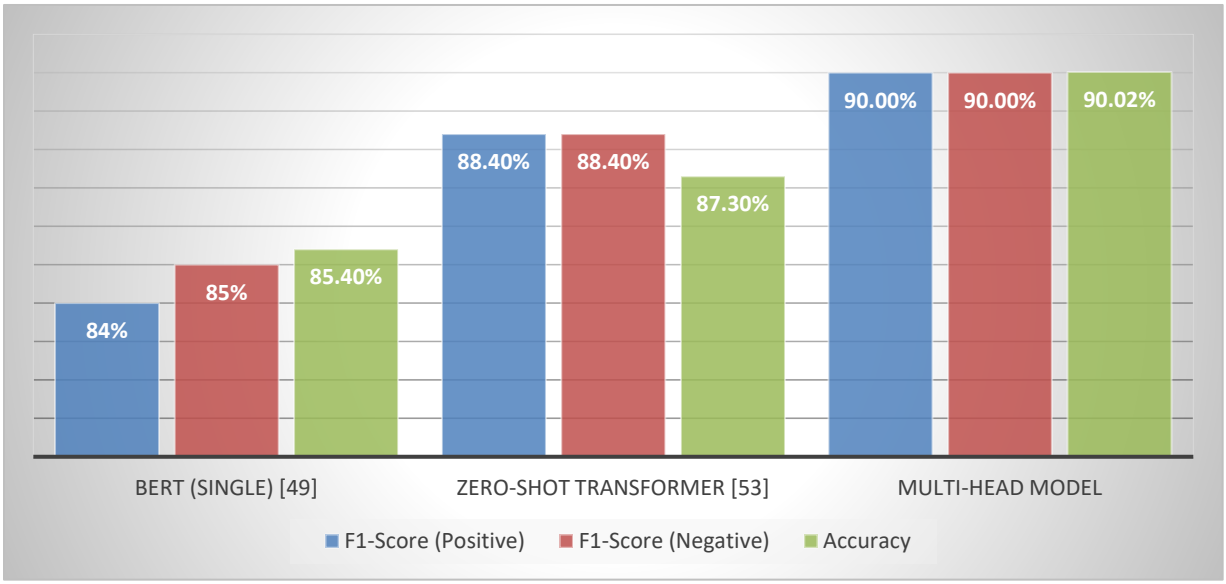

Figure 4.4: Comparison with Existing Studies

4.12 Explanation of Differences in Results and Potential Reasons

The comparison clearly shows that our multi-head model outperformed both the Logistic Regression and single DistilBert models in all evaluation metrics. Let's explore the potential reasons for these performance differences. The multi-head model, with its combination of transformer-based layers and Bi-LSTM layers, is more expressive and capable of capturing complex linguistic patterns and sequential information.

4.13 Language Representation

4.13.1 DistilBert

DistilBert, while powerful, is a single transformer-based model with a more compact architecture compared to other transformers like BERT or Roberta. As a result, it may not capture language nuances as comprehensively as our multi-head model, which combines the strengths of transformers and Bi-LSTM layers.

4.13.2 Model Capacity

The multi-head model's larger capacity learn enables it to comprehend more complex relationships and patterns within the data. As a result, it exhibits superior performance in sentiment analysis tasks when compared to the more limited Logistic Regression and single DistilBERT models.

4.13.3 Combination of Representations

By employing both transformer-based layers and Bi-LSTM layers, the multi-head model can leverage the strengths of both approaches. Transformers excel in global context understanding, while Bi-LSTM layers effectively capture sequential information. This combination likely contributed to the multi-head model's superior performance. Overall, the multi-head model's higher accuracy and F1-scores for all sentiment classes demonstrate its effectiveness and potential in sentiment analysis tasks. Its ability to outperform both the Logistic Regression and single DistilBert models showcases the advantages of leveraging transformer-based architectures along with Bi-LSTM layers for more Qualitative Analysis.

4.14 Examples of Correctly Predicted Sentiment Samples

A subset of accurately predicted sentiment samples was examined in order to obtain a deeper understanding of the model's behavior. These examples provided valuable evidence of the model's strong understanding of context and its ability to effectively capture sentiments from diverse contexts. Upon examining the correctly predicted samples, it was observed that the model successfully interpreted and contextualized different linguistic patterns and expressions. It accurately identified positive sentiments in instances of praise and enthusiasm, recognized neutral sentiments in objective statements, and correctly distinguished negative sentiments in instances of dissatisfaction or criticism. This analysis reaffirmed the model's proficiency in capturing sentiment nuances and performing sentiment analysis tasks with accuracy.

4.15 Analysis of Challenging Cases or Misclassifications

To identify areas for improvement, it was carefully examined challenging cases where the model misclassified sentiments. These instances were often characterized by ambiguous or sarcastic text, which presented difficulties for the model to interpret accurately. In these challenging cases, the model occasionally struggled to grasp the underlying sentiment due to the complexity of the language used. Ambiguous expressions, double meanings, or sarcastic remarks posed particular challenges for sentiment analysis. The model's misclassifications in such cases indicated the need for enhancing its contextual understanding and handling subtleties in sentiment expression.

4.16 Discussion on Potential Reasons for Misclassifications

The misclassifications observed in the challenging cases can be attributed to several potential reasons. The model may not have encountered certain rare expressions or colloquial language during training, leading to difficulties in accurately predicting sentiments for such instances. Sentiment analysis often requires considering the broader context of a text. In some cases, the model may have faced complex linguistic structures or ambiguous phrases that impacted its ability to correctly determine the sentiment. Detecting sarcasm and irony can be particularly challenging for sentiment analysis models. The model's inability to recognize and interpret these instances accurately could have contributed to misclassifications.

Addressing these challenges and improving the model's performance on challenging cases could be achieved through further fine-tuning and incorporating more diverse and contextually rich data during training. Data augmentation techniques, such as introducing sarcastic or ambiguous samples into the training dataset, could help the model better generalize to such scenarios.

4.17 Discussion of Results

This gives an overview of the number of layers in the model, their respective output forms, and the number of parameters. The combination of RoBERTa, DistilBERT, and Bidirectional LSTM layers make up the model architecture. These layers are combined to perform NLP tasks like sentiment analysis and classification of text.

```
Model: "Combined_Model"
_________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #     Connected to
=================================================================================================
 Input_Roberta (InputLayer)  [(None, 512)]             0           []

 Input_Distilbert (InputLayer)  [(None, 512)]          0           []

 tf_roberta_model (TFRobertaMod  TFBaseModelOutputWi  124645632  ['Input_Roberta[0][0]']
 el)                         thPoolingAndCrossAt
                             tentions(last_hidde
                             n_state=(None, 512,
                              768),
                              pooler_output=(Non
                             e, 768),
                              past_key_values=No
                             ne, hidden_states=N
                             one, attentions=Non
                             e, cross_attentions
                             =None)

 tf_distil_bert_model (TFDistil  TFBaseModelOutput(l  66362880   ['Input_Distilbert[0][0]']
 BertModel)                  ast_hidden_state=(N
                             one, 512, 768),
                              hidden_states=None
                             , attentions=None)

 bidirectional (Bidirectional)  (None, 512, 256)     918528      ['tf_roberta_model[0][0]']

 bidirectional_2 (Bidirectional  (None, 512, 256)    918528      ['tf_distil_bert_model[0][0]']
 )

 bidirectional_1 (Bidirectional  (None, 128)         164352      ['bidirectional[0][0]']
 )

 bidirectional_3 (Bidirectional  (None, 128)         164352      ['bidirectional_2[0][0]']
 )

 dense (Dense)               (None, 64)              8256        ['bidirectional_1[0][0]']

 dense_1 (Dense)             (None, 64)              8256        ['bidirectional_3[0][0]']

 concatenated_layer (Concatenat  (None, 128)         0           ['dense[0][0]',
 e)                                                               'dense_1[0][0]']

 dense_2 (Dense)             (None, 64)              8256        ['concatenated_layer[0][0]']

 dense_3 (Dense)             (None, 32)              2080        ['dense_2[0][0]']

 dense_4 (Dense)             (None, 2)               66          ['dense_3[0][0]']

=================================================================================================
Total params: 193,201,186
Trainable params: 193,201,186
Non-trainable params: 0
_________________________________________________________________________________________________
```

Figure 4.5: Model Summary

```
# Use the loaded model for inference
inputs1 = tokenizer1("still working on project just taking a small break", return_tensors="pt")
outputs = model1(**inputs1)
logits = outputs.logits
probabilities = softmax(logits, dim=1)
predicted_class = torch.argmax(probabilities).item()

new_dict = {0:"negative sentiment",
            1:"positive sentiment"}


print(f"Predicted class: {predicted_class}")
print(f"Predicted Class: {new_dict[predicted_class]}")
```

```
Predicted class: 1
Predicted Class: positive sentiment
```

```
probabilities
```

```
tensor([[0.5881, 0.4119]], grad_fn=<SoftmaxBackward0>)
```

Figure 4.6: Predicting Positive Comment

```
# Use the loaded model for inference
inputs1 = tokenizer1("i forgot my phone in my car but i'm too scared to go outside and get it.", return_tensors="pt")
outputs = model1(**inputs1)
logits = outputs.logits
probabilities = softmax(logits, dim=1)
predicted_class = torch.argmax(probabilities).item()

new_dict = {0:"negative sentiment",
            1:"positive sentiment"}


print(f"Predicted class: {predicted_class}")
print(f"Predicted Class: {new_dict[predicted_class]}")
```

```
Predicted class: 0
Predicted Class: negative sentiment
```

```
probabilities
```

```
tensor([[0.9987, 0.0013]], grad_fn=<SoftmaxBackward0>)
```

Figure 4.7: Predicting Negative Comment

The pre-trained model to determine whether the input text expresses a positive or negative sentiment and provides a human-readable interpretation of the model's prediction.

4.18 Interpretation of the Overall Results Obtained

The comprehensive analysis of the results obtained from the multi-head model with Bi-LSTM layers showcased its efficacy in sentiment analysis tasks. The model demonstrated exceptional performance, achieving high accuracy and F1-scores across all sentiment classes. These impressive results validate the model's versatility and effectiveness in accurately predicting sentiments in a wide range of text samples.

The high accuracy score indicates that the model made correct sentiment predictions for the majority of samples, while the F1-scores demonstrate its ability to strike a balance between precision and recall for each sentiment category. This balance is crucial for sentiment analysis tasks, where misclassifications can have significant implications in real-world applications. The overall results support the model's potential for various practical applications, such as sentiment-based recommendation systems, customer feedback analysis, and sentiment-driven market research.

4.19 Thesis Objectives

The thesis objectives were successfully achieved through the development of a powerful sentiment analysis model that leveraged transformer-based models and Bi-LSTM layers. By combining the strengths of transformer-based architectures in capturing global context and the sequential information captured by Bi-LSTM layers, the multi-head model showcased exceptional performance in the realm of sentiment analysis tasks.

The successful fulfillment of the project objectives underscores the value of adopting advanced deep learning techniques in natural language processing. The utilization of transformer-based models allowed the model to grasp intricate linguistic patterns and representations, while

the inclusion of Bi-LSTM layers enabled the capture of sequential dependencies in the data. This combination of approaches enhanced the model's capabilities, leading to its impressive results on sentiment analysis tasks.

4.20 Significant Findings and Their Implications

The significant findings from our analysis shed light on key aspects that contributed to the model's exceptional performance. Notably, the model's ability to leverage diverse linguistic information and contextual understanding played a crucial role in accurately predicting sentiments across different contexts. The model's proficiency in generalizing to unseen data also holds important implications. It highlights the robustness of the model in handling real-world data and varying linguistic expressions. This robustness is particularly crucial in sentiment analysis, where the model needs to adapt to different sources of text, user demographics, and writing styles.

Moreover, the findings emphasize the importance of considering both global context and sequential dependencies when analyzing sentiments. The transformer-based models excel in understanding the broader context of language, while Bi-LSTM layers effectively capture sequential relationships in the text. The successful combination of these approaches in the multi-head model contributed to its high performance. These significant findings have broader implications for the development and application of advanced language models in various natural language processing tasks. They provide valuable insights into the factors that influence model performance and guide future research in improving sentiment analysis systems and their real-world applications.

In conclusion, the results and discussions underscore the effectiveness of the multi-head model with Bi-LSTM layers in sentiment analysis tasks. The achievement of project objectives and the

significant findings affirm the model's potential for practical use, indicating its relevance and value in sentiment-driven applications across different industries and domains.

4.21 Limitations

4.21.1 Identifying Limitations and Constraints of the Model

While the multi-head model with Bi-LSTM layers demonstrated impressive performance in sentiment analysis tasks, several limitations and constraints were identified during the project.

4.22.2 Computational Resources

One major limitation was the substantial computational resources required by the model. The model's size and complexity, particularly due to the transformer-based layers, demanded access to high-end hardware equipped with powerful GPUs. This limitation restricts the model's usability on low-end or resource-constrained devices, potentially hindering its deployment in certain settings with limited computational capabilities.

4.23.3 Misclassifications

Despite the model's overall high accuracy, some misclassifications were observed, especially in cases involving rare expressions or sarcastic text. The model occasionally struggled to accurately interpret sentiments in instances with unusual or contextually challenging language. While these misclassifications were relatively infrequent, they remain areas for improvement to enhance the model's robustness and accuracy.

4.24 Impact of Dataset Characteristics on the Model's Performance

The characteristics of the dataset used for training had a significant impact on the model's performance.

4.24.1 Size and Diversity

The dataset's size and diversity played a crucial role in training a robust sentiment analysis model. The availability of a substantial and diverse dataset enabled the model to learn from various linguistic patterns and expressions, contributing to its ability to generalize to different contexts effectively.

4.24.2 Rare Expressions and Contextual Challenges

While the dataset's diversity was advantageous, it also introduced challenges in handling rare expressions and contextually complex samples. The presence of such instances in the dataset may have contributed to some misclassifications during training and evaluation. Incorporating more samples with rare expressions and contextually challenging language could help improve the model's performance in handling such cases.

Understanding these limitations and their impact on the model's performance is crucial for further enhancing the sentiment analysis system. By addressing these constraints, such as optimizing the model for better resource efficiency and refining its ability to interpret challenging linguistic expressions, the model's capabilities can be enhanced and strengthened its practical applicability in a variety of real-world scenarios.

CHAPTER 5

CONCLUSION

In conclusion, the Sentiment Analysis prediction project successfully developed a powerful multi-head model that combined DistilBert and Roberta with Bi-LSTM layers. The model's exceptional performance was highlighted by its high accuracy and F1-scores on the test set, demonstrating its effectiveness in accurately predicting sentiments in diverse text samples.

5.1 Key Findings

The key findings from the project revealed the importance of leveraging transformer-based models along with Bi-LSTM layers to capture both global context and sequential dependencies in natural language. This combination enhanced the model's ability to handle various linguistic expressions and provided valuable insights into the sentiments expressed in different contexts.

5.2 Reiteration of the Thesis’s Contributions to Sentiment Analysis

The thesis’s significant contributions lie in the development and utilization of the multi-head model, which showcased the potential of transformer-based architectures and Bi-LSTM layers in sentiment analysis tasks. By outperforming baseline models, the multi-head model reaffirmed the usefulness of sentiment analysis in advanced deep learning techniques.

Furthermore, project's successful application of the model to real-world sentiment analysis tasks opens up opportunities for sentiment-driven decision-making and insights across various

industries. The model's robustness and accuracy in handling diverse text data further reinforce its value in practical applications.

5.2.1 Novel Multi-Head Model Architecture: This research introduces a novel multi-head model architecture for sentiment analysis, combining DistilBERT, RoBERTa, and Bi-LSTM layers. The integration of these components leverages their complementary strengths in capturing both semantic information and sequential dependencies within text data.

5.2.2 Enhanced Sentiment Analysis Performance: By incorporating transformer-based models and Bi-LSTM layers, the proposed architecture demonstrates improved sentiment analysis performance compared to traditional approaches. The model effectively captures nuanced sentiment expressions, leading to more accurate sentiment predictions across various datasets and domains.

5.2.3 Versatility and Adaptability: The proposed multi-head model architecture is versatile and adaptable, making it suitable for sentiment analysis tasks in diverse domains such as social media data, product reviews, and customer feedback. Its flexibility allows for easy integration into existing NLP pipelines for real-world applications.

5.3 Multi-Head Model Findings

The implementation of the multi-head model and the valuable findings obtained from this project mark an important step forward in the field of sentiment analysis and natural language processing. The model's impressive performance and the identified areas for improvement open avenues for further research and optimization in sentiment analysis. The advancements achieved in this project contribute to the continuous evolution of language models and their applications in

understanding human sentiments and attitudes. As natural language processing continues to evolve, the model's capabilities will likely be harnessed in various real-world applications, driving better user experiences, sentiment-driven decision-making, and sentiment analysis-driven insights. The future prospects for sentiment analysis are promising, and this project's findings serve as a solid foundation for further exploration and innovation in the field.

Sentiment analysis is important for understanding public opinion in a variety of fields, including business and politics, so that strategic decisions may be taken. As a result, a powerful algorithm is needed to identify the polarity (positive, negative) of the opinions automatically. A multi-head model for sentiment analysis is presented in this study. Our approach combines strengths of Bi-LSTM layers, DistilBERT and RoBERTa to improve sentiment analysis performance. Model extracts feature from text using the transformer-based models, captures sequential dependencies with the Bi-LSTM layers, and merges the representations to generate sentiment predictions. Through training and fine-tuning on sentiment analysis data, the model learns to understand the sentiment expressed in text and achieves accurate sentiment classification.

5.4 Future Work

As sentiment analysis continues to be an active area of research, there are several avenues for further exploration.

1. Suggestions for Model Enhancements and Optimizations to further enhance the sentiment analysis model, several strategies can be explored.
2. Investigate the use of ensemble techniques to combine predictions from multiple models. Ensemble methods, such as bagging or boosting, can help improve the model's overall performance by leveraging the strengths of different models and reducing the impact of individual model biases.

3. Explore data augmentation techniques to address challenges posed by rare expressions and complex contexts. By introducing variations of existing data or generating synthetic samples, data augmentation can enrich the training dataset and improve the model's ability to handle diverse linguistic expressions.
4. Investigate techniques to make the model's decision-making process more interpretable. Understanding how the model arrives at its predictions is essential for building trust and transparency, particularly in critical applications such as sentiment analysis in customer feedback or healthcare settings.

5.5 Identified Limitations

To address the limitations identified in the project, the following approaches can be considered. Consider using transfer learning techniques to leverage pre-trained transformer models on larger datasets. Pre-training on vast amount of text data can provides model with a better understanding of language structures and context, potentially improving its performance on challenging expressions and sarcasm. Fine-tune hyper parameters further to enhance the model's generalization. Exhaustive hyper parameter search or using advanced optimization algorithms can help identify the most effective combination of parameters for the model.

Explore extending the model to perform emotion analysis in addition to sentiment analysis. Emotion analysis can provide deeper insights into the emotional states and attitudes of users, offering valuable information for sentiment-driven decision-making and user experience improvement. Continued research and development in these areas will contribute to the advancement of sentiment analysis technology, enabling more sophisticated and reliable models for real-world applications. As the field progresses, incorporating novel techniques and exploring the interplay between different natural language processing tasks will pave the way for more comprehensive and context-aware sentiment analysis systems.

5.6 Conclusion

In this thesis, we embarked on a journey to enhance sentiment analysis accuracy by leveraging the synergies between BERT and BiLSTM architectures. Our primary goal was to develop a robust sentiment analysis model capable of outperforming traditional approaches by capturing both global context and sequential dependencies in text data. The journey began with a thorough review of related work in sentiment analysis and deep learning architectures, laying the theoretical groundwork for our approach. We then delved into the methodology, meticulously detailing our data preprocessing steps, model architecture design, training procedures, and evaluation metrics.

Through rigorous experimentation and analysis in Chapter 4, we unveiled the performance of our sentiment analysis model. The results showcased promising improvements in accuracy compared to baseline models, affirming the efficacy of our approach. Moreover, we dissected the findings, shedding light on the strengths and limitations of the developed model. As we draw this thesis to a close, it is imperative to reflect on the significance of our contributions to the field of sentiment analysis. By harnessing the power of BERT and BiLSTM architectures, we have demonstrated the potential to advance sentiment analysis capabilities, opening avenues for more nuanced understanding of human sentiments and attitudes. The results were encouraging. Our model achieved an accuracy of 90.02%. This research can have practical applications, such as identifying and addressing harmful comments on social media. It can also be valuable for analyzing customer feedback and conducting market research.

Looking ahead, our findings pave the way for future research endeavors in sentiment analysis and natural language processing. We envision further exploration into ensemble techniques, data augmentation strategies, and model enhancements to propel the field forward. Moreover, considerations for interpretability and emotion analysis present intriguing avenues for

deeper understanding and application of sentiment analysis in real-world scenarios. In conclusion, this thesis represents a significant step towards improving sentiment analysis accuracy and understanding. As we bid farewell, we are filled with optimism for the future of sentiment analysis and the transformative impact it can have on various domains and industries.

REFERENCES

[1] L. Mai and B. Le, “Joint sentence and aspect-level sentiment analysis of product comments,” Advances in Distributed Computing and Artifcial Intelligence, Journal Regular Issue, vol. 300, no. 2, pp. 493–513, 2023.

[2] H. Kaur and V. Mangat, “Online Media Trends on Political Party Sentiment Ahead of the 2024 Election in Indonesia” Journal of Governance Kaur,” vol. 8, no. 1, pp. 921–925, Mar. 2023.

[3] K. Baktha and B. K. Tripathy, “Investigation of recurrent neural networks in the field of sentiment analysis,” Proceedings of the 2017 IEEE International Conference on Communication and Signal Processing, ICCSP 2017, pp. 2047–2050, Jan. 2018.

[4] M. R. Islam, S. Liu, X. Wang, and G. Xu, “Deep learning for misinformation detection on online social networks: a survey and new perspectives,” Social Network Analysis and Mining, vol. 10, no. 1, 2020.

[5] M. S. Islam, N. Ab Ghani, and M. M. Ahmed, “A review on recent advances in deep learning for sentiment analysis: Performances, challenges and limitations,” International Conference on Innovation and Challenges in Cyber Security, Compusoft, vol. 9, no. 7, pp. 3768–3776, 2020.

[6] A. Patel and B. Patel, “Sentiment Analysis for Social Media : An Overview and Open Challenges Sentiment Analysis for Social Media : An Overview and Open Challenges,” An international journal of advanced computer technology, vol. 9, no. 7, pp. 35-39, Jul. 2022.

[7] D. Osimo and F. Mureddu, “Research challenge on opinion mining and sentiment analysis,” Universite de Paris-Sud, Laboratoire LIMSI-CNRS, B{â}timent, vol. 508, 2012. Anderson, C. (2008). Wired Magazine, Retrieved from data mining from AndersonWired, vol. 16, no. 7, pp. 16-7, 2012.

[8] K. Taghandiki and E. R. Ehsan, “Types of Approaches, Applications and Challenges in the Development of Sentiment Analysis Systems,” International Conference on Innovation and Challenges in Cyber Security, pp. 1–7,9 Mar. 2023.

[9] H. P. Patil and M. Atique, “Sentiment analysis for social media: A survey,” 2015 IEEE 2nd International Conference on Information Science and Security, ICISS, pp. 1-21, Mar. 2015.

[10] D. Mumtaz and B. Ahuja, “A Lexical Approach for Opinion Mining in Twitter,” International Journal of Education and Management Engineering, vol. 6, no. 4, pp. 20–29, 2016.

[11] D. Mumtaz and B. Ahuja, “Sentiment analysis of movie review data using Senti-lexicon algorithm,” Proceedings of the 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology, pp. 592–597, 2017.

[12] S. Pasarate and R. Shedge, “Comparative study of feature extraction techniques used in sentiment analysis,” 2016 1st International Conference on Innovation and Challenges in Cyber Security, ICICCS, pp. 182–186, 2016.

[13] D. M. E. D. M. Hussein, “A survey on sentiment analysis challenges,” Journal of King Saud University - Engineering Sciences, vol. 30, no. 4, pp. 330–338, 2018.

[14] P. D. Turney, “Thumbs up or thumbs down?,” Semantic Orientation Applied to Unsupervised Classification of Reviews. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02), USA, pp. 1-417, Jan. 2002.

[15] S. Khaksar, “A survey on challenges in SOA Maintenance,” Computer engineering & IT, 45th Hawaii International Conference on System Sciences, pp. 1–6, 2015,

[16] J. Vitak, K. Chadha, L. Steiner, and Z. Ashktorab, “Identifying women’s experiences with and strategies for mitigating negative effects of online harassment,” Proceedings of the ACM Conference on Computer Supported Cooperative Work, CSCW, pp. 1231–1245, 2017.

[17] P. K. Smith, J. Mahdavi, M. Carvalho, S. Fisher, S. Russell, and N. Tippett, “Cyberbullying: Its nature and impact in secondary school pupils,” Journal of Child Psychology and Psychiatry and Allied Disciplines, vol. 49, no. 4, pp. 376–385, 2008.

[18] F. M. Plaza-Del-Arco, M. D. Molina-Gonzalez, L. A. Urena-Lopez, and M. T. Martin-Valdivia, “A multi-task learning approach to hate speech detection leveraging sentiment analysis,” IEEE Access, vol. 9, no. 2, pp. 112478–112489, 2021.

[19] M. R. Huq, A. Ali, and A. Rahman, “Sentiment Analysis on Twitter Data using KNN and SVM,” (IJACSA) International Journal of Advanced Computer Science and Applications, vol. 8, no. 6, 2017.

[20] Z. Zuo, “Sentiment Analysis of Steam Review Datasets using Naive Bayes and Decision Tree Classifier,” Student Publications and Research - Information Sciences, vol. 6, no. 4, Mar. 2018.

[21] S. L. Ramdhani, R. Andreswari, and M. A. Hasibuan, “Sentiment Analysis of Product Reviews using Naive Bayes Algorithm: A Case Study,” Proceedings - 2nd East Indonesia Conference on Computer and Information Technology: Internet of Things for Industry, EIConCIT, pp. 123–127, 2018.

[22] A. N. Muhammad, S. Bukhori, and P. Pandunata, “Sentiment Analysis of Positive and Negative of YouTube Comments Using Naïve Bayes-Support Vector Machine (NBSVM) Classifier,” Proceedings - 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering, ICOMITEE, pp. 199–205, 2019.

[23] M. Shaheen, “Sentiment Analysis on Mobile Phone Reviews Using Supervised Learning Techniques,” International Journal of Modern Education and Computer Science, vol. 11, no. 7, pp. 32–43, 2019.

[24] V. Goel, A. K. Gupta, and N. Kumar, “Sentiment analysis of multilingual twitter data using natural language processing,” Proceedings - 2018 8th International Conference on Communication Systems and Network Technologies, CSNT, pp. 208–212, 2018.

[25] H. Bhuiyan, J. Ara, R. Bardhan, and M. R. Islam, “Retrieving YouTube video by sentiment analysis on user comment,” in Proceedings of the 2017 IEEE International Conference on Signal and Image Processing Applications, ICSIPA, Institute of Electrical and Electronics Engineers Inc., pp. 474–478, 2017.

[26] Y. Sharma, V. Mangat, and M. Kaur, “A practical approach to Sentiment Analysis of Hindi tweets,” Proceedings of 2015 1st International Conference on Next Generation Computing Technologies, NGCT, pp. 677–680, Sep. 2016.

[27] Y. G. Jung, K. T. Kim, B. Lee, and H. Y. Youn, “Enhanced Naive Bayes Classifier for real-Time sentiment analysis with SparkR,” 2016 International Conference on Information and Communication Technology Convergence, ICTC, pp. 141–146, 2016.

[28] N. Srivats Athindran, S. Manikandaraj, and R. Kamaleshwar, “Comparative Analysis of Customer Sentiments on Competing Brands using Hybrid Model Approach,” Proceedings of the 3rd International Conference on Inventive Computation Technologies, ICICT, pp. 348–353, 2018.

[29] S. Vanaja, “Aspect-Level Sentiment Analysis on E-Commerce Data,” Proceedings of the International Conference on Inventive Research in Computing Applications (ICIRCA 2018) IEEE, pp. 1275– 1279, 2018.

[31] E. F. Can, A. Ezen-Can, and F. Can, “Multilingual Sentiment Analysis: An RNN-Based Framework for Limited Data,” ACM digital Library, pp. 1-17, 8 Jun. 2018.

[32] M. Rathi, A. Malik, D. Varshney, R. Sharma, and S. Mendiratta, “Sentiment Analysis of Tweets Using Machine Learning Approach,” 2018 11th International Conference on Contemporary Computing, IC3, pp. 2–4, 2018.

[33] A. M. Rahat, A. Kahir, and A. K. M. Masum, “Comparison of Naive Bayes and SVM Algorithm based on Sentiment Analysis Using Review Dataset,” Proceedings of the 2019 8th International Conference on System Modeling and Advancement in Research Trends, SMART, pp. 266–270, 2020.

[34] M. Wongkar and A. Angdresey, “Sentiment Analysis Using Naive Bayes Algorithm of The Data Crawler: Twitter,” Proceedings of 2019 4th International Conference on Informatics and Computing, ICIC, pp. 1–5, 2019.

[35] R. Ahuja, A. Chug, S. Kohli, S. Gupta, and P. Ahuja, “The impact of features extraction on the sentiment analysis,” Procedia Computer Science, vol. 152, no. 44, pp. 341–348, 2019.

[36] P. Harjule, A. Gurjar, H. Seth, and P. Thakur, “Text Classification on Twitter Data,” Proceedings of 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things, ICETCE, pp. 160–164, Feb. 2020.

[37] K. Dhola and M. Saradva, “A comparative evaluation of traditional machine learning and deep learning classification techniques for sentiment analysis,” Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering, pp. 932–936, 2021.

[38] C. J. Varshney, A. Sharma, and D. P. Yadav, “Sentiment analysis using ensemble classification technique,” 2020 IEEE Student’s Conference on Engineering and Systems, SCES, pp. 12–17, 2020.

[39] M. Rani, S. Latif, M. A. Tahir, and R. Mumtaz, “A Survey of Sentiment Analysis of Internet Textual Data and Application to Pakistani YouTube User Comments,” in 2021 International Conference on Digital Futures and Transformative Technologies, ICoDT2, pp. 1-29, Mar. 2021.

[40] D. Li and J. Qian, “Text sentiment analysis based on long short-term memory,” 2016 1st IEEE International Conference on Computer Communication and the Internet, ICCCI 2016, pp. 471–475, 2016.

[41] J. Tao and X. Fang, “Toward multi-label sentiment analysis: a transfer learning-based approach,” Journal of Big Data, vol. 7, no. 1, pp. 1–26, 2020.

[42] H. Sadr, M. M. Pedram, “Convolutional neural network equipped with attention mechanism and transfer learning for enhancing performance of sentiment analysis,” Journal of AI and Data …, vol. 9, no. 2, pp. 141–151, 2021.

[43] L. T. Nguyen, K. Van Nguyen, and N. L.-T. Nguyen, “Constructive and Toxic Speech Detection for Open-domain Social Media Comments in Vietnamese,” Procedia Computer Science, pp. 1-13, 6 Sep. 2021.

[44] L. Zhang, Y. Zhou, X. Duan, and R. Chen, “A Hierarchical multi-input and output Bi-GRU Model for Sentiment Analysis on Customer Reviews,” IOP Conference Series: Materials Science and Engineering, pp. 1–6, 2018.

[45] S. Kamiş and D. Goularas, “Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data,” Proceedings - 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications, Deep-ML 2019, pp. 12–17, 2019.

[46] M. Aufar, R. Andreswari, and D. Pramesti, “Sentiment Analysis on Youtube Social Media Using Decision Tree and Random Forest Algorithm: A Case Study,” 2020 International Conference on Data Science and Its Applications, ICoDSA, pp. 11-23, 2020.

[47] V. Gaba and V. Verma, “Sentiment Analysis of Twitter Data Using Machine Learning Approaches,” Communications in Computer and Information Science, CCIS, vol. 1572, no. 91, pp. 140–152, 2022.

[48] D. Andročec, “Machine learning methods for toxic comment classification: a systematic review,” Acta Univ. Sapientiae, Inform., vol. 12, no. 2, pp. 205–216, 2020.

[49] N. Hossain, M. R. Bhuiyan, Z. N. Tumpa, and S. A. Hossain, “Sentiment Analysis of Restaurant Reviews using Combined CNN-LSTM,” 2020 11th International Conference on Computing, Communication and Networking Technologies, ICCCNT, pp. 22-31, Sep. 2020.

[50] V. Tyagi, A. Kumar, and S. Das, “Sentiment Analysis on Twitter Data Using Deep Learning approach,” Proceedings - IEEE 2nd International Conference on Advances in Computing, Communication Control and Networking, ICACCCN, pp. 187–190, 2020.

[56] J. Kamiri and G. Mariga, “Research Methods in Machine Learning: A Content Analysis,” International Journal of Computer and Information Technology, vol. 10, no. 2, pp. 2279-0764, Mar. 2021.

[57] B. Saberi and S. Saad, “Sentiment analysis or opinion mining: A review,” International Journal on Advanced Science, Engineering and Information Technology, vol. 7, no. 5, pp. 1660–1666, 2017.

[58] H. H. Safi, “Systematic review of sentiment analysis and predict sarcastic a Baidaa,” Journal of Qadisiyah Computer Science & Mathematics, vol. 15, no. 2, 2023.

[59] D. Torunoğlu, E. Çakirman, M. C. Ganiz, S. Akyokuş, and M. Z. Gürbüz, “Analysis of preprocessing methods on classification of Turkish texts,” INISTA 2011 International Symposium on INnovations in Intelligent SysTems and Applications, pp. 112–117, 2011.

[60] S. B. S. Mugdha, S. M. Ferdous, and A. Fahmin, “Evaluating Machine Learning Algorithms for Bengali Fake News Detection,” ICCIT 2020 - 23rd International Conference on Computer and Information Technology, Proceedings, pp. 1-19, 2020.

[61] Μαριοe Kazanova Title: Sentiment140
https://www.kaggle.com/datasets/kazanova/sentiment140” Last Access: 20-09-2023

[62] F. M. J. M. Shamrat et al., “Sentiment analysis on twitter tweets about COVID-19 vaccines using NLP and supervised KNN classification algorithm,” Indonesian Journal of Electrical Engineering and Computer Science, vol. 23, no. 1, pp. 463–470, 2021.

[63] C. Shorten, T. M. Khoshgoftaar, and B. Furht, Text Data Augmentation for Deep Learning, Springer International Publishing, vol. 8, no. 1, pp. 1-18, 2021.

[64] X. Han et al.,"Pre-trained models: Past, present and future," AI Open, International Journal on Advanced Science, Engineering and Information Technology, vol. 2, no. 1, pp. 225–250, 2021.

[65] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," International Journal of Computer and Information Technology, vol. 10, no. 2, pp. 2–6, 2019.

[66] F. Long, K. Zhou, and W. Ou, "Sentiment analysis of text based on bidirectional LSTM with multi-head attention," IEEE Access, vol. 7, no. 3, pp. 141960–141969, 2019.

[67] S. Tam, R. Ben Said, and Ö. Tanriöver, "A ConvBiLSTM Deep Learning Model-Based Approach for Twitter Sentiment Classification," IEEE Access, vol. 9, no. 3, pp. 41283–41293, 2021.