

عبدالحافظ العصار

Sentiment Analysis for Mobile App Reviews A Comparative Study.docx

1- Final Thesis (Word Format)

Thesis_(2)_AI901_Dr_Wessam_Ahmed_Third Trimester 2025 (Moodle PP)

Midocean University

Document Details

Submission ID

trn:oid::1:3450647764

Submission Date

Dec 25, 2025, 2:45 PM GMT+4

Download Date

Dec 25, 2025, 10:40 PM GMT+4

File Name

xcod....621_ydutS_evitarapmoC_A_sweiveR_ppA_eliboM_rof_sisylanA_tnemitneS_عبدالحافظ_العصار_28492

File Size

5.2 MB

89 Pages

22,077 Words

125,396 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.





Sentiment Analysis for Mobile App Reviews: A Comparative Study

Prepared By

Name: Sultan Khalaf Alotaibi

ID No: 222000226

Name: Abdel Hafez Adeeb Alassar

ID No: 251002398

Submitted in fulfillment of the requirements for the degree Master of
Science in Artificial Intelligence

At

College of Informatics

Midocean University

UAE

Supervisor Name: Dr. Wesam Ahmed

College of Informatics, Midocean University, Comoros

Email: wesamahmed@midocean.edu.km

TABLE OF CONTENTS

Introduction.....	5
1.1 Introduction and Background	6
1.2 Problem Statement.....	6
1.3 Research Objectives	7
1.4 Research Contributions	7
1.5 Thesis Organization	8
Overview: Background	9
2.1 Introduction to sentiment Analysis.....	10
2.2 Levels of Sentiment	10
2.2.1 Document-Level Sentiment Analysis	11
2.2.2 Sentence-Level Sentiment Analysis.....	11
2.2.3 Phrase-Level Sentiment Analysis	11
2.2.4 Aspect-Level Sentiment Analysis	12
2.3 Feature Extraction	12
2.4 Term Frequency (TF) and TF-IDF.....	13
2.5 Bag of Words (BoW)	13
2.6 Word Embeddings	14
2.6.1 Word2Vec.....	14
2.6.2 GloVe (Global Vectors for Word Representation).....	15
2.6.3 FastText	15
2.6.4 ELMo (Embeddings from Language Models)	15
2.7 Sentiment Analysis Approaches.....	16
2.7.1 Machine Learning Approach in Sentiment Analysis.....	17
2.7.1.1 Supervised Learning.....	17
2.7.2 Machine learning Models.....	18
2.7.2.1 Naive Bayes.....	18
2.7.2.2 Decision Tree	19
2.7.2.3 Support Vector Machines (SVM).....	20
2.7.2.4 Random Forest.....	21
2.7.3 Deep Learning Approach in Sentiment Analysis.....	23
2.7.4 Deep Learning Models Approaches to Sentiment Analysis.....	23
2.7.4.1 Convolutional Neural Networks (CNNs)	23
2.7.4.2 Recurrent Neural Networks (RNNs).....	24
2.7.4.3 Long Short-Term Memory (LSTM).....	25
2.7.4.4 Gated Recurrent Unit (GRU)	25

2.7.5 Large Language Models (LLMs Approaches to Sentiment Analysis	26
2.7.5.1 Introduction to Large Language Models (LLMs).....	26
2.7.5.2 BERT (Bidirectional Encoder Representations from Transformers).....	27
2.7.5.3 GPT (Generative Pre-trained Transformer).....	28
2.7.5.4 RoBERTa (Robustly Optimized BERT Approach	28
2.7.5.5 DistilBERT.....	28
2.7.5.6 XLNet.....	29
2.8 Evaluation Matrices	31
2.9 Summary	31
Literature Review	33
3.1 Machine Learning Models.....	34
3.2 Deep Learning Models	39
3.3 Large Language Models	43
Proposed Approach.....	52
4.1 Introduction	53
4.2 Dataset Information	54
4.3 Preprocessing.....	54
4.4 Machine Learning Models.....	56
4.4.1 Support Vector Machine (LinearSVC).....	56
4.4.2 Logistic Regression	57
4.4.3 Random Forest Classifier.....	57
4.4.4 LightGBM Classifier	58
4.4.5 Naïve Bayes Classifier.....	59
4.5 Large Language Model.....	59
4.5.1 DistilBERT-Base-Uncased	59
4.5.2 DeBERTa-v3-Base	61
4.5.3 RoBERTa-Base	62
4.5.4 BERT-Base-Uncased.....	63
4.5.5 ELECTRA-Large-Discriminator	64
4.6 Deep Learning Models	65
4.6.1 Bi-LSTM.....	65
4.6.2 CNN + BiGRU Hybrid Model.....	66
4.7 Summary.....	68
Experimental Results and Discussion.....	69
5.1. Introduction	70
5.2. Results of Individual Models.....	70

5.2.1. Traditional Machine Learning Models.....	70
5.2.2. Deep Learning Models.....	72
5.2.3. Results of Transformer-Based Models.....	73
5.3. Discussion of Findings.....	78
5.4. Limitations of the Proposed Work	79
5.5. Summary of the Chapter	79
Conclusion and Future Work.....	80
6.1. Conclusion.....	81
6.2. Future Work	81
References.....	83

Abstract

Since mobile apps marketplaces receive large volumes of noisy and contextualized information, user-generated text sentiment analysis is a proper task in interpreting consumer responses. This has been particularly so in the markets of mobile applications. This research study aims to provide a detailed analysis of the numerous machine learning, deep learning, and transformer-based algorithms employed in the common classification of the sentiment of free-text reviews. We performed experiments on conventional algorithms, including Logistic Regression, Support Vector Machines, Naive Bayes, Random Forest and LightGBM, based on TF-IDF feature representations. Also, we had a tune of these models. We used deep learning models, namely CNN-BiGRU and Bi-LSTM in order to detect local semantic trends and sequential dependences. DistilBERT, RoBERTa, ELECTRA and DeBERTa-v3 were all fine-tuned models built using a transformer to enable the use of contextualized language representation through self-attention. Classical machine learning models are effective with high-dimensional sparse text input as is seen in the outcome of the experiments; but it is difficult to teach these models deep semantics. Despite the fact that deep models have far deeper understanding of the context, they hardly explain the long-range relationships. Transformer-based approaches have a much better performance compared to deep learning and usual learning methods. DeBERTa-v3 can come up with the best possible accuracy of 91.64%. The results of the study suggest that pretrained language models can be used to carry out subtle sentiment tasks, the use of contextualised embeddings is a key ingredient into understanding subtle user sentiments. This paper illuminates the merits and demerits of different modeling paradigms, which, in turn, helps to create a great background of the appearance of sentiment analysis systems, which could be implemented in the real world.

Chapter 1

Introduction

1.1 Introduction and Background

Reviews posted by users have emerged as a precious source of insights to both the stakeholders and the developers [1]. That is because the mobile applications are growing in popularity in the digital market. Mobile application reviews constitute the most crucial form of user-generated data sources since it is tasked with creating new download, influencing the user opinion, and informing product changes to businesses. The efficient use of these studies is therefore becoming more important, and systematic review is necessary [2].

One of such subfields is natural language processing (NLP) which allows recognizing and classifying the opinion or the feelings that are conveyed within the textual data [3]. NLP has a sub-field known as sentiment analysis in written text. The sentiment analysis contributes to derive relevant information among the patient reviews of a very large number of user reviews through categorizing the user reviews into positive, negative, and neutral sentiment [4]. This way, engineers can understand more about the overall happiness of the users, resolve problems that have existed over the course of the years and get to know how people actually feel about a product.

NLP methods provide real solutions in managing, processing and interpreting amount of unstructured text data. These solutions have been created in various applications. Under this approach, the natural language processing (NLP) models can convert the raw text based input to forms that could be subjected to machine learning or deep learning operations. This is achieved by using methods like the tokenization, lemmatization, and the use of vectors. These technologies help to carry out a systematic and scaled analysis of feedback that extends far much more than human beings could read it in the past [5].

This project aims to use the sentiment analysis to evaluate the ratings of users on the mobile game Mobile Legends Bang Bang. This will be achieved by analyzing various methods of categorizing sentiment, traditional machine learning, deep learning networks, and models based on transformers. We aim at finding out the most efficient way of capturing the feelings of users in terms of reviewing mobile applications. Due to the results of this research, the results could be considered helpful to ad developers, marketers, user experience designers who are striving to achieve the objective of increasing the capability of apps to leverage on user feedback.

1.2 Problem Statement

Mobile device applications have become a part of the general lifestyle of most people and have been widely utilized in so many applications and most of them fall under the areas of communication, entertainment, studies, and banking. Although they are so popular, the usage of application continues to be marred by problems in its performance, crashes, and poor designs that only a small number of them have been addressed by updates. Combined collectively, there is a possibility that these poor user experiences would result in poor reviews, poor retention rates and eventual market failure. The developers and other parties that supervise apps must understand the complaints raised by users in time and efficiently to ensure the quality of the app and the satisfaction of the users.

The stored user reviews in the repositories in app stores consist of unstructured information which embodies the actual opinions and sentiments of users. Manual review processing of thousands of reviews is, however, not practical. Sentiment analysis is one of the methods which can be effectively scaled. It does so in that it automatically classifies the comments and feedbacks left by users to be either good, negative or neutral. This helps engineers to know the areas of most development of the system.

Although a number of methods apply in sentiment analysis, the process is still challenging to achieve. Among the studies in the context of mobile app hallings, there are a limited number of research-based studies that had done an extensive study of classical machine learning, deep learning, and transformers to analyze their performances.

To address this gap, this article proposes to adopt and test different sentiment analyzer methods with the application in the context of user ratings of Mobile Legends. This work is aiming at offering aid to developers in the process of utilizing the feedback offered by the users with the aim of making product improvements by the identification of the most helpful models and preprocessing techniques.

1.3 Research Objectives

To achieve the key goal of the given research, a complete sentiment analysis of user reviews of the Mobile Legends app will be conducted, and the results of the functioning of various categorization models will be compared.

1.4 Research Contributions

This study has the following contributions based on the research objectives:

1. **Data Preprocessing Framework:** Developed a robust and efficient preprocessing pipeline to clean up and prepare the app review data, with challenges being on noise, slang, emojis, inconsistent formatting on user-generated data.
2. **Model implementation and evaluation:** Multiple sentiment analysis methods, such as those based on conventional machine learning methods e.g., SVM, Random Forest, deep learning e.g., LSTM, CNN and transformer e.g., BERT methods, have been implemented and evaluated rigorously with regard to mobile app review classification.
3. **Comparison with current models:** performed a thorough analysis of the performance of alternative sentiment analysis models and their strengths, weaknesses, and metrics as defined by accuracy, F1-score, and computational efficiency.
4. **Recommendation of Best Approach:** Established most efficient and feasible model to making review analysis of mobile apps, actionable suggestions to mobile app developers and researchers to gain insight into user sentiment to enhance the experience of their mobile apps.
5. **Practical Insights:** Equipped an organized approach, along with the results that could be used in further research and implementation of sentiment analysis in the mobile application world, specifically in games such as the Mobile Legend.

1.5 Thesis Organization

Chapter 2: provides an overview of sentiment analysis methods, including machine learning, deep learning, and transformer-based approaches, as well as feature extraction techniques.

Chapter 3: reviews the existing literature on sentiment analysis, identifies its strengths and weaknesses, and highlights the research gaps addressed in this study.

Chapter 4: outlines the research framework, including data preparation, model development, and evaluation strategies for sentiment classification.

Chapter 5: presents the performance results of all models using evaluation metrics and discusses their practical implications.

Chapter 6: Summarizes the main findings, tackles the research goals, and suggests possible paths for future studies in sentiment analysis.

Chapter 2

Overview: Background

2.1 Introduction to sentiment Analysis

Sentiment analysis (SA) in natural language processing (NLP) involves the computer providing the capability to analyze and understand human or human expressed feelings or opinions about the subject in written form. The main aim of such a method is to show whether a text, a comment, review, feedback, and post have positive, negative, or neutral mood. It works in the following way: At the very beginning, it has the words and phrases existing on print and digital media measured, and then the measurements of words are taken and submitted to a computer model so that to specify the emotional state of the author who wrote those words. One of the key reasons behind the relevance of sentiment analysis is that it gives businesses, researchers and developers an opportunity to understand the bigger picture in which specific input fits without having to scroll through the syllables. An example of how it can be applied in business is that a business can know how much satisfaction his consumers are with the item of service provided by the business by evaluating the feedback left by the customers. In the case of the mobile application development, the same technique can be applied to see what the people like and dislike about an application by analyzing the app-store reviews, which are left by them. This is incredible in improving the software, corrections to bugs which occur regularly, and a more fulfilling experience. It is also applied in other areas, including politics where it can assess the opinion of people in regards to a policy, or in financial areas where it can track market sentiment by scanning the news or social media. These two are just some of its applications. With the consideration that more and more individuals are writing on the Internet, the sentiment analysis is becoming a resource more and more useful in achieving the goal of turning in that text into meaningful and useful information within as short a period as possible.

2.2 Levels of Sentiment

The level of variability of the sentiment analysis is also flexible and may vary depending on the text nature and the level of sophistication within it. The major levels are document level, sentence level, phrase level and aspect level and each of the levels provides different level of intelligence. Figure 2.1 shows how these levels aid in the process of generalising the sentiment of more specific to general components of the text. All these levels have been indicated in the figure with respective functions.

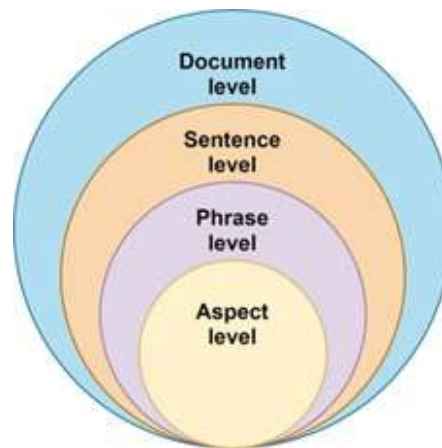


Figure 0.1 Levels of Sentiment Analysis

2.2.1 Document-Level Sentiment Analysis

Document level sentiment analysis is considered to be the most basic form of sentiment analysis. This kind of analysis asserts that document review, article or post-feedback has one thing. The task is to determine the feel of a whole piece of work as falling under three different categories positive, negative, and neutral. It works best when there is only one side or emotion expressed in a flowing way and the whole way through, which is common on products reviews, comments on apps and other similar cases where the author is concerned with one thing. Consider the case of a customer to whom the software has been shown and s/he says, This app is incredible! As the whole paper indicates, the statement is a positive one because it simply works and I love it. The article, however, does not promote well as there is an averaging of everything into one title [6]. This is the case since it articulates both positive and negative ideas regarding different features of the application.

2.2.2 Sentence-Level Sentiment Analysis

Under the system of this very method the text has been divided into sentences and the sentiment analysis has been done without considering each sentence separately. Its application is especially useful in the instances when a review or paper comprises a number of sentences that represent multiple points of view. As an example, in one of the reviews, it is stated, I love the design of the app. Despite the fact that it crashes intensively often, the former is good, and the latter is bad. We can carry out even such analysis of sentiment variation on sentence basis, even on the topic of one document. It provides more visualisation information as compared to document-level analysis; however, can be difficult to convey a long or complex sentence and numerous sentiments within one sentence [7].

2.2.3 Phrase-Level Sentiment Analysis

The analysis of the sentiment on a phrase level is made with the consideration of the smaller pieces of writing, e.g. phrases and expressions that represent high degree of emotional strength. These words can be adjectives, verbs or brief noun phrases, which portray an opinion, possible. An example of such words will be: very useful, too slow or extremely satisfied. The tenor of phrases can be detected through this kind of analysis as can occur with a text that is impartial, but contains neutral remarks. Such an analysis is finer-grained. One such sentence is the following statement: I expected more with the update but the new features are quite useful it is a sentence with a negative tone in the sense of the fact that the sentence includes the words that are negative in terms of tone expected more. A positive one, in its turn, is one that is quite useful. As far as it is concerned with retrieving finer details of the opinions, the phrase level sentiment analysis is a more challenging yet ultimately more fruitful task that involves shorter amount of text including expressions and phrases that can communicate a high degree of emotional intensity. These words may be adjectives, verbs and brief noun phrases that express an opinion. As an illustration, the use of very useful, too slow, and extremely satisfied are all such expressions. Even when the text is neutral the sentiment in phrases may be identified, with the help of this sort of analysis. This kind of analysis is finer. An example of a sentence contains a negative feeling that refers to the sentence which includes the phrase expected more. The sentence sounds as follows: I expected more with the update but the new features are quite useful. A positive one is in turn a quite useful one. In the case of pinpointing the nuances of opinions, sentiment analysis, at the phrase level, is a harder but more potentially successful thing to tackle [8].

2.2.4 Aspect-Level Sentiment Analysis

As far as analysis is concerned, sentiment aspect level is the most mature and advanced level. It is the process of determining which specific characteristics/feature of a product or service are being discussed by the users, which are known as aspects, and of concluding as to the sentiment opinion that users have indicated about each aspect of the product or service. This proves very useful in the case of reviews, when several distinguishing traits can be discussed. Indicatively, the model finds two aspects in the line; The user interface is clean but the load time is taking way too long. The attitude depicted in the "UI" is positive and the negative opinion is represented by the load time aspect. Using such textual contents, so that, in the mobile app reviews, i.e. people post their challenges with an app, its usability, speed, bugs and

battery, this works is lying to be analysed. As compared to evaluations that are blindly interpreted [9], [10], developers can still get the targeted feedback using aspect-based sentiment analysis.

2.3 Feature Extraction

In the field of natural language processing (NLP), feature extraction refers to the process of transforming unstructured text into a format that can be understood by machines and then used for additional statistical analysis. Given that the algorithm is unable to interpret text in the same manner that humans do, it is necessary to extract features from it that can be used to convey the meaning of the text. The frequency of words, the importance of words, the grammatical pattern, and the structure of sentences are all examples of these features. When it comes to improving the performance of sentiment analysis models, it is essential to have efficient feature extraction in order for these models to be able to recognize emotional tones, patterns, and meaningful information from text data [11].

2.4 Term Frequency (TF) and TF-IDF

The frequency of the appearance of a term in a document is also known as the term frequency (TF). One such line would be the one that goes like The app is fast, smooth, and it is fast-loading. In this sentence, we can see that the word fast comes twice, this means that it is most often used as compared to other words. This is the idea on which this ranking is based because a term is said to be more important when it appears more commonly in the text. Words, though, which are said to be general, like the is, or the, or app, may contain only a little bit of sentiment. According to this reason the TF-IDF (Term Frequency-Inverse Document Frequency) algorithm ranks a weight of each word in terms of the extent to which it stands out when compared to a sample of documents. The TF-IDF algorithm allocates more weights to the words that are not common yet common in a given document but are rare and common among various documents. In the example of an app review, e.g., the word crashes may have a great TF-IDF weight because it is a highly correlated variable with negative feeling and may not be used in each and every review [12].

2.5 Bag of Words (BoW)

One of the techniques is the Bag of Words (BoW) model, where a text is encoded as a set of words in it (also called a bag of words), disregarding grammar and word sequence. It cares only

about the issue of the presence or absence of recognizable words and how they occur often in the text. As an example, we will use two reviews, which are:

- *Review 1: "The app is good and fast."*
- *Review 2: "Fast performance but not good support."*

The vocabulary that BoW will form will include all the words (e.g., app, good, fast, performance, and support) and will count how many times each of these words is used in both the appraisals. BoW works well in the task of general sentiment patterns, although it is used in combination with algorithms like the Naive Bayes or the SVM [13]. Although it does not consider the phrase structure, BoW is applicable in this area.

2.6 Word Embeddings

Word embeddings are a technique applied in the natural language processing (NLP) field in order to properly present the words as numerical vectors, therefore, allowing machines to understand and manipulate the words. Unlike simple models like Bag of Words or TF-IDF, word embeddings can be used to capture the phonetic significance of words along with the context, unlike simple models, which only consider the frequency of particular words [14], [15]. With these type of representations of words, similar words to each other like "happy" and "joyful" can be represented in a high dimensional space and can be compared to each-other. The understandable behind this is that similar words (that is, words that have similar meanings) will be almost proximate even when their meanings are not exactly similar. Word embeddings have been important in the field of sentiment analysis, machine translation, and question answering to play a major role in considerably enhancing the results of models [16].

2.6.1 Word2Vec

Word2Vec is believed to be one of the first and the most influential models of word embedding. It was developed by Google in 2013 and it can tell the meaning of the words by the context they are applied in. Word2Vec has two training options: one is called CBOW (Continuous Bag of Words), where a word is predicted by words, surrounding it, and the other is called Skip-Gram where it predicts words, surrounding a word. As an example, in the model, the word as is combined with the word easy, and the word use is frequently combined with the word easy and thus, probably, same meanings are related to each other. In this sentence (The app is very fast and easy to use) this is evident. The final product is a vector embedding of every word which reflects well how they are used in actual language that is taken into practice.

Some of the relationships featured in Word2Vec vectors include the following: king - man + woman = queen [17].

2.6.2 GloVe (Global Vectors for Word Representation)

Another model of word embedding that was devised at the Stanford University is referred to as GloVe. GloVe is trained using world statistics in the whole set of words, unlike Word2Vec, which is trained to learn the meaning of the words found in the immediate vicinity (words that are in close proximity). It achieves this by building a word co-occurrence matrix or the frequency with which words occur together occurs in all the documents. Subsequently, it utilizes mathematical tools to transform that matrix to word vectors that can be of useful value. In case, e.g., the word battery is overused in connection to the word long and the word life in millions of Sage combinations, GloVe will place such words too close to each other on the vector space. Through this global context, the model can understand more finer details of relationships. GloVe can be applicable especially in the retrieval of information in large scale corpora like the data retrieved through twitter or Wikipedia [18].

2.6.3 FastText

FastText is an extension of Word2Vec because it was developed by Facebook AI Research and introduced to the world. In comparison to Word2Vec, it is superior since it takes into account not only full words but also information about subwords (for example, character n-grams). This indicates that FastText is able to comprehend the meanings of words that are morphologically similar to one another or even ones that it has never encountered before. For instance, if the model has been exposed to the terms "running" and "runner," it is able to ascertain the meaning of "runs" or "runnable" by utilizing the components that are common between the two words (run, "ing," and "er"). When analyzing noisy data, such as material from social media platforms or reviews that contain mistakes or slang, this is a very important ability to have. Out-of-vocabulary terms are something that many other embedding models are unable to manage, but FastText is able to handle them [19].

2.6.4 ELMo (Embeddings from Language Models)

The Allen institute of artificial intelligence came up with a contextualised word embedding model called ELMo. It does this therefore shows that it produces different embeddings of the same word in different phrases according to which it is written. Unlike Word2Vec and GloVe, which assign single fixed vectors to the words in their vocabulary, ELMo uses deep

bidirectional LSTM, a type of neural network, to compute word vectors, operative of the situation in which they occur. Words such as river bank and savings bank are some examples of words that will be represented in the ELMo with multiple vectors. These two words include the word bank. This enhances and brings ELMo closer by being more powerful and more accurate in tasks wherein the meaning of words in a variety of contexts needs to be ascertained, including answering a question or sentiment analysis [20].

2.7 Sentiment Analysis Approaches

Figure 2.2 depicts the overall categorisation of methods to sentiment classification, which can be further subdivided into two categories: methods that rely on machine learning and methods that rely on lexicons. The methods available under the branch of Machine Learning-based include Supervised Learning, Unsupervised Learning, Semi-Supervised Learning and Deep Learning. Depending on the nature of data that these learners are training upon, these learners may train sentiment models with labelled data or unlabeled data. And the Lexicon-based technique, in its turn, may be further divided into two groups, one of them being the Dictionary-based technique and the other one being Corpus-based Approach. These two methods are based on the predefined sentiment lexicons in an attempt to determine the text polarity. Historically and differentiating methodologies used to analyze sentiment may be regarded along these two broad lines, which go through the rule-based systems, to the more data-driven and automated models.

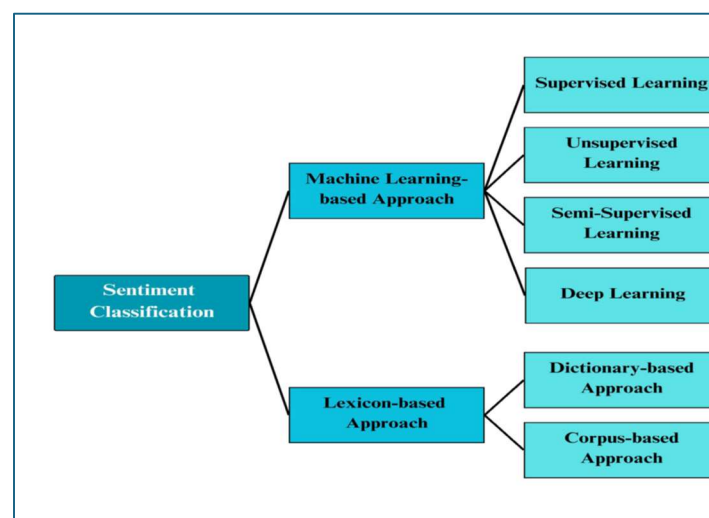


Figure 0.2: Sentiment Analysis Approaches

2.7.1 Machine Learning Approach in Sentiment Analysis

In the case of sentiment analysis, the Machine Learning (ML) method presupposes teaching a computer model to interpret the emotions expressed through written language through the knowledge acquired through data. It does this by means of algorithm, which identifies patterns in text which has been tagged or not and then labels the text patterns as being positive, negative or neutral. The fact that machine learning algorithms can learn through contexts, process complex phrases, as well as get better with additional data collections is a major merit compared to rule-based or lexicon-based systems. There are four methodologies namely deep learning, supervised learning, unsupervised learning and semi-supervised learning that are already being applied in the sphere of machine learning as far as sentiment analysis is concerned [21].

2.7.1.1 Supervised Learning

In supervised learning, the model is trained on a dataset that contains text data and corresponding sentiment labels [22].

The algorithm learns to associate certain words or patterns with sentiment classes. Common supervised algorithms used in sentiment analysis include:

- **Naive Bayes:** A probabilistic classifier that assumes features (words) are independent. It's fast and performs well with small data.
- **Support Vector Machines (SVM):** It finds a boundary (hyperplane) between sentiment classes in a high-dimensional space.
- **Logistic Regression:** A statistical model that predicts probabilities of classes (positive, negative, neutral).
- **Decision Trees / Random Forests:** Rule-based classifiers that split text features to reach a decision path.

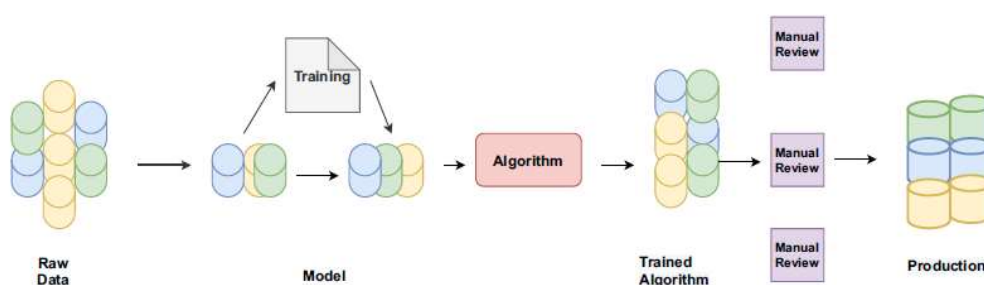


Figure 0.3: Process of Supervised ML Approaches

2.7.2 Machine learning Models

2.7.2.1 Naive Bayes

Naive Bayes has been used and is widely known to be one of the simplest and common algorithms in the sentiment analysis area [23]. This is because it is a probabilistic model and therefore, it includes predictions about the sentiment given the probability of the frequency at which words occur within each group of sentiment (e.g. positive, negative, or neutral). The idea here is to find the probability that a certain piece of writing is in a certain sentiment category by examining how often the words being used in the writing have previously appeared in other writings of the same sentiment category in the past.

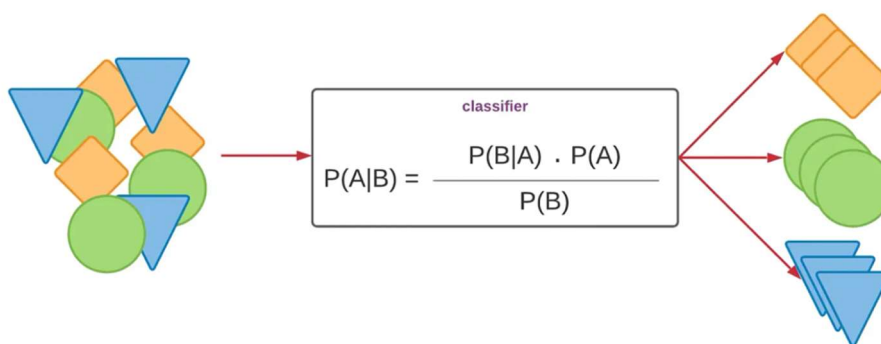


Figure 0.4: The architecture of Naïve bayes classifier [24]

This strategy is said tactic to be naive as it works on the premise that each and every word in the phrase contributes to the sentiment as being independent of the other words. Naive Bayes would make different senses of the words fast and reliable, in the line The app is fast and reliable, e.g. treat fast and reliable as unrelated indicators of sentiment, even though fast and reliable have different meanings. Despite the fact that this assumption is not entirely right when used in the real-world English, the algorithm is still able to work more than miraculously well, especially in cases where it is provided with a large amount of labelled training data.

In the training process, it will be informed on how many times a given word appears in each sentiment category. Then once a new sentence to be classified is typed, Naive Bayes uses the frequencies of words that it has learned so as to make an estimation on the classification of sensation that most probably the sentence will fit into. The result will be the determination of the output wherein the category with the greatest likelihood will be selected. Naive Bayes is an ideal option due to its speed, efficiency and its minimal number of training data, which can be applied on large scale application like categorizing thousands of checks that have been

provided on mobile applications. Nonetheless, it can be challenged where words have meaning that is dependent on the context or where sequence is crucial which is not depicted using this model.

2.7.2.2 Decision Tree

A Decision Tree is a subtype of machine learning model when it comes to making the decision regarding the data that is entered and it operates much like a flowchart [25]. It sorts out the data into small bits using a series of yes/no or condition based question formats and rearranges the data in a more effective manner. The data is directed through another tree of this question depending on the option it gives, based on which a particular character trait (the appearance of a word or statement, etc.) is studied. The process is continued until the model reaches a definite decision or category, either positive or negative sentiment that characterizes the sentiment. A decision tree might pose some questions in the context of sentiment analysis; some of them are: Does the statement include the word excellent? Does the sentence have such a word as bad? Based on the responses it narrows the number of options it has until it eventually moves the review to a category which matches the sentiment. These nodes of the tree are the questions (or conditions) with the branches being the representations of the possible responses and the leaf nodes being the representations of the final sentiment label.

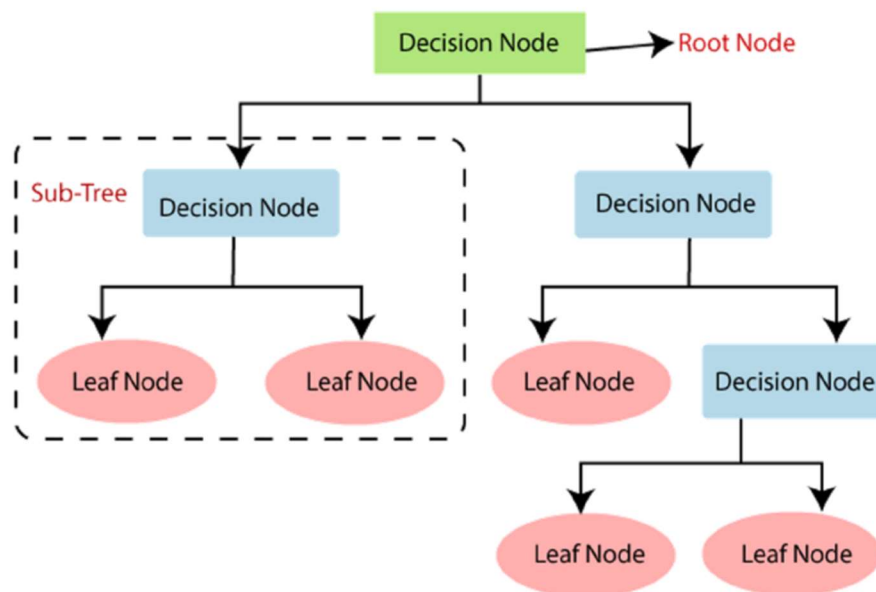


Figure 0.5: The architecture of Decision tree classifier [26]

This is why decision trees are a tool of special value when you intend to explain how a model arrives at the results it has since they are easy to understand and have easy interpretation. They can consider non-linear relations in the data and thus they can be more flexible unlike simple

linear models. However, they can also become too complex and overfit, especially when they center on it too much and are trying to perfectly recreate the training data of an excessive level of accuracy. Such techniques like pruning, i.e. trimming the tree, or such algorithms as Random Forest are often used to avoid this problem. Decision trees are useful and easy to use when identifying sentiment with structured data or pre-processed data of reading produced by a text. Overall, decision trees represent an efficient but simple approach.

2.7.2.3 Support Vector Machines (SVM)

SVMs often called Support Vector Machines are very useful machine learning models that are applied in numerous classification tasks such as sentiment analysis [27]. Among the main objectives of the support vector machines (SVM) is also to find the best border, also known by the name of hyperplane separating various classes of data. The information stored in this specific case is a collection of different sentiments, including positive or negative. This choice of boundary is so well made, as to produce the maximum gap, or margin, which can be created between the two groups. A larger margin means that it is more likely that the model will apply to the data that has not been observed in the past.

supporting (very useful in high-dimensional spaces, like those observed with text data) does that is, each word may be regarded as a feature, support vector machines (SVMs) are very useful. The model only takes care of the most significant data points that are known as support vectors and the ones that are closest to the decision boundary. These items are quite crucial as they indicate the location of the separator hyperplane and the direction that it is taking.

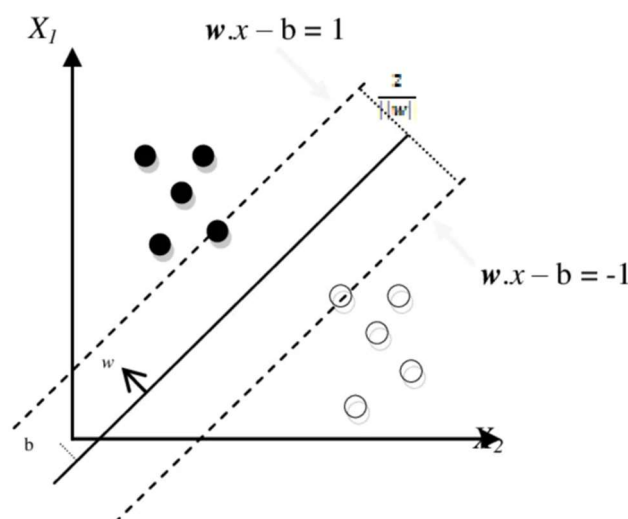


Figure 0.6: The architecture of Support vector machine [28]

One of the strengths of the support Vectors machine (SVM) is that, it can address both the linear and non-linear classification problems. The support vector machine (SVM) involves the use of mathematical functions, also referred to as kernels, to project the data into a different dimensions that have a distinct boundary. It comes in very handy where the data is not divisible by any straight line, which is a condition of linear separability. Consequently, it is highly flexible and suited to perform challenging duties such as the one on sentiment classification. The actual performance of the support vector machines (SVMs) is presented to be very good, especially when there are many more features than samples involved, the way it almost always happens in natural language processing. Nonetheless, to achieve optimal results, they are prone to consume much of the computational resource and demand accurate parameters adjustment.

2.7.2.4 Random Forest

Random Forest is a complex machine learning algorithm that is made efficient through combining the results obtained by many decision trees to come up with a decision which is more precise and constant [29]. The fact that it belongs to a group of models referred to as ensemble methods implies that it builds a huge number of models and then blends the results of those models so as to improve the overall performance. Although one decision trees may err and or overfit the training data, the concept argues that combining the decisions of many trees may reduce the amount of errors which transpire and moreover enhance the quality of the model.

The algorithm starts with creation of a number of separate decision trees and then proceeds to create a random forest. But instead of utilizing the full dataset with every tree, it creates its own sample of the data with the tree, randomly sampling the data with replacement, which is referred to as bagging. This will enable it to reflect more on the data of every tree. Moreover, at any given point in time that every tree is being built, only a random set of features is considered at every step. It is thanks to this chance that the trees can differentiate themselves among each other and do not repeat similar mistakes towards each other.

When the trees are all trained the Random Forest model will start making predictions by taking into account all the results as one. In the problem of classification such as sentiment analysis, it frequently involves choosing the most voted of the trees. As an example, when most of the trees are saying that a review is positive then the ultimate outcome will as well be given as positive.

One of the most important pros of this method is the ability of the Random Forest to handle both noisy data and massive feature spaces, as both should be observed quite often during text

analysis. Alongside this, it also has a particular performance of capturing complex patterns and relationships among words without significant architecture of feature. The large number of trees it needs to be trained, conversely, can raise its computation and interpretation cost and complexity compared to simpler models, like a logistic regression or a pure single decision tree. Random Forests are commonly used in the sphere of sentiment analysis where the accuracy and reliability are valued more than the transparency of model. This is particularly the case with terms of the categorization of large volumes of user reviews, social media posts, or customer feedback.

2.7.2.5 Logistic Regression

Logistic Regression is a widely used statistical model for classification tasks, particularly useful when the goal is to predict whether something belongs to one of two categories like positive or negative sentiment [30]. Despite the name, logistic Regression is used for classification, not Regression. It works by estimating the probability that a given input belongs to a particular class and then assigning the corresponding class label.

In sentiment analysis, the model learns from labelled training data, where the sentiment of each text (such as a product review or tweet) is already known. Each word or feature in the text is assigned a numerical value, and logistic Regression uses these values to calculate the probability of the text being positive, negative, or neutral. If the probability for “positive” is higher than 0.5, the model classifies the text as positive; otherwise, it is classified as negative. What makes logistic Regression effective is its ability to model the relationship between input features and the likelihood of an outcome in a very efficient way. It is particularly suitable for datasets with a large number of features, such as word frequencies in documents, which is common in natural language processing.

One of the advantages of logistic Regression is that it is easy to implement and interpret. The model’s output reveals the contribution of each word or feature to the final prediction, making it valuable for researchers who seek not only high accuracy but also insight into why a text was classified in a certain way. However, it assumes a linear relationship between the features and the log-probability of the outcome, which may limit its performance in highly complex or non-linear problems.

Overall, logistic Regression is often used as a baseline model in sentiment analysis tasks because of its simplicity, speed, and interpretability, making it a solid choice for both minor- and large-scale text classification problems.

2.7.3 Deep Learning Approach in Sentiment Analysis

Deep learning is a subfield of machine learning that uses neural networks with multiple layers to automatically learn complex patterns from text data. It is especially powerful in handling long sentences, contextual meaning, and sarcasm.

Common deep learning models include:

- **Convolutional Neural Networks (CNNs):** Originally used for images, CNNs can also identify local features or n-gram patterns in text (e.g., “very bad service”) that influence sentiment.
- **Recurrent Neural Networks (RNNs):** Especially Long Short-Term Memory (LSTM) models, are used to understand sequence and context of words. For example, they can learn that “not good” is negative, even if “good” is usually positive.
- **Transformer-based Models (e.g., BERT):** BERT understands context by looking at the entire sentence at once and is one of the most accurate models for sentiment tasks today [31].

2.7.4 Deep Learning Models Approaches to Sentiment Analysis

2.7.4.1 Convolutional Neural Networks (CNNs)

The convolutional neural networks started processing images, but also peruse feelings in the text. Every filter reads a couple of adjacent words and it reminds a sliding window [32]. It is able to identify words like very good or not happy. The network maintains the best matches, layers them, and sends the layer on to a layer which selects the last label.

It is quick and is appropriate with short posts, tweets and chat messages. It has limits. The filters concentrate on the words immediately around each other, thus the model can overlook clues that are located long sentences or multiple lines of words. Recurrent nets or transformers are also followed by wider links but require longer time and power.

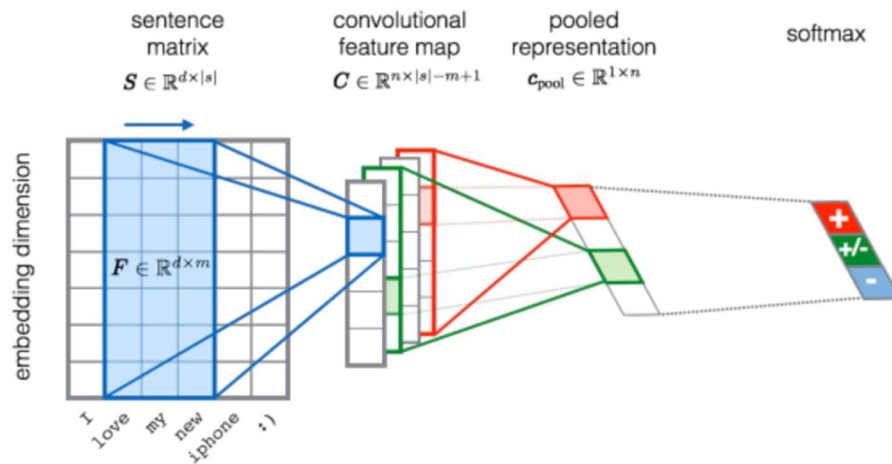


Figure 0.7: The architecture of Convolutional Neural Network for sentiment analysis [33]

2.7.4.2 Recurrent Neural Networks (RNNs)

Recurrent neural networks (RNN) are used to read words in a sequence. Adding a word to the model updates an internal state which records what the model has already observed [34]. This running memory is used to trace context within the network. In sentiment work, such a memory associates earlier words with later ones and therefore the system can identify changes of tone within a sentence. Even simple RNNs cannot overcome an obstacle. Gradients may die out in the course of training. Long sentences are then flooding the memory and the network forgetting initial cues. An extended critique can begin by flattery and end with an attack. Enhanced bottleneck RNN causes that turn to be overlooked and incorrectly labelled.

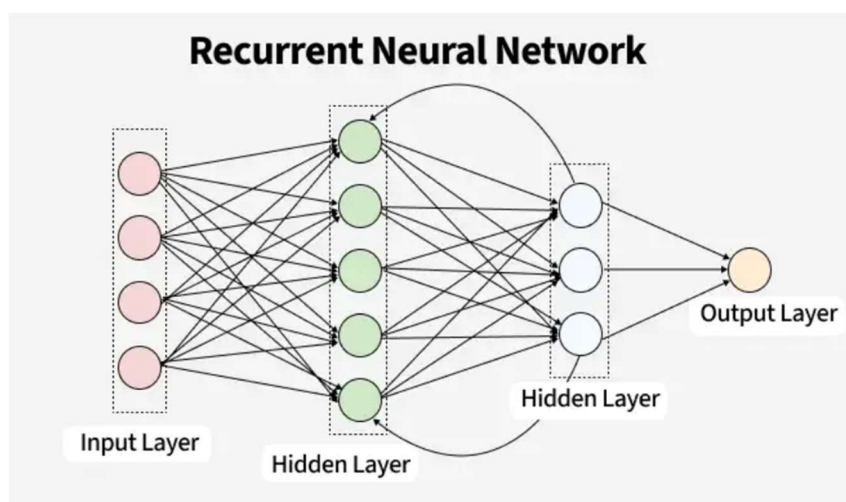


Figure 0.8: The architecture of Recurrent Neural Network [35]

2.7.4.3 Long Short-Term Memory (LSTM)

LSTM is a class of RNN that is built to overcome the shortcoming of the regular RNNs [36]. A recurrent neural network is a type of neural network in the form of LSTM. However, normal RNNs forget information after a long time. LSTM rectifies this problem by having small units known as gates where data is stored, discarded or passed forward. The gates have made the model take note of the occurrences that are longer in text such as a good positioning and a bad placement. In sentiment analysis, LSTM helps to monitor the tone of emotion throughout whole paragraphs or discussions and it is efficient with short and long texts. It is a deep learning model with one of the broad applications in SA tasks, attributed to its capability to deal with contextual change and negation.

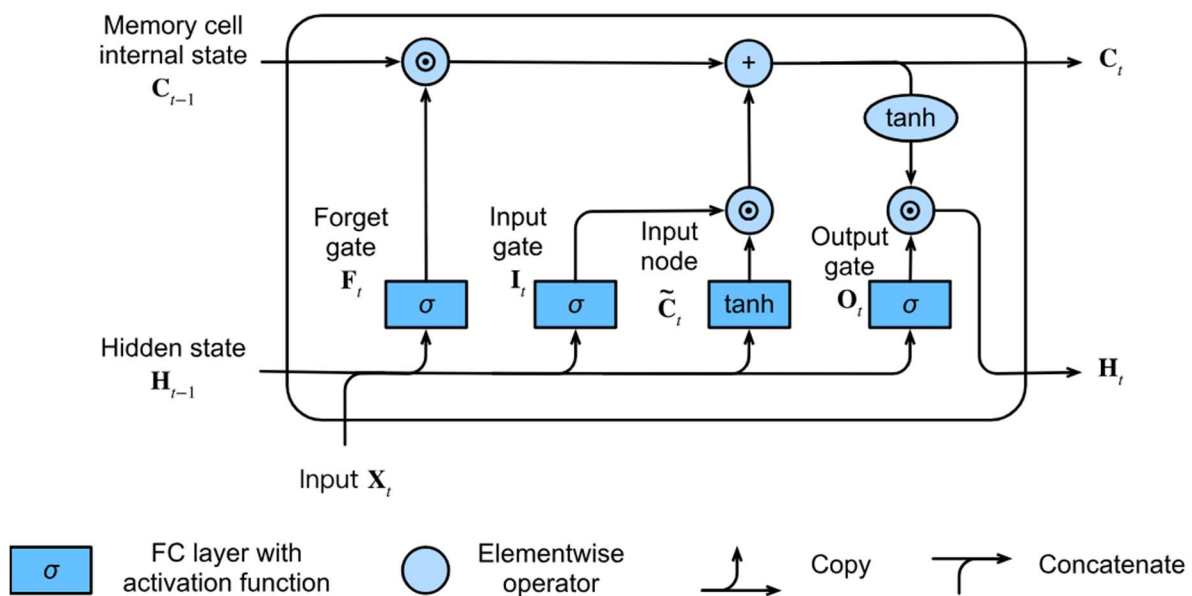


Figure 0.9: The architecture of LSTM [37]

2.7.4.4 Gated Recurrent Unit (GRU)

GRU is a simpler and faster alternative to LSTM that also solves the problem of vanishing gradients in RNNs [38]. It combines the memory and forget gates into fewer components, making it less complex while still being effective at capturing sequential patterns. GRUs are often used when computational efficiency is important or when the dataset is not large enough to train more complex models like LSTM or transformers. In sentiment analysis, GRU models perform similarly to LSTM in many cases and are especially useful for real-time or mobile applications where memory and speed are critical.

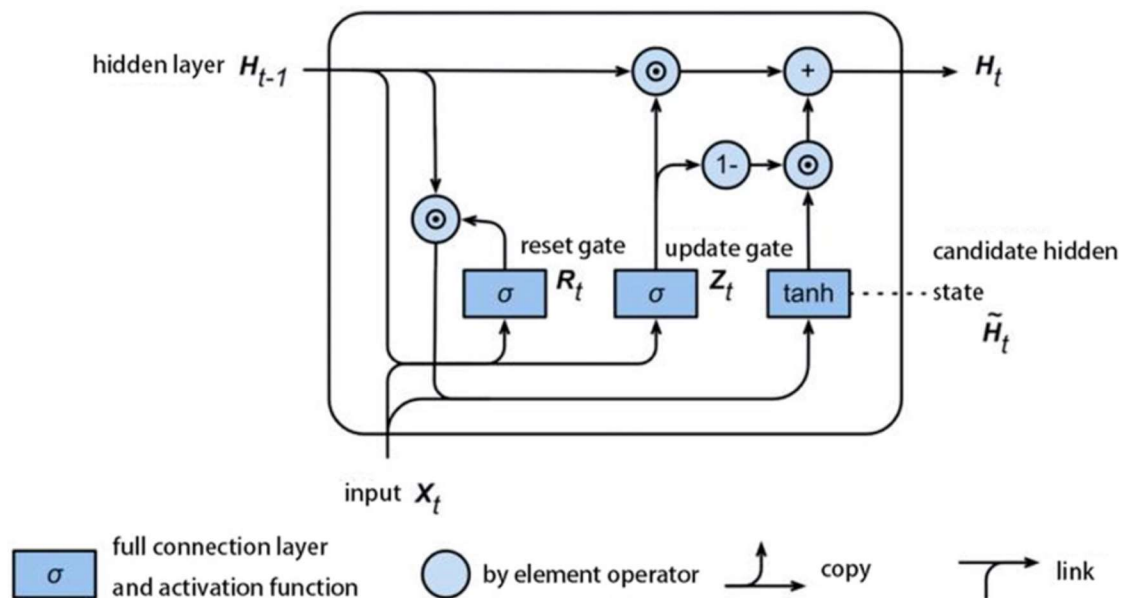


Figure 0.10: The architecture of GRU [39]

2.7.5 Large Language Models (LLMs Approaches to Sentiment Analysis

2.7.5.1 Introduction to Large Language Models (LLMs)

Large Language Models (LLMs) learn language from huge text collections [40]. They include GPT from OpenAI, BERT from Google, and LLaMA from Meta. Each network holds billions of parameters. These weights store patterns about grammar, facts, and tone. In sentiment analysis, LLMs excel. They study full sentences and paragraphs rather than fixed word lists. This depth lets them catch irony and shifting context.

Transformers give LLMs this skill. Self-attention lets the model weigh every word against every other word in one step. The process captures links between distant terms. The model then flags sentiment shifts, double meanings, and subtle cues. Engineers can fine-tune an LLM for a new dataset with only a brief training run, or use it in zero-shot form.

Vaswani et al. introduced the transformer in 2017. The design replaced recurrent nets such as RNNs and LSTMs. It processes the full input at once, so training runs on many GPUs in parallel and finishes sooner. The lack of recurrence also helps the model learn long-range patterns.

During training the network views the entire sentence, paragraph, or document as a single sequence. This parallel view cuts training time and raises accuracy on large corpora. Modern

LLMs benefit from that speed and size. New variants keep improving sentiment analysis, question answering, and text generation.

2.7.5.2 BERT (*Bidirectional Encoder Representations from Transformers*)

BERT is a Google-made model that is transformer based which reads in both directions (left and right) as compared to earlier models that could only read in one direction [41]. This two-way training allows BERT to know the entire context of a word including words preceding it and those that follow it. The pre-training of BERT is done on large text corpora in two training tasks: masked language modeling (guessing the missing words) and next sentence prediction (interpreting sentence relationships).

BERT is also very effective in sentiment analysis as it addresses delicate meanings, sarcasm, and the contextual element of interpretation of words. In the example, the term great may have quite a different meaning when it comes before the word not or barely.

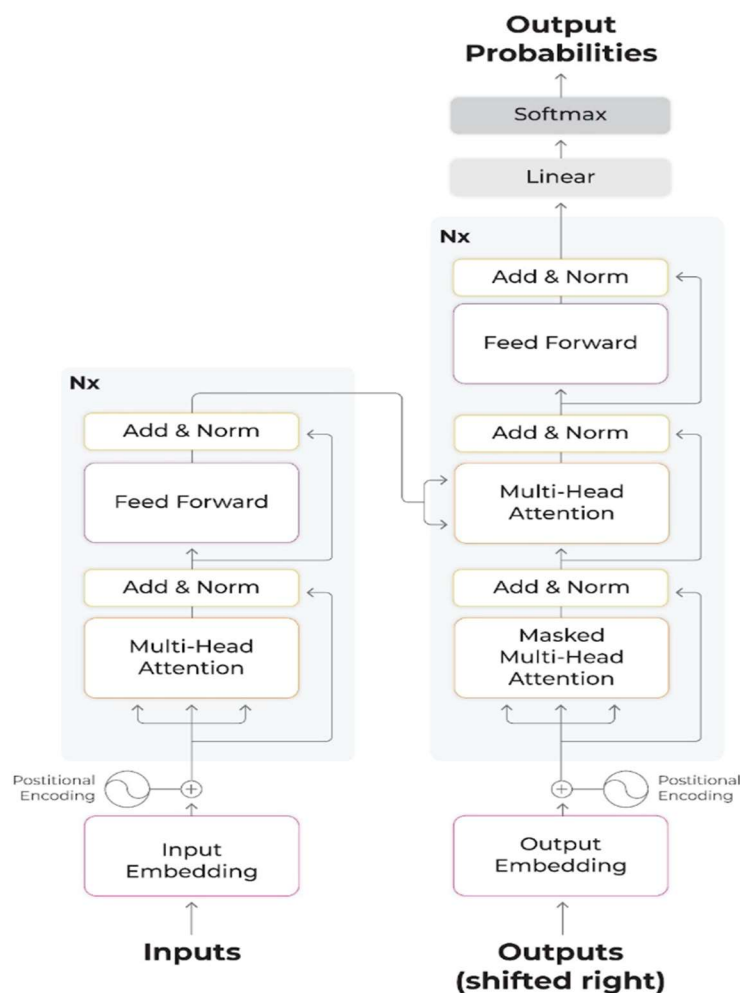


Figure 0.11: The architecture of BERT [42]

2.7.5.3 GPT (Generative Pre-trained Transformer)

GPT is a GPT text generator, which is a one direction transformer that produces text left to right [43]. It is trained to know the word that comes next in a sequence and thus it is really good at language generation tasks. Compared to BERT, GPT aims at generative workloads, although it also achieves good classification performance on the classification workload that BERT is trained on, e.g., sentiment analysis on fine-tuning of labelled datasets.

The model is easy to train and is large. The fact that it comprehends grammar, world knowledge, and even conversational tone allows the model to understand the world with significant volumes of data. GPT is capable of detecting very complicated sentiments, irony and opinion across a multiplicity of sentences or even documents in SA tasks.

2.7.5.4 RoBERTa (Robustly Optimized BERT Approach)

RoBERTa is an improved version of BERT created by Facebook AI. It keeps BERT's two-way transformer but drops the "next sentence prediction" stage [44]. Facebook AI fed it ten times more text: 160 GB from Common Crawl plus BookCorpus and news. They also raised the batch size and the learning rate, then trained for 500 k steps. These choices let the model learn richer word patterns and longer contexts.

Benchmarks confirm the gains. On GLUE, RoBERTa-large scores 90.2, while BERT-large scores 82.2. On the SST-2 sentiment task, RoBERTa reaches 96.4 % accuracy, two points above BERT. Similar jumps appear on emotion and intent datasets like GoEmotions and SNIPS. For a production pipeline, fine-tune RoBERTa on labeled reviews or tweets for three epochs. Use a learning rate near 2×10^{-5} and a batch size of 32. Freeze nothing; the full model adapts fast. With these settings, you can expect one to three accuracy points over an equally tuned BERT model.

2.7.5.5 DistilBERT

DistilBERT is a smaller and faster implementation of BERT that uses fewer resources and maintains the majority of performance [45]. It is constructed through a process termed knowledge distillation whereby a smaller model is trained to replicate a bigger model. DistilBERT is especially beneficial in system environments such as resource-heavy (such as mobile phones) or real-time. DistilBERT, despite using fewer parameters, demonstrates high levels of performance in such applications like text classification, as well as sentiment analysis, which is why it is popular in the industry.

2.7.5.6 XLNet

XLNet is a second transformer model but it is enhanced over the BERT model, as its algorithm is known as permutation-based training, whereby all orderings of words are taken into account in the model during training [46]. This assists XLNet in learning context and sequence better without masking or forgetting the information about the order of words. It has a tendency to be better at several NLP tasks, and especially when the context and word order are important to sentiment classification, it is better than BERT.

Table 0.1 Comparison Matrix of Sentiment Analysis Methods

Category	Model/Method	Key Characteristics	Advantages	Limitations	Common Use Cases
Machine Learning	Naive Bayes	Probabilistic model assuming word independence	Fast and works well on small datasets	Assumes features are independent (not realistic for language)	Spam filtering, basic SA
	Decision Tree	Tree-like structure using rules to classify sentiment	Interpretable and easy to visualize	Prone to overfitting	Structured text classification
	Random Forest	Ensemble of decision trees	Higher accuracy and robustness	Slower, less interpretable	Large text datasets
	SVM	Maximizes margin between classes	Performs well on high-dimensional data	Requires careful tuning; slower on large data	Review classification
	Logistic Regression	Predicts probability of sentiment class	Interpretable and efficient	Limited expressiveness for complex data	Binary sentiment tasks
Deep Learning	CNN	Uses filters to extract local features from text	Good at detecting n-grams and phrases	Weak at capturing long-term dependencies	Short text sentiment

	RNN	Processes sequences word by word with memory	Maintains word order and context	Struggles with long sequences	Sequential sentiment tasks
	LSTM	Gated RNN variant that remembers long-term dependencies	Handles long and complex sequences	Computationally heavy	Paragraph/document-level SA
	GRU	Simplified LSTM with fewer gates	Faster than LSTM; good for small data	Slightly less expressive than LSTM	Real-time sentiment tasks
LLMs / Transformers	BERT	Bidirectional transformer trained on masked word prediction	Strong contextual understanding	Requires fine-tuning; resource intensive	Complex sentence-level SA
	GPT	Left-to-right transformer trained to predict next word	Excellent at text generation and coherent output	Unidirectional (less context-aware); large model size	Generative SA tasks
	RoBERTa	Optimized version of BERT with better pretraining	State-of-the-art performance	Requires extensive computing power	Sentiment benchmarks
	DistilBERT	Compressed BERT with fewer parameters	Fast and lightweight	Slight performance drop from full BERT	On-device SA
	XLNet	Permutation-based transformer with full context	Handles sequence order and context well	Complex training; slower than BERT	Long text and context-heavy SA

2.8 Evaluation Matrices

To assess the performance of various machine learning models for sentiment analysis, standard evaluation metrics such as Accuracy, Precision, Recall, and F1-Score are used.

1. Accuracy

Accuracy measures the overall correctness of the model. It shows how many total predictions were correct.

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \quad (1)$$

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

2. Precision

Precision tells how many of the predicted positive results are actually correct. It reflects the model's ability to avoid false positives.

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) \quad (2)$$

3. Recall

Recall indicates how many of the actual positive cases were correctly predicted. It reflects the model's ability to avoid false negatives.

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \quad (3)$$

4. F1 Score

F1 Score is the harmonic mean of Precision and Recall. It balances the two when there is an uneven class distribution.

$$\text{F1 Score} = 2 \times [(\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})] \quad (4)$$

2.9 Summary

In the present chapter, sentiment analysis along with its importance, its granularity (the document level down to aspect-level), its application to domains of mobile applications analysis, social media analytics and customer satisfaction have been thoroughly informed. Various feature selection mechanisms including Bag of Words feature extraction, TF-IDF feature extraction feature extraction, advanced word embeddings (Word2Vec, Glove, Fasttext

and ELMo) were outlined in order to convert the unstructured data to data that could be read by a machine. The chapter also mentioned the other types of sentiment classification, such as different traditional machine learning models (Naive Bayes, Decision Trees, SVM, Logistic Regression) and deep learning models (CNN, RNN, LSTM, GRU), and the most recent models of transformer-based large language models (BERT, GPT, RoBERTa, XLNet). Finally, the chapter also touched upon the practicable tasks and challenges such as sarcasm, idiom, lexical ambiguity and context dependency. The knowledge that is generated in this chapter is the one upon which the other chapters are built. It preconditions the area where sentiment analysis techniques are provided and analyzed.

Chapter 3

Literature Review

The sentiment analysis has grown significantly in the last ten years due to the skyrocketing number of user-generated content in the majority of social media, review websites, and other online communication services. As the new methods and techniques are invented to extract worthwhile knowledge in this giant expanse of text information, the issue of creating effective models of the sentiment classifications becomes more tinderious than ever. The literature review within this study will show the analysis of the recent development of the sentiment analysis field in the context of machine learning (ML), deep learning (DL), and large language models (LLMs). The literature review will compare models according to their accuracy, interpretability, computational cost and generalize on different tasks e.g. product reviews, social media posts and multilingual data.

Such insights will be incorporated to inform the design and future of this thesis particularly on what forms the most promising and viable ways of creating a useful sentiment analysis system. The further sections are organized into three major parts involving the literature about conventional machine learning, deep learning, and transformer-based LLMs; and finally, the identification of gap in the research and why one is to work on the proposed project.

3.1 Machine Learning Models

The research study performed by Jannah et al. [47] is realized to examine the performance of Naive Bayes and Support Vector Machine (SVM) in sentiment classification of e-commerce product reviews of beauty products such as Shopee. To determine the sound performance evaluation, the authors employed cross-validation by using 10-fold. The preprocessing pipeline involved lemmatization and TF-IDF vectorization which converts raw text into numeric features which are applicable with machine learning. Using a synthetically generated dataset was a subapproach that was based on being as close to artificially generated reviews of the Shopee platform as possible and had their labels manually assigned, positive, negative, or neutral. On the comparison of the results, SVM model outperformed the Naive Bayes and the average accuracy was 86.14, but Naive Bayes gave the accuracy 85.53 in some folds. Though the research has produced good results, there are certain limitations. It does not provide the analysis of error and feature importance which are necessary to understand the model behavior and misclassifications. Besides, it might also not be mentioned that the research study has been carried out on a synthetic dataset and has not been verified on actual Shopee data; this leads to the absence of the ecological validity and generalizability of the results.

In the work by Juliane et. al. [48], they exploited the Naive Bayes classifier algorithm in a Twitter sentiment analysis via the use of machine learning pipeline that is conventional. The

method was to get the tweets using an API of Twitter and then underwent the preprocessing which involved lowercasing the tweets, stemming and removal of stop-words. The preprocessed text was then transformed into features using term frequency (TF) and the process of classification was finally attained using the Weka software environment. The information consisted of twitter remarks denoting opinions of individuals on various social issues. It was divided into a training set and a small validation set with final 30 tweets as a validation set. The test score was a much higher 93.33 on the training set but the validation resulted in a somewhat lower score of 86.66 indicating that the level of generalization was more or less good. The result of the research is however constrained in the light of the scarcity of the data collected where the validation set is limited in size consequently restricting the size of performance estimates. The authors did not also make comparisons of Naive Bayes against other models, including the SVM or the Logistic Regression that would have strengthened the analysis in regard to the bench mark data.

Azeraf et al. [49] researcher introduces a novel enhancement of the traditional Neural Naive Bayes (NNB) to the classical Naive Bayes concept as the representation of shallow neural representation is more situation-specific in determining semantics. This model operates with pre-trained FastText embeddings, unlike the traditional ones that operate on Bag of Words or TF-IDF, as a result of which it can also consider the word-level context when it performs the classification. The model has been applied to the IMDb movie reviews database and sentiment was predicted in the binary mode (positive or negative). Their results show that the NNB classifier was highly valuable in decreasing the error of classification 4.5 times when compared to the normal Naive Bayes and scores were very impressive when it comes to long or context-dependent reviews. The research suffers limitations in spite of good performance. It does not have a comparison to other direct popular models such as Logistic Regression or SVM, using the same embeddings. Besides, the experiment involved the use of a single dataset, thereby constraining its validity generalization nature exhibited by the model.

A recent study [50] applied Support Vector Machines (SVM) and SMOTE Tomek in classification of sentiments in fashion e-commerce review. The post-processing e.g. cleaning, TF-IDF vectorization SVM model has achieved 92 percent accuracy, 0.89 precision, 0.89 recall, 0.89 F1-score as compared to the insignificant degree of accuracy 0.68, F1-score 0.71 used by the non-augmented baseline. No multi-classification has been performed of binary (positive/negative), no exploration of a neutral sentiment, or the comparison of other classifiers was done in the same oversampling strategy.

The study article by Wirawan et. al. [51] addresses the issue of identifying cyberbullying in Instagram comments based on SVM data augmentation with an easy data augmentation (EDA). The result of the enriched data increased the resistance, accuracy at 92.5 and lastly another improvement was 2.5 percent beyond to the normal SVM model. It aims at one of the domain-specific datasets but numeric improvement is not compared to the specification of logistic regression or decision tree baselines.

A Logistic Regression, Decision Tree, Random Forest and SVM in Turkish languages were used to conduct the sentiment analysis of the reviews on the e-commerce products [52]. SVM was doing comparatively better than others but Logistic Regression was still capable of presenting a fair amount of performance in various categories which again proved to be handy in bigger text classification too. Even though the few models were compared, hyperparameter optimization, model interpretability analysis of the classifier behavior was not carried out.

Danyal et al. [53] did a comparative evaluation of multiple machine learning algorithms to perform sentiment analysis in a wholesome manner with the references to the Linear Support Vector Machines (SVM), Multinomial Naive Bayes (MNB) and Logistic Regression (LR) among the other baseline classifiers. The study aim was to identify the most effective traditional classifiers when it comes to popular sentiment data. It was assessed using two gold standards of datasets: IMDb 50K dataset which comprises of 50,000 positive or negative movie reviews, and Sentiment Polarity Dataset where a long-established benchmark of positive and negative sentence-based reviews is used.

Before training each of the models, the researchers used the standard preprocessing methods, which included tokenization and TF-IDF vectorization. One of the models that had the best performance with 89.48 accuracy was linear SVM. However, Logistic Regression with hyperparameter tuning, especially together with SVM, was marginally ahead of SVM with an accuracy of 90.27%. In this case also, Passive Aggressive Classifier was expected to perform in a competitive manner when fine-tuned indicating that simple, linear models can still be very useful in sentiment classification particularly on clean and well-labeled data such as IMDb.

Irrespective of the encouraging findings, the research has some weaknesses. Among the weakness is that the accuracy in performance was mostly considered in the analysis, whereas there was little concern on the precision, recollection or F1-score, which is acute when dealing with the imbalanced or real-world data. Second, the research failed to concern the model performance on multilingual data or the domain-specific reviews such as a product review, an app store reviews, or a social media text which are language diversified. Lastly, the models have not deployed more stamp-contextualized embeddings rather than more popular TF-IDF

base features which, in theory, would enable the classification performance of more convoluted texts.

Khan et al. [54] have made a comparison between the sentiments analysis of the Random Forest and Support Vector machine (SVM) classifier between a news and a Social media post. The aim of the research was to explore the workability of classical machine learning techniques in finding the classification of opinionated text utilizing different types of real world information. Both the models were applied by the authors to sentiment data produced with the pre labeled headlines of news items, samples of blog article and social media messages labelled as positive or negative. In obtaining features, standard natural language processing steps such as tokenization, stop-word removal and TF-IDF vectorization have been used. Collectively, SVM model worked slightly better as compared to the accuracy of the Random Forest with 80.39 percent and 78.56 percent, respectively. However, the authors noted that the Random Forest gave a more consistent output and was able to process data quicker particularly with a noisy and informal text, which is always the case with social media. The collective aspect of Random Forest allowed it to manage variability and outliers in a better manner than SVM which is likely to be affected by noise in the features.

All of the above are the strong points of the study but there are certain limitations. Specifically, the authors have not delved into the issue of polarity of phrases at the sectional level, e.g., sentiment of sentences or polarity of aspects, which limit the amount of information brought out by the text. All the decisions were done on a document level that was able to conceal the split sentiments in one piece of writing. Besides, the paper has not employed recently represented features, but only TF-IDF as a method in converting features into vectors. This restricts effectiveness of the models in deriving semantic relationship between words, idioms and contextual meaning. Also, it was also limited to binary classification (positive and negative) without regard to neutral and ambiguous ones, which play a critical role in the opinion mining subtleties.

The reviewed literature has also indicated that sentiment analysis is still extensively dependent on the conventional machine learning models. Naive bayes remains a strong baseline since it is simple to test and quick, but the assumption of independence of features may have an adverse effect on its behavior in more elaborate text. The Support Vector Machines are highly accurate in most situations where they have been augmented with data or oversampling process but are susceptible to the faulty data. Logistic Regression is interpretable and competitive in terms of results when it is tuned. Decision Trees and Random Forest express power and additionally, are more suitable on noisy or unstructured data but slightly lower than SVM. On the whole,

such models are good but in most of the studies they merely use simple text representations and the concept is to get binary classification. It has also been noted that more complex methodologies are required that may incorporate richer feature embeddings, aspect-level sentiment and utilize more diverse measures of evaluation.

In an experiment introduced by Singhal et al. [55], the Support Vector Machines (SVM) classification was contrasted with Naive Bayes, Logistic Regression, and the method of random forest when detecting fake reviews using the BERT embeddings. The research has utilized the Amazon product reviews data because the data has categorized the information on the reviews regarding the authenticity and fakery information. The reviews were pre-processed by the BERT embedding and it helped the models to comprehend the semantic meaning of words in the review context. The models were trained and evaluated using the same data and they led to a direct performance comparison.

The results indicated that SVM performed better than other models with a rate of 87.81%. It implies that SVM was the most effective model applied to the case of fake review detection as it has such features as the possibility to find optimal hyperplane on high-dimensional space. Despite the good performance of SVM, the authors have not compared the approach with other advanced algorithms like XGBoost or LightGBM that were considered to yield better performance concerning classification. Moreover, hyperparameter optimization was not researched, which could have given the further optimization of the models and potentially their positive work. These limitations mean that further research ought to take such additional models and refinement schemes into account in the recognition of fake reviews.

Despite the good results, there were serious limitations of the research. One of the significant research gaps was the lack of the error analysis and the feature importance rating. They are essential in the description of the model performance, the rationalization of the misclassification and the visualization of the features that contributed the most value relative to the model predictions. In addition, the study has been conducted on a fabricated data that was but an estimation of the real-life data. The results, as a result, do not guarantee, perhaps, the overview of the issues that happened when the model was applied to real Shopee data that may introduce increased noise and complexities. These are the gaps that signify the possibility of the future research including the application of the model to the real data and a more in-depth study of the model errors and feature play.

3.2 Deep Learning Models

Since unstructured user feedback, in form of app reviews, social media commentary, and digital ratings grow, deep learning has become extremely effective as a sentiment analysis instrument as well. Unlike machine learning models, deep learning models can automatically learn valuable patterns and textual relational dependency, therefore deep learning models can be best applicable in the analysis of the opinions of the user. CNNs, RNNs, and the Long Short-Term memory (LSTM) networks have been shown to exhibit a higher performance on the recognition of sentiments particularly when trained to process large and noisy data like reviews of a mobile app. The next section provides an in-depth literature review of recent articles which have applied deep learning in sentiment analysis problems on various types of user-generated review information. The research has also presented all the studies including their methodology, dataset, performance and research limitations.

Ahammad et al. [56], have created a multifaceted sentiment analysis of the user reviews of AI-based mobile applications of Google play store. These six deep learning architectures include CNN, LSTM, BiLSTM, GRU, BiGRU, and a hybrid Grenada of CNN+LSTM and have been examined by the authors. Their experiment was rather thorough and featured much pre-processing in the form of cleaning, tokenizing, and transforming data into vectors, training, and testing on the whole data. The data included 122231 user reviews and scores collected on the most well-known AI-based applications: ChatGPT, Lensa, Bing, and Alexa and manually labeled each of these reviews as positive, neutral, or negative. CNN + LSTM hybrid architecture model had the highest accuracy of approximately 92.1 that is closely followed by LSTM with an average of 91.97 and the other resulting model (BiLSTM, GRU and CNN) had a relatively lower accuracy. The paper also has an effective examination of the comparison of deep learning architectures in sentiment classification but emphasizes largely on accuracy. It lacks detailed analysis by sentiment classes (which would involve precision, recall, and F1-score) and goes no further in taking comparison with transformer-based models, such as BERT, and lightweight models that can run on a mobile device, which future opportunities can pursue. In order to analyze the performance of different models in classifying the effect of sentiment towards the reviews on the mobile application on the Google Play Store, a comparative research study was conducted by Samanmali and Rupasingha [57]. The authors contrasted three approaches, which are, the shallow Artificial Neural Network (ANN), Long Short-Term Memory (LSTM) network and the Support Vector Machine (SVM). They followed certain common stage of data processing such as tokenization, TF-IDF based vectorization and class

balancing in all so as to give same representation of all classes of sentiment. The information of approximately 33,000 user reviews about 15 most popular mobile applications went through as either being a positive or a negative one or as being a neutral review. The LSTM model was doing better than the ANN and SVM model and performed the best when it came to the capability of having high accuracy scores (in the range of 80-90) in different apps and also had the highest F1-score. Nonetheless, hyperparameter optimization was not explicitly described and no analysis on the performance of the models based on long form reviews was done. Also, it did not have any embedding-based applications such as Word2Vec or contextual application such as BERT which would have offered the semantic insights of the model.

In the article by Amirkhalili and Wong [58], the research is grounded on a sentiment analysis of the user applications to mobile banking applications utilizing the iOS App Store and Google Play Store. In order to complete this task, the authors have attempted to employ a mixture of both the traditional machine learning algorithms, including Multinomial Naive Bayes algorithm, along with Long Short-Term Memory (LSTM) neural networks to discover sentiment. Secondly, they also included Latent Dirichlet Allocation (LDA) to perform topic modeling to represent thematic structures, and sentiment patterns. The data was composed of 2,025 manually classified reviews of banks in Canada that offer mobile applications which were collected under five giant mobile banking platforms. The findings revealed that, the accuracy of the LSTM model at the reviews of the iOS app was almost 82 percent on the Naive Bayes, which produced an approximate of 77 percent accuracy at its reviews of the application in the Google Play. The paper showed the benefit of LSTM in detecting subtle sentiment patterns in the text. A number of limitations have been reported, the dataset size was not very extensive, and there were no other outcomes on the performance measures by the class e.g. precision, recall, or F1-score. Furthermore, it was region-specific, Canadian banking apps were taken into account, and it did not presuppose the usage of a transformer-based model that would be considered a better benchmark when it comes to comparison.

Recently, Zaman et al. [59] explored opinions of the users in the context of utilizing mobile Extended Reality (XR) experiences on the Google Play Store. The researchers employed deep learning classifications to do a sentiment analysis of the reviews with the objective of extracting the insights that relate to quality. Preprocessing To prepare the models with the text it employed a structured Knowledge Discovery in Databases (KDD) pipeline of steps such as; tokenization, stopping-word/stemmed enabled it to prepare the text it was going to feed them with. This list of data consisted of user reviews obtained in 2025 and it could be used to target the XR related mobile apps. It indicated that the neural network models used in the study generated higher

sentiment classification results than the conventional machine learning baselines. The summary of the results nevertheless did not state the specific performance results like accuracy, precision and recall or subdivide them into classes of sentiments. Moreover, there is also a significant limitation of the study, namely, the fact that the specific deep learning network used and whether it was LSTM, CNN, GRU, or some other format are not clearly stated, and the evaluation of the findings is not compared with the current state-of-the-art models, which makes the interpretability and reproducibility of the study results rather limited.

An article by Ranjan and Mishra [60] evaluated the application of sentiment analysis to estimate the efficiency of model based on LSTM with the GloVe embedding versus traditional machine-learning frameworks and their combination techniques such as bagging. The analysis of the user sentiment of the ratings of Google Apps was the central focus of the study and the university students were given special consideration. The data entailed enormous collection of Google App Reviews in the many languages and they also added a specialized data of Student App Reviews (SAR) that are indicators of the academic setting. The best of all the applied methods was the utilization of LSTM model with GloVe word embeddings whose accuracy and F1-score were 95.2 and 88 respectively. The other more traditional classification ones like SVM, Logistic Regression, and bagging also got 93-94% accuracy though deep learning approach was a slight better. Despite the excellent outcomes that the research has been carried out, the research has some limitations. It does not assess the model extrapolability to other applications and other areas, which is essential in the practical implementation. Also, the research does not entail a comparative performance in terms of transformer-based models i.e. BERT or RoBERTa which are the current state of the art in sentiment classification tasks.

The studies provided reveal that the models based on LSTM and its augmented versions, in particular, with pre-existing word embeddings (GloVe in particular) are always capable of high performance when it comes to sentiment analysis of mobile app reviews and have proven to be accurate at 82-95 per cent. Further, the use of CNN+LSTM allows to provide even better that performance due to the combination of the advantage of spatial and time features. These models are extremely strong in dealing with discordant user-generated written language, and they can learn contextual and sequential dependencies, which are decisive in interpreter feelings since the stuff is written in comment regarding the app.

The article by Basri, Nur Faizal, and Ema Utami. [61], the article is called the Application of Word2Vec and LSTM Models in Sentiment Analysis of Mobile Legends User Reviews and was dedicated to the usage of deep learning to review user reviews about the game Mobile Legends. The problem that the research addresses is the possibility of the deep learning models

to improve sentiment classes by analyzing the reviews of users of one of the most popular mobile games in Indonesia.

A mixture of Word2Vec embeddings and Long Short-Term memory (LSTM) networks constituted the methodology aspect of the research. Word2Vec was used to translate words into vectors representation in order to elicit the semantic meaning of the words. These embeddings were then fed into an LSTM model which predicts the extent to which a review is positive or negative. The model was trained with the help of supervised learning, but the reviews were already marked with the sentiment. A dataset of 100,000 reviewers of the Mobile Legends game was used in the research, but it was also collected in the Google Play store. They were primarily reviewed in the bahasa Indonesia language that most of the user base of the game would comprehend. The high amount of data provided a range of sentiments because it would be reasonable to use it in training the model.

These results were good since Word2Vec-LSTM model achieved a sentiment classification of 87.88%. The analysis has revealed that the majority of user review were favorable with words like good, exciting, and awesome used extensively in the reviews hence indicating that most of the users responded to the game well. However, the research had extremely critical loopholes. The Word2Vec-LSTM model performance was good but not compared to BERT or RoBERTa that is demonstrated to be doing better in the sentiment analysis task. Furthermore, the study was not aimed at the effects of the different preprocessing procedures and the processing of noisy data that might have caused an effect on the model performance and its generalization. Though all these studies possess their strengths, there is still a certain number of limitations in common. Specifically, the primary focus on accuracy as the main performance measure and the lack of the results of precision, recall and F1-scores as the ones required in imbalanced data sets is excessive. In addition, some of the studies use the transformer-based models such as BERT which have shown a state of art in NLP tasks to design a useful performance benchmark. Another gap that is required to identify sentiments that are associated with particular properties or any features of an app is the lack of fine-grained or aspect-level sentiment analysis. Besides, the majority of articles analyzed apply one-language and one-domain data and are not applicable to other language and contextual settings.

Lastly, the comparison reveals that as much as deep learning, and its LSTM/hybrid versions, is able to consistently exhibit and effectively classify sentiment in mobile review applications, with the use of deep learning, there can still be a significant improvement in understanding and improving deep learning-based systems. The additional researches can be developed with the

inclusion of transformer schemes, polylingual datasets as well as aspectual level sentiment data which will be scaled and aspectual featured inferences on various app ecosystems.

3.3 Large Language Models

Due to a constant progress of Natural Language Processing, the sentiment analysis has been transformed by deep contextualizing language in the shape of Large Language Models (LLMs) like BERT, RoBERTa, and DeBERTa. In comparison with the standard machine learning or even to pre-trained deep learning model, LLMs that have been trained on massive collections of data have better generalisation, and consequently are quite practical in user-generated tasks, such as written reviews written on an application. The next section will address the recent studies that apply LLMs to the area of sentiment analysis particularly in the mobile or the review-based nature. Every study is researched in terms of the methodology, data source, conclusion, and gaps in the research studies.

The objectives of the article by Eser and Sahin [62] were to compare the capabilities of various transformer-based methods of reviewing user reviews of the Spotify application. The experimented models including the BERT, DistilBERT, RoBERTa and XLM-RoBERTa were produced and trained on the classification of the reviews into positive, neutral or negative sentiment categories. The researchers pre-treated the data with lengthy procedures such as deemojifying, making corrections to the errors and tokenization and trained the models. Further, they retrieved the textual sentiment scores through VADER sentimenticity analysis and gave a corresponding perception in user rating categories and trends. The reviews of Spotify app in Google play store were included in the database. DistilBERT was the most accurate and recalls 71.68, whereas XLM-RoBERTa acquires the largest F1-score 69.24, therefore, it is a good balance between accuracy and recall. However, the research experienced some shortcomings: the overall performance of the research turning out to be quite poor as compared to the state of the art standards, moreover, models could not work with the neutral classes and the underrepresented classes. In addition, the experiment failed to investigate multilingual input and real-time implementation of models, which is necessary when it comes to implementation on realistic and dynamic applications.

Motger et al., [63] suggested T-FREX that was a transformer-based approach with a goal of extracting features of mobile app reviews with encoder only transformer such as BERT. As opposed to the all-word syntactic rule as in classical rule-based systems, T-FREX shapes the task into a token-based classification that leaves the model to discover app features in unstructured user reviews. The method includes pre-training on large corpus, and instance

selection ought to be performed to find a compromise between instance fine-tuning cost and recall and accuracy. It was a crowd sourced and industrial scale data set on ground truth according to correct annotations referring to actual review of mobile applications in Google Play Store. In contrast to the use of the baseline methods, T-FREX was found to be improving feature-level accuracy and recalls and, consequently, it provided compositional gains in the entire series of downstream predictions of sentiment analysis. However, the actual feature sentiment classification or polarity extraction of the extracted features was not dwelled upon in the study though. Therefore, it does not analyze the sentiment aspect per se, or talk about the prospect of enforcing fines-grained sentiment labeling per-feature of a per-feature basis during the feature extraction stage, although it does extend the preprocessing and feature extraction stage with LLMs.

According to Nkhata et al. [64], they proposed the hybrid model which incorporates BERT as a layer that runs with the bidirectional LSTM (BiLSTM) layer with an aim of enhancing the binary or fine-grained sentiment classification tasks. It was inspired by the incentive to utilize excellent understanding of content BERT offered and sequential dependencies in the forward and reverse directions offered by BiLSTM. To address the problem of the class imbalance, the study employed data augmentation techniques such as SMOTE and NLPAUG. The model was trained and tested using two benchmark data sets namely IMDb and SST-5 datasets that evaluate a binary sentiment classification and a five-level sentiment category respectively. It achieved an impressive score of 97.67 percent accuracy on IMDb (an increase of 0.27 percent over the state of the art) and 59.48 percent on SST-5 (an increase of 3.6 per cent. over a typical BERT base). However, the big disadvantage of the article is that it addresses the domain. The model was not implemented on a set of data on the review of mobile apps that more frequently includes other types of noise and user language. Besides, the work also failed to explore the sentiment tagging of neutral or ambivalence reviews, which are common in mobile applications. The model however, would have a high likelihood of being put into cross-domain application primarily on sentiment-filled platforms like app stores.

Chi et al. [65] also enhanced the BERT embeddings and added it with Temporal Convolutional Networks (TCN), as well as, Bidirectional LSTM (BiLSTM) and Self-Attention to formulate a sophisticated sentiment analysis model, which gives a rich text semantic meaning of the text data. This hybrid design was elaborated to ease into integration of the contextual recentness of BERT, time sequence acquisition of TCN, aware of sequence by BiLSTM, and ability to focus and enrich the self-attention, with the goal of enhancing sensitivity of the categorization of sentiments. The model was applied to the dataset of human-rated hotel reviews collected at a

variety of resources, including TripAdvisor, and Booking.com. The results of the models showed that this hybrid model worked as predicted and always outperformed baseline BERT models, in accuracy and F1-score in particular, demonstrating the hypothesis of the usefulness of the combination of deep contextual and sequential learning layers. However, the study was limited to the research in the hospitality industry and its application in other types of user-generated reviews, such as feedback in mobile application cases, was not explored. In addition, the efficiency with which the model could be applied in infringement was not examined in any part of the research and neither was the aspect-level sentiment analysis that is critical not only in real life but also in making insights available to the user.

The other hybrid framework that has been developed is the Instruct-DeBERTa by Jayakody et al., [66] framework which is designed to perform Aspect-Based Sentiment Analysis (ABSA), and it has the aspect term extraction (ATE) and aspect-level sentiment classification (ASC). It is fine-tuned by supervised learning and the strategy pays the DeBERTa-V3 strategy, which becomes task-specific and applied to various tasks of ABSA. This approach was trained and obtained validation on multilingual and multi-domain SemEval datasets (with regard to reviews in restaurants and laptops). The model demonstrated better results in aiming at a particular aspect (i.e., opinion targets) and categorising sentiment polarity concerning the targets of these among the available ABSA models with the heart of which is the BERT family. These results illustrate how DeBERTa has a strong potential with regards to its disentangled attention structure and convened positional encodings in fine-grained sentiment detection. However, the framework was experimented with mobile app review datasets that rely on domain specific language, non-scholarly language, and ambiguous intentions of people. Thus, it has not been experimented yet whether Instruct-DeBERTa can deal with the realistic app review scenarios, and it is this potential limitation of this model to the real life application.

Lee et al. [67] implemented the usage of sentiment analysis in user reviews with the help of the most suitable Korean specific language models of KoBERT, KoGPT-2, and KoBART. Their goal of the study was chiefly to gauge the emotions expressed in the user reviews particularly those available in Korean language. The models, the authors trained the hyperparameters, ensure those can work more effectively in sentiment classification and repeat the typical procedures of text preprocessing by carrying out text tokenization and by excluding stop words. These filtering operations were also vital in getting mutilated text ready to the models.

The information which was used in the study has been user reviews maintained on various platforms, though the total number of reviews that were reviewed was not stated. The research

demonstrated the usefulness of Korean specific model terms in sentiment Analysis by the use of KoBERT and KoGPT-2 and KoBART. The labeled optimized models successfully classified the sentiment of the user reviews and this fact supports the fact that the optimized models have the potential of sentiment-based analysis when used in the Korean language.

There were however gaps to the research. Though it focused on Korean language models, it did not study the application of multilingual models and models that have been trained on another language that would have enhanced the model to represent global application. Additionally, the performance of these Korean-specific models was not compared to other popular powerful models (e.g. BERT or RoBERTa), which may have also led to making more informative comparisons of their performance and functionality in terms of their cross-language and cross-use applications.

In their article called BERT Fine-Tuning for Sentiment Analysis on Indonesian Mobile App Reviews the researchers Nugroho et al. [68] had the target to find out whether fine-tuning bert-based sentiment analysis tasks can be useful in the context of linguistic factors, that is, the users of mobile applications in the countries of Indonesia. The research was geared towards modifying the affective assessment of the user reference of the Google Play Store. The authors employed two BERT pre-trained models namely a multilingual model and a pre-trained model based on the Southeast Asian text in Indonesian. These models were adjusted using a set of Indonesian user remarks on the 10 best mobile applications of 2020. The fine-tuning has involved the use of two labeling strategies, i.e., score-based and lexicon-based, the hyperparameter tuning has been used to fine-tune the performance of the models.

The study dataset was the set of user reviews of the Google Play Store of the 10 most popular mobile applications in Indonesia in 2020. The reviews were written in Indonesian (Bahaka), and thus reflected the language of the primary user group, that is why this dataset is highly applicable to the testing of the language-specific fine-tuning of BERT. They noted that the fine-tuned Indonesian-specific BERT model achieved the accuracy of 84 per cent when lexicon-based labeling was applied to the model training than when training multilingual BERT or classical machine learning model. The result revealed that domain specific pre-training is helpful in case of sentiment analysis, particularly in situations where we require to learn more about the Indonesian language especially in the case of user reviews.

Nevertheless, some gaps existed in the study. Even though, it was directed towards reviews by the Indonesian users, it failed to address how it can implement multilingual sentiment analysis, which will broaden the scope of the model to be forecasted in other languages. The study did

not also involve any comparison of the BERT model to other developed models such as RoBERTa or DeBERTa, which had been established to do well at sentiment analysis tasks as well. Besides, the issue of processing unreliable or informal language, which is widespread in user-generated content and may disrupt the efficiency of the sentiment classification when applied to a real-life scenario was overlooked in the paper.

Big Language Models (LLMs) such as BERT, RoBERTa, DeBERTa, and various others have been found to command acceptable assertions in their sentiment categorization and aspect-based sentiment classification in other areas of reviews. The accuracies reported change between approximately 71 and 97 percent relying on the complexity of data sets and levels of classification tasks involved (e.g. binary or multi-class sentiment). The suggested models have shining pros: they operate on the potent contextual representations, they have pre-trained embeddings, which are able to transfer across the tasks, and they are also sensitive to the learning of the fine-grained opinion of a particular aspect of the user-generated text. However, it still has several weaknesses. These models have not been used or even tested on mobile app review data, which has a unique linguistic and domain-specific problem. The models at other times are only focused on feature extraction and not necessarily on the sentiment polarity classification hence their use in giving sentiment insights are minimal. In addition, it can be said that multilinguality exists, and an ability to use it may not be utilized or practiced in certain uses. Not many studies have succeeded in developing aspect-level sentiment extraction to a dense end-to-end pipeline to accommodate app feedback on the phone. The same indication suggests that the foundation developed by LLMs is strong, and there are evident prospects to grow towards mobile-specific, multilingual, and aspect-sensitive sentiment systems.

In conclusion, it can be affirmed that multiple models that can investigate the sentiment analysis of the Mobile Legends user reviews have proved to be incredibly beneficial to make use of not only traditional Machine Learning (ML) but also more advanced Deep Learning and Large Language Models (DL/LLM). It has been shown that such features extraction methods as TF-IDF and Word2Vec are successful in sentiment classification where research on classical ML algorithms (such as Support Vector Machines(SVM) and Naive Bayes) further demonstrated the success of such features extraction methods. In spite of their efficiency as they do, these methods are usually unable to entirely represent the complexity and subtlety of informal user-generated content, such as slang and emotionally colored style.

Sentiment analysis has been better applied to Deep Learning (DL) models based on Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs) capable of detecting long range dependencies and local features of the text. Their application in sentiment

classification was enhanced by combination of the models with the existing pre-trained embeddings like Glove and Word2Vec. However, the DL models were yet to overcome the issue of working with noisy data, and the expense of training it on large volumes of data.

With the emergence of Large Language Models (LLMs) and a specific one, BERT and all of its variants, including BERT-like RoBERTa, KoBERT, and others, sentiment analysis is taking a different dimension, as pre-trained models able to perform more rigid, context-sensitive tasks emerge. LLMs are more effective than conventional ML and DL models in a wide range of studies, and they are very accurate and can operate with cross-linguistic data, fine-grained language context, and colloquial language. These models also exploit the effect of transfer learning which pre optimization can dramatically boost the performance of a particular task even when there is no huge data specific to the task.

All in all, the traditional models have a solid reliable foundation, although the deep learning models/LLMs have their obvious advantage, particularly in case of using huge and complicated data sets, i.e., the one supplied by the current Mobile Legends user reviews. Future studies can continue to design these sophisticated models with certain domain-specific enhancements, including the deployment of such models in real-time, in multiple languages, and the mechanism of identifying sarcasm. As the deep learning and LLM continue to be advanced, it will become a leader in gaming industry and other versions of sentiment analysis.

Table 3.1: Literature review summary table

	Study	Model	Dataset	Key Findings	Limitations / Gaps
Machine Learning Models	Jannah et al. [47]	SVM, Naive Bayes	Synthetic Shopee beauty product reviews	SVM achieved 86.14% accuracy; Naive Bayes 85.53%; TF-IDF + lemmatization effective	Synthetic dataset limits generalizability; no error analysis or feature importance; neutral class not fully addressed
	Juliane et al. [48]	Naive Bayes	Twitter comments	High training accuracy (93.33%), validation 86.66%; simple preprocessing effective	Small dataset; only NB tested; no comparison with other ML models
	Azeraf et al. [49]	Neural Naive Bayes (NNB)	IMDb movie reviews	Reduced error 4.5x vs NB; better on context-dependent and long reviews	Only binary sentiment; no comparison with SVM/LR; single dataset limits generalization
	Fashion e-commerce study [50]	SVM + SMOTE-Tomek	Fashion product reviews	Accuracy 92%, F1 0.89 with oversampling; baseline accuracy 68%	Binary classification only; neutral sentiment not studied; no

					comparison under same oversampling plan
	Wirawan et al. [51]	SVM + EDA	Instagram cyberbullying comments	Accuracy improved to 92% (+2.5% from baseline SVM); domain-specific improvement	No comparison with LR or Decision Tree baselines
	Özmen & Gündüz [52]	SVM, Logistic Regression, Decision Tree, Random Forest	Turkish e-commerce reviews	SVM performed best; LR decent across categories	No hyperparameter tuning; no interpretability analysis
	Danyal et al. [53]	Linear SVM, Multinomial NB, Logistic Regression, Passive Aggressive	IMDb 50K, Sentiment Polarity Dataset	LR slightly better (90.27%) than SVM (89.48%); simple linear models still effective	Only accuracy reported; no aspect-level sentiment; no multilingual or domain-specific testing
	Khan et al. [54]	SVM, Random Forest	News headlines, blog articles, social media posts	SVM slightly better (80.39%) vs RF (78.56%); RF more robust to noise	Only binary classification; no segment-level polarity; only TF-IDF features used
	Singhal et al. [55]	SVM, Naive Bayes, Logistic Regression, Random Forest (with BERT embeddings)	Amazon product reviews (fake vs genuine)	SVM best (87.81%); BERT embeddings improved semantic understanding	Synthetic/approximate data; no hyperparameter tuning; no error analysis or feature importance
	Ahammad et al. [56]	CNN, LSTM, BiLSTM, GRU, BiGRU, CNN+LSTM	122,231 AI app reviews (Google Play)	CNN+LSTM: 92.1%, LSTM: 91.97%; hybrid CNN+LSTM performed best; deep models learn contextual patterns well	Focuses on accuracy only; no per-class metrics; no transformer comparison; not lightweight for mobile
	Samanmali & Rupasingha [57]	ANN, LSTM, SVM	~33,000 reviews from 15 popular apps	LSTM: 80–90%; LSTM best across apps; highest F1-score	No hyperparameter tuning; no embedding/contextual features; long reviews not analyzed

Deep Learning Models	Amirkhalili & Wong [58]	LSTM, Naive Bayes	2,025 mobile banking reviews (Canada)	LSTM: 82%, NB: 77%; LSTM captures subtle sentiment patterns better	Small dataset; region-specific; no per-class metrics; no transformer benchmark
	Zaman et al. [59]	Neural networks (type unspecified)	XR app reviews (Google Play, 2025)	Not specified; deep learning better than ML baselines	Model type unclear; no detailed metrics; no SOTA comparison
	Ranjan & Mishra [60]	LSTM + GloVe embeddings	Google Apps reviews + Student App Reviews (SAR)	Accuracy: 95.2%, F1-score: 88; LSTM + embeddings outperforms traditional ML models	No transformer comparison; generalization to other apps/domains not evaluated
	Basri et al. [61]	Word2Vec + LSTM	100,000 Mobile Legends reviews (Bahasa Indonesia)	87.88%; most reviews positive; model captures semantic meaning via embeddings	No BERT/RoBERTa comparison; impact of preprocessing/noisy data not explored
Large Language Models	Eser & Şahin [62]	BERT, DistilBERT, RoBERTa, XLM-RoBERTa; VADER for sentiment score	Spotify app reviews from Google Play Store	DistilBERT: highest accuracy & recall (71.68%); XLM-RoBERTa: highest F1-score (69.24%)	Poor performance on neutral/underrepresented classes; no multilingual or real-time deployment
	Motger et al. [63]	T-FREX (token-level classification with BERT encoder)	Crowdsourced industrial dataset from Google Play Store	Improved feature-level precision and recall compared to baselines	Does not classify sentiment on extracted features; no aspect-level sentiment
	Nkhata et al. [64]	Hybrid: BERT + BiLSTM; data augmentation (SMOTE, NLPAUG)	IMDb (binary), SST-5 (five-level) datasets	IMDb: 97.67% accuracy; SST-5: 59.48% accuracy; better than standard BERT	Focused on movies, not app reviews; neutral/ambivalent sentiment not handled
	Chi et al. [65]	BERT embeddings + TCN + BiLSTM + Self-Attention	Hotel reviews from TripAdvisor, Booking.com	Outperformed baseline BERT in accuracy & F1; rich semantic representation	Limited to hospitality sector; no aspect-level sentiment or real-time deployment

	Ullah, W et al. [66]	Instruct-DeBERTa fine-tuned on SemEval datasets	Restaurant & laptop reviews (multilingual, multi-domain)	High performance in aspect term extraction & sentiment classification	Not tested on mobile app reviews; informal language handling unclear
	Lee et al. [67]	KoBERT, KoGPT-2, KoBART	User reviews in Korean	Effective Korean-specific models; better sentiment classification	No multilingual testing; no comparison with general LLMs (BERT, RoBERTa)
	Nugroho et al. [68]	Fine-tuned BERT (multilingual + Indonesian) with score-based and lexicon-based labeling	Google Play Store reviews of top 10 Indonesian apps	Indonesian BERT: 84% accuracy (lexicon-based)	No multilingual expansion; comparison with RoBERTa/DeBERTa missing; noisy/informal text not fully addressed

Chapter 4

Proposed Approach

4.1 Introduction

The approaches that are suggested to achieve the aims of this research study are presented in this chapter. The purpose of this paper is to compare and contrast the results of classical Machine Learning (ML) classifier with a recent pre-trained Transformer-based Large Language Model (LLM) with user-generated reviews of mobile games. To fulfil this objective a frameworked approach to methodology was constructed. It includes dataset retrieval, large-scale text processing, exploratory visual analytics, feature generation and selection process, model building, also the fine-tuning and hyperparameter optimization of LLMs done through gradient descent.

Research objective There is a trade-off between the classification objective, which can be more than other LLM-based solutions objectively better in both accuracy and robustness than traditional ML classifiers in processing real-world noisy Google Play Store reviews [69]. Mobile gaming reviews can be informal, multilingual and highly versatile in the styles and way they have to be expressed can give insightful content on the performance of generalization and on the manner MC and LLM models will be applicable to our problem statement.

Moreover, each model undergoes all the methodological protocols of data cleaning preprocessing to terminal model training with scientific rigour, reproducibility as well as transparency. Overall, this chapter provides complete information about the method suggested, the foundation of experimental findings in the following chapters.

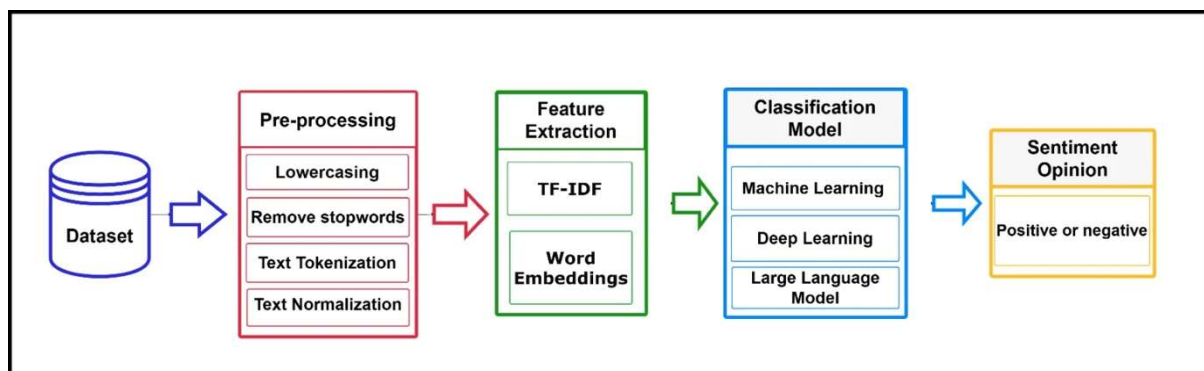


Figure 4.1: Model Training Pipeline of Proposed Methodology

4.2 Dataset Information

This study uses the Mobile Legends Google Play Reviews dataset [69]. The dataset contains 52,000 user reviews scraped from the Google Play Store. It has six columns, most of these columns are review text, rating, user id and date. The time range spans:

Start: February 16, 2025

End: June 10, 2025

Google Play Store reviews offer rich insights into user sentiment, perceptions of game balance and updates, user frustrations e.g., lag, bugs, monetization concerns, satisfaction drivers e.g., graphics, gameplay, rewards.

4.3 Preprocessing

A rigorous preprocessing pipeline was applied to convert raw user reviews into model-ready numerical representations. Text cleaning was performed to eliminate noise and standardize the user-generated text before any modelling. First, we lowered cased all text to eliminate case-based inconsistencies after that URLs, emojis, special characters and numerals were deleted using regular expressions. The cleaned text was tokenized with the function `word_tokenize` from NLTK to divide sentences into their tokens. Most frequent stop words such as “the,” and “is” were eliminated by the NLTK stopword corpus to exclude non-informative terms. After that duplicate words and missing values were handled and punctuation marks and special symbols were deleted. Lastly, we lemmatized (WordNetLemmatizer) to bring words down to their base form e.g., “running” → “run” providing a more regular and meaningful representation for downstream machine-learning models. Sentiment labels as shown in Table 4.1 were generated based on Google Play Store rating logic.

Table 4.1: Class Label distribution based on ratings

Rating	Class Label
1-3 stars →	Negative
4-5 stars →	Positive

Thus, a new column label was added with categorical sentiment values. For ML pipelines, the notebook explicitly created as text → cleaned user review and label → sentiment category. These columns were used in all models SVM, Logistic Regression, RF and LLM

preprocessing. A word cloud was generated for frequently occurring cleaned tokens to visually display dominant themes in user reviews shown in Figure 4.2.



Figure 4.2: Word Cloud of Most Frequent Terms in Mobile Legends Reviews

A distribution analysis was performed using a Bar chart shown in Figure 4.3 to show the count of positive, neutral, and negative reviews.

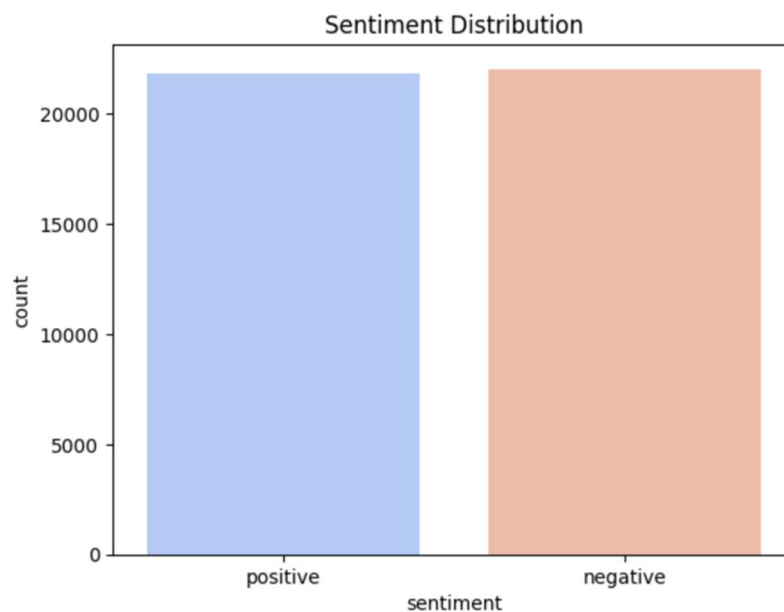


Figure 4.3: Bar Chart of Sentiment Category Distribution

Text was transformed into numeric features for machine-learning models using TF-IDF vectorization, which scored terms based on frequency in individual documents and rarity across the corpus. By choosing parameters such as `max_features` to cap the number of terms in the vocabulary, `ngram_range = (1,2)` to take unigrams and bigrams into account, `min_df = 5` to ignore terms that appear more rarely and gain a compact representation less sensitive to random noise. Lasso beta regression achieved both these objectives with good trade of reduces up to context. Grid Search was run using a TF-IDF + Logistic Regression to fine tune the modeling pipeline, with a systematic scan across combinations of vocabulary sizes (1000 features/words up to 3000) and regularization strengths $C = \{0.1, 1, 10\}$. By 3-cross-validation, GridSearchCV found values of hyperparameters that provide the best accuracy, leading to an experimentally verified and well-tuned configuration of feature-model for sentiment classification.

4.4 Machine Learning Models

In this section, we briefly describe all the machine-learning and deep-learning models that are trained in our experiments while focusing only on how they function and relate to TF-IDF or embedding-based sentiment classification.

4.4.1 Support Vector Machine (LinearSVC)

LinearSVC selects an optimal separating hyperplane to maximize the margin between positive and negative classes suiting for high-dimensional TF-IDF vectors.

Table 4.2: SVM model parameter

Parameter / Metric	Value
Vectorizer	TfidfVectorizer() (default parameters)
Train/Test Split	80% Train / 20% Test
Grid Search Parameters	C: [0.01, 0.1, 1, 10, 100] loss: ['hinge', 'squared_hinge']
Best Parameters (GridSearchCV)	C = 0.1, loss = 'squared_hinge'
Model Used	LinearSVC()

4.4.2 Logistic Regression

With its sigmoid / softmax function parameterization, Logistic Regression model can estimate class probabilities which is useful for TF-IDF features being linearly separable to distinguish classes.

Table 4.3: Logistic Regression model parameter

Parameter / Metric	Value
Vectorizer	TfidfVectorizer(max_features=10,000, ngram_range=(1, 2), stop_words='english')
Text Column	Cleaned
Label Column	Sentiment
Train/Test Split	80% Train / 20% Test (random_state=42)
Model	LogisticRegression()
C (Regularization Strength)	1.0
Penalty	l2
Solver	Liblinear
max_iter	2000

4.4.3 Random Forest Classifier

We used Random Forest Ensemble predictions across bootstrapped decision trees designed to decrease the high variance and improve robustness of noisy TF-IDF vectors.

Table 4.4: Random Forest model parameter

Parameter / Metric	Value
Vectorizer	TfidfVectorizer(max_features=15,000, ngram_range=(1,2), stop_words='english')
Text Column	Cleaned
Label Column	Sentiment
Train/Test Split	80% Train / 20% Test (random_state=42)

Model	RandomForestClassifier()
n_estimators	200
max_depth	None (fully grown trees)
min_samples_split	2
min_samples_leaf	1
random_state	42
n_jobs	-1 (uses all CPU cores)
Evaluation Metric	Test accuracy

4.4.4 LightGBM Classifier

LightGBM is another boosting algorithm based on gradient boosting that implements novel techniques such as GOSS and EFB, which can result in both fast training time but high accuracy on high-dimensional feature space like TF-IDF matrices.

Table 4.5: LightGBM model parameter

Parameter / Metric	Value
Vectorizer	TfidfVectorizer(max_features=15,000, ngram_range=(1,2), stop_words='english')
Text Column	Cleaned
Label Column	Sentiment
Train/Test Split	80% Train / 20% Test (random_state=42)
Model	LGBMClassifier()
n_estimators	500
learning_rate	0.05
max_depth	-1 (no depth limit)
num_leaves	64
subsample	0.8
colsample_bytree	0.8
objective	"binary" (since only two sentiment classes)
random_state	42
n_jobs	-1 (use all CPU cores)

4.4.5 Naïve Bayes Classifier

Multinomial Naïve Bayes calculates the probabilities of the class on term frequencies, and works well with sparse high-dimensional TF-IDF features. Its joint word with character-level n-grams preserve lexical and subword features shared across informal reviews.

Table 4.6: Naïve Bayes model parameter

Parameter / Component	Value
Vectorizer 1 (Word-level TF-IDF)	TfidfVectorizer(max_features=20,000, ngram_range=(1,2), stop_words='english')
Vectorizer 2 (Character-level TF-IDF)	TfidfVectorizer(analyzer='char', ngram_range=(3,5), max_features=20,000)
Merged Feature Space	hstack([word_TFIDF, char_TFIDF])
Text Column	Cleaned
Label Column	Sentiment
Train/Test Split	80% Train / 20% Test (random_state=42)
Model	MultinomialNB()
alpha (default)	1.0
fit_prior	True
Total Feature Dimensions	Word + Char TF-IDF combined $\approx$ up to 40,000 features

4.5 Large Language Model

4.5.1 DistilBERT-Base-Uncased

DistilBERT is a distilled version of BERT with ~97% of its effectiveness, smaller in size and faster training hence it can be applied to mid-size datasets. As the model is trained with small learning rates, dropout, label smoothing and early stopping, the framework excels in seizing contextual sentiment cues than traditional ML models.

Table 4.7: DistilBERT model parameter

Parameter / Component	Value
Base Model	distilbert-base-uncased
Task	Binary sentiment classification (positive / negative)

Input Text Column	Cleaned
Label Column	sentiment (after removing score = 3 and mapping $\leq 2 \rightarrow$ negative, $\geq 4 \rightarrow$ positive)
Train/Test Split	80% Train / 20% Test (seed=42)
Max Sequence Length	128 tokens
Tokenizer	AutoTokenizer.from_pretrained("distilbert-base-uncased") with padding = max_length, truncation = True
num_labels	2
Dropout	0.20
attention_dropout	0.20
label_smoothing	0.10
Training Framework	Hugging Face Trainer
learning_rate	1e-5
warmup_ratio	0.10
lr_scheduler_type	Cosine
num_train_epochs	3
per_device_train_batch_size	16
per_device_eval_batch_size	16
gradient_accumulation_steps	2 (effective batch size 32)
weight_decay	0.15
eval_strategy	"steps" (every 100 steps)
save_strategy / save_steps	"steps", every 300 steps
metric_for_best_model	eval_accuracy (higher is better)
Early Stopping	EarlyStoppingCallback(early_stopping_patience=2)
Mixed Precision	fp16 = True

4.5.2 DeBERTa-v3-Base

DeBERTa-v3 promotes contextual understanding via disentangled attention and augmented mask decoding, which results in higher-quality semantics. Larger dropout rates, label smoothing and weight decay were utilized to prevent overfitting during fine tuning.

Table 4.8: DeBERTa-v3 model parameter

Parameter / Component	Value
Base Model	microsoft/deberta-v3-base
Task	Binary sentiment classification (positive / negative)
Input Text Column	Cleaned
Label Column	sentiment (class-encoded)
Train/Test Split	80% Train / 20% Test (seed=42)
Max Sequence Length	256 tokens
Tokenizer	AutoTokenizer.from_pretrained("microsoft/deberta-v3-base") with truncation=True, padding="max_length"
num_labels	2
hidden_dropout_prob	0.30
attention_probs_dropout_prob	0.30
label_smoothing_factor	0.15
Training Framework	Hugging Face Trainer
learning_rate	1e-5
warmup_ratio	0.10
weight_decay	0.18
lr_scheduler_type	Cosine
num_train_epochs	7
per_device_train_batch_size	8
per_device_eval_batch_size	16
gradient_accumulation_steps	2 (effective batch size 16)
eval_strategy	"steps" , every 350 steps
save_strategy / save_steps	"steps", every 350 steps
logging_steps	80
metric_for_best_model	eval_accuracy (greater_is_better=True)
Early Stopping	EarlyStoppingCallback(early_stopping_patience=2)

Mixed Precision	fp16 = True
-----------------	-------------

4.5.3 RoBERTa-Base

RoBERTa further improves the BERT by discarding NSP, applying dynamic masking and training with optimized batch schedules for more powerful contextual encoding.

Table 4.9: RoBERTa model parameter

Parameter / Component	Value
Base Model	roberta-base
Task	Binary sentiment classification (positive / negative)
Input Text Column	Cleaned
Label Column	sentiment (class-encoded)
Train/Test Split	80% Train / 20% Test (seed=42)
Max Sequence Length	256 tokens
Tokenizer	AutoTokenizer.from_pretrained("roberta-base") with truncation=True, padding="max_length"
num_labels	2
hidden_dropout_prob	0.25
attention_probs_dropout_prob	0.25
label_smoothing_factor	0.12
gradient_checkpointing	True
Training Framework	Hugging Face Trainer
learning_rate	8e-6
warmup_ratio	0.10
lr_scheduler_type	Cosine
per_device_train_batch_size	16
per_device_eval_batch_size	16
gradient_accumulation_steps	2 (effective batch size 32)
num_train_epochs	8
weight_decay	0.15
eval_strategy	"steps" every 300 steps
save_strategy / save_steps	"steps", every 300 steps
logging_steps	60

metric_for_best_model	eval_accuracy (greater_is_better=True)
Early Stopping	EarlyStoppingCallback(early_stopping_patience=2)
Mixed Precision	fp16 = True

4.5.4 BERT-Base-Uncased

BERT utilizes bidirectional Transformer encoders pre-trained with MLM and NSP, which allows the model to effectively learn contextual representation.

Table 4.10: BERT-Base-Uncased model parameter

Component	Setting / Value
Base model	roberta-base
Task	Binary sentiment classification (positive / negative)
Input text column	Cleaned
Label column	sentiment (class-encoded)
Label rule	negative if score ≤ 3 , else positive
Train/test split	80% train / 20% test (seed=42)
Preprocessing	Lowercasing, emoji removal (regex), punctuation & special-char removal, whitespace normalization, NLTK stopwords removal, duplicate removal on content
Tokenizer	AutoTokenizer.from_pretrained("roberta-base")
Max sequence length	256 tokens (padding="max_length", truncation=True)
num_labels	2
hidden_dropout_prob	0.25
attention_probs_dropout_prob	0.25
label_smoothing_factor	0.12
Metric	Accuracy (evaluate.load("accuracy"))
Learning rate	8e-6
Warmup ratio	0.10
LR scheduler	Cosine
Weight decay	0.15
Epochs	8
Per-device train batch size	16
Per-device eval batch size	16

Gradient accumulation steps	2 (effective batch size 32)
Evaluation strategy	"steps" (every 300 steps)
Save strategy / steps	"steps", every 300 steps
Logging steps	60
Early stopping	EarlyStoppingCallback(early_stopping_patience=2)
Mixed precision	fp16=True
Best-model criterion	load_best_model_at_end=True, metric_for_best_model="eval_accuracy", greater_is_better=True
Output directory	"sentiment_roberta"

4.5.5 ELECTRA-Large-Discriminator

Replaced Token Detection allows ELECTRA to be trained more effectively than traditional masked language models and retains the ability to model the entire distribution of linguistic context for words without making independence assumptions between predictions.

Table 4.11: ELECTRA-Large Discriminator model parameter

Component	Setting / Value
model_name	"google/electra-base-discriminator"
num_labels	2
max_length	256
Padding	"max_length"
truncation	True
test_size	0.2
hidden_dropout_prob	0.30
attention_probs_dropout_prob	0.30
classifier_dropout	0.30
weight_decay	0.15
label_smoothing_factor	0.12
learning_rate	1.5e-5
warmup_ratio	0.15
lr_scheduler_type	"cosine"
num_train_epochs	8
per_device_train_batch_size	16

gradient_accumulation_steps	2
eval_strategy	"steps"
eval_steps	250
save_strategy	"steps"
save_steps	250
load_best_model_at_end	True
metric_for_best_model	"eval_accuracy"
greater_is_better	True
early_stopping_patience	2
output_dir	"sentiment_electra"
logging_steps	60
fp16	True

4.6 Deep Learning Models

4.6.1 Bi-LSTM

This model utilizes 100-dimensional GloVe embeddings and a bi-GRU layer to encode forward-backward context information from user reviews. It used a custom attention mechanism to focus on sentiment-aware tokens and applied dense layers in combination with dropout for stable classification.

Table 4.12: Bi-LSTM model parameter

Component	Setting/Value
MAX_WORDS	20000
MAX_LEN	120
test_size	0.2
random_state	42
Embedding Type	GloVe 6B
EMB_DIM	100
Trainable	False
Type	Bidirectional GRU
Units	64
Dropout	0.3

return_sequences	True
Type	Custom Layer
Dense Layer 1 units	64
Dense Layer 1 activation	"relu"
Dropout Layer rate	0.4
Output Layer units	1
Output Layer activation	"sigmoid"
Loss	"binary_crossentropy"
Optimizer	"adam"
Metrics	["accuracy"]
Epochs	12
batch_size	64
validation_split	0.2
Monitor	"val_loss"
Patience	2
restore_best_weights	True

4.6.2 CNN + BiGRU Hybrid Model

The hybrid model integrates multi-kernel 3,5,7 filters CNNs to capture local n-gram patterns and a BiGRU layer expressing long-range dependencies. After the global pooling, batch normalization and dropout are applied to stabilize learning along with the GloVe embeddings providing semantic classification.

Table 4.13: CNN + BiGRU Hybrid model parameter

Component	Setting/Value
MAX_WORDS	16000
MAX_LEN	120
oov_token	"<UNK>"
test_size	0.2
random_state	42
Embedding Type	GloVe 6B
EMB_DIM	100
trainable (Initial)	False

trainable (Second Stage)	True
SpatialDropout1D	0.25
Filter units	128 (x3)
Kernel size	3, 5, 7
activation	"relu"
Concatenate	N/A
MaxPooling1D size	2
Type	Bidirectional GRU
units	64
dropout	0.3
recurrent_dropout	0.3
return_sequences	True
Pooling Layers	GlobalMaxPooling1D, GlobalAveragePooling1D
Output Concatenate	N/A
Dense 1 units	128
Dense 2 units	64
kernel_regularizer	l2(1e-4)
BatchNormalization	N/A
Dropout 1 rate	0.5
Dropout 2 rate	0.35
Output Layer units	1
Output Layer activation	"sigmoid"
optimizer	AdamW
learning_rate	2e-4
weight_decay	1e-4
epochs	15
validation_split	0.2
learning_rate	1e-4
epochs	5
EarlyStopping patience	3
RestoreBestWeights	True
ReduceLROnPlateau patience	1
factor	0.4

min_lr	1e-6
--------	------

4.7 Summary

This chapter has detailed the entire process that was used in the sentiments recognition of reviews on Mobile Legends databank. Following a massive pre-processing of dataset, three modelling levels have been used, (1) classical machine learning baselines using TF-IDF includes SVM, Logistic Regression, Naive Bayes, Random Forest, LightGBM, (2) deep-learning models with semantic embeddings and sequence learning e.g. Bi-GRU with attention, CNN + BiGRU hybrid and (3) modern transformer-based LLMs DistilBERT, BERT-base, RoBERTa-base, DeBERTa-v3-base and ELECTRA-discrim. All the models had been tuned to sane values of hyperparameters and regularizations, and their performance was evaluated on held out test data in the accuracy on classification. Through this multi-level methodological structure, we also give solid empirical findings to compare the three different modes of architectures i.e., traditional, deep and transformer-based architecture which renders our findings and analysis powerful.

Chapter 5

Experimental Results and Discussion

5.1. Introduction

This chapter provides and explains experimental outcomes of machine learning (ML), deep learning (DL) and transformer-based models which are introduced in this paper. The experiments are conducted on large text dataset of 41957 samples and 8779 has testing samples. The ultimate objective was to evaluate and compare predictive performance of standard machine learning models, neural network structures as well as the most recent transformer-based models in the task of text classification of the target.

In all three model classes in which evaluation metrics (e.g. accuracy, precision, recall, and F1-score) are the same. The joint highlighting of performance trends, strengths, and limitations of all the three approaches are based on comparative tables and visualizations. This admits to the straightforward identification of the gains in performances made by more sophisticated modeling paradigms. The chapter finishes with the discussion of the findings in details, and then the analysis of the limitations of the proposed work is provided, as well as a brief overview of the most important insights.

5.2. Results of Individual Models

The model-wise performance is presented in terms of classifier accuracy with precision, recall and F1-score. These models are strong baselines and can be compared with state-of-the-art transformer-based architectures.

5.2.1. Traditional Machine Learning Models

Logistic Regression with Grid Search and TF-IDF data provided good results in the high dimensionality text datasets. The parameters Regularization parameter $C=1$ and a TF-IDF vocabulary size of 3,000 contributed to stable and balanced performance. The Support Vector Machine performed well. The best SVM model had an accuracy of 0.89, with both classes being equally represented by precision and recall. It was able to effectively address sparse representations of text by maximizing the margin between classes.

For LightGBM, the accuracy was 0.89 seems to perform well, however, it is slightly lower than that for linear models. Although LightGBM is good with structured data, its performance implies a weak use of context information from text features. Naïve Bayes was very simple but performed quite well (accuracy = 0.89), especially for negative class recall. But its strong independence assumption limited the overall power of classification.

The Random Forest algorithm achieved an accuracy of 0.88, because the ensemble learning it adopted but showing reduced effectiveness compared to linear classifiers due to the high dimensionality of TF-IDF features.

Table 5.1: Performance Metrics of ML Models

Model	Precision	Recall	F1-Score	Accuracy
Logistic Regression	0.89	0.89	0.89	0.89
SVM (Optimized)	0.90	0.89	0.89	0.89
LightGBM	0.89	0.89	0.89	0.89
Naïve Bayes	0.89	0.88	0.88	0.88
Random Forest	0.88	0.88	0.88	0.88

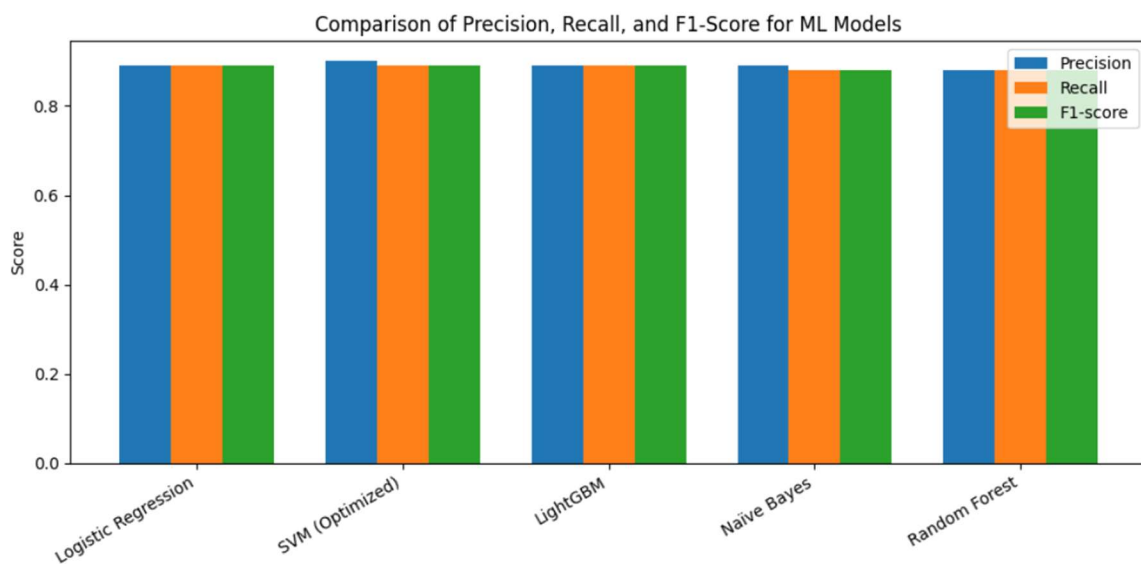


Figure 5.1 Comparison of precision, recall, and F1-score for machine learning models

The performance of the evaluated machine learning models in terms of precision, recall, and F1-score enables a better understanding of their classification capabilities beyond global accuracy. The focused SVM model obtained the most precision (0.90), This is an indication that SVM performs very well in separating class boundaries when used to model high-dimensional TF-IDF feature spaces.

5.2.2. Deep Learning Models

The BiLSTM model and the CNN + BiGRU had 0.8779 and 0.8743 accuracy, respectively shown in Table 5.2. Their accuracy and loss graphs are shown in Figure 5.2, and 5.3. The results show that the sequential neural architectures achieved better contextual understanding than the traditional ML models, yet they still underperformed in comparison to transformer-based models which can model long range dependencies.

Table 5.2: Performance results of Deep Learning Models

Model	Precision	Recall	F1-Score	Accuracy
Bi-LSTM	0.8798	0.8793	0.8780	0.8792
CNN + BiGRU Hybrid	0.8779	0.8779	0.8779	0.8778

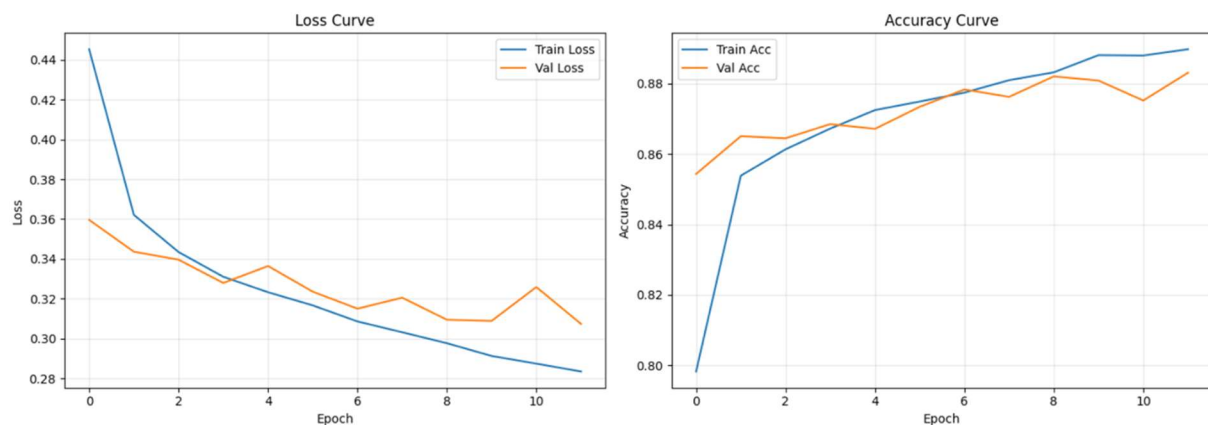


Figure 5.2. Training and validation loss and accuracy curves of the Bi-LSTM model.

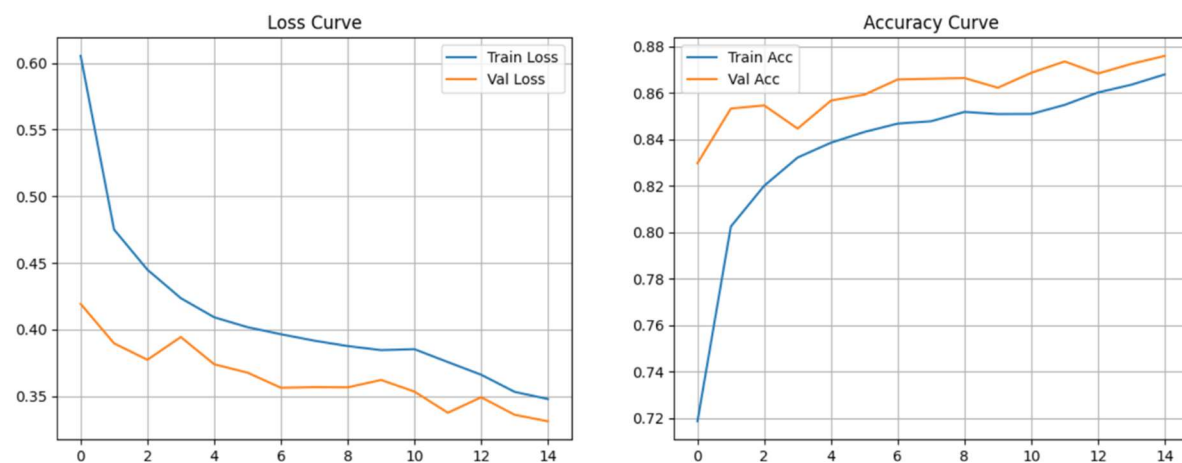


Figure 5.3. Training and validation accuracy and loss of the CNN-BiGRU model.

5.2.3. Results of Transformer-Based Models

Transformer-based models performed much better than classical ML and DL techniques because they could learn contextual semantics through self-attention mechanisms. The considered models are DistilBERT, RoBERTa, ELECTRA, and DeBERTa-v3. The performance results of LLMs are summarized in Table 5.3.

Table 5.3: Performance results of Transformer-Based Models

Model	Precision	Recall	F1-Score	Accuracy
Distilbert-Base-Uncased	0.9019	0.9014	0.9012	0.9013
DEBERTA-V3-BASE	0.9179	0.9164	0.9162	0.9164
Roberta-Base	0.9038	0.9038	0.9038	0.9037
Electra-Large-Discriminator	0.8990	0.8989	0.8988	0.8988

DistilBERT, a lightweight transformer model scored around 0.9013 accuracy, accuracy and loss curves are shown in Figure 5.4, surpassing all classical ML models with more computationally efficient. DEBERTA-V3 Figure 5.5 achieved the highest precision rate of 0.9164 due to robust pretraining and tokenization attack strategies optimization. ELECTRA obtained test accuracy of 0.8988 Figure 5.7, competitive with but slightly below RoBERTa in Figure 5.6 due to its discriminator-based training nature.

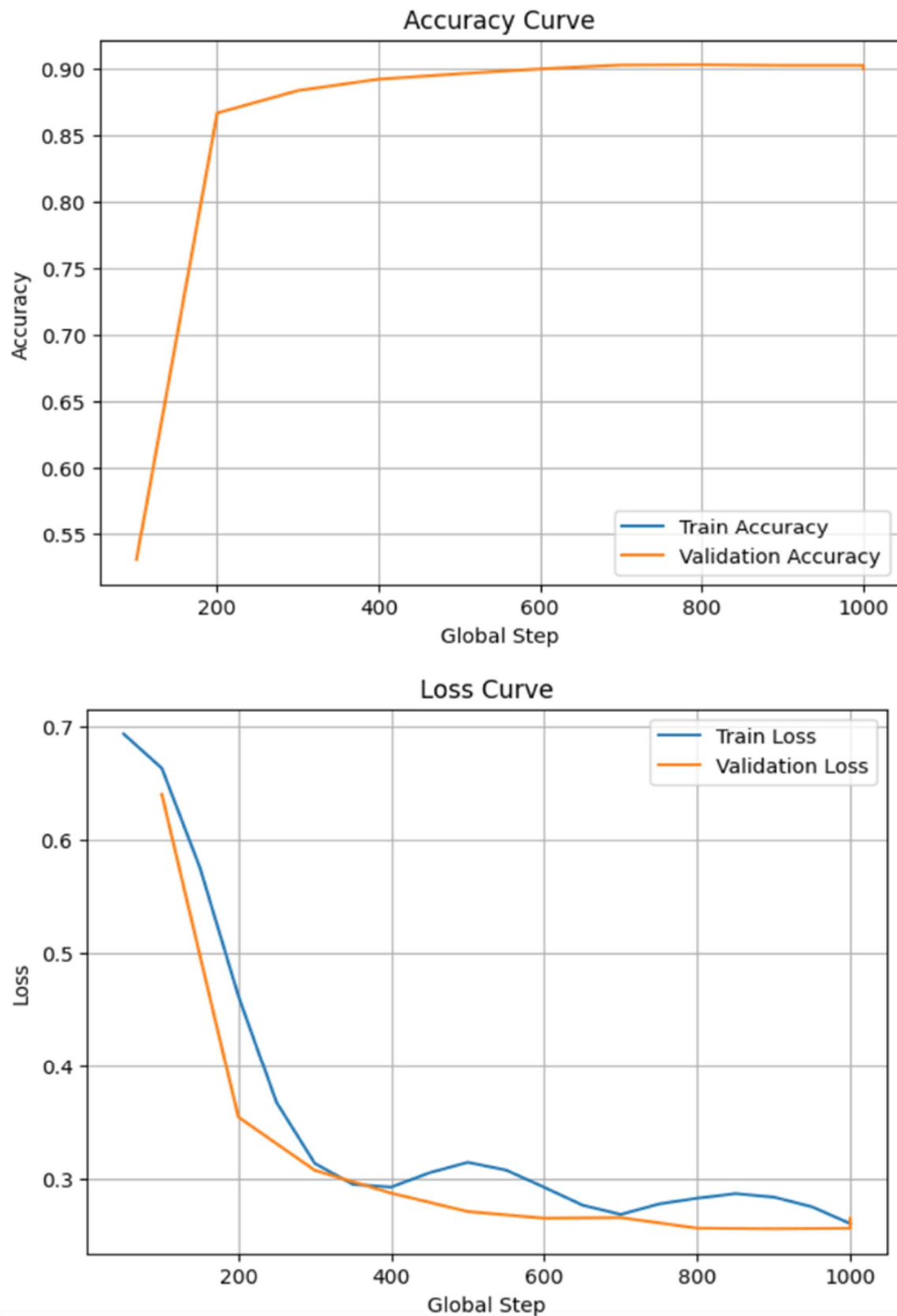


Figure 5.4. Training and validation accuracy and loss of the DistilBERT model.

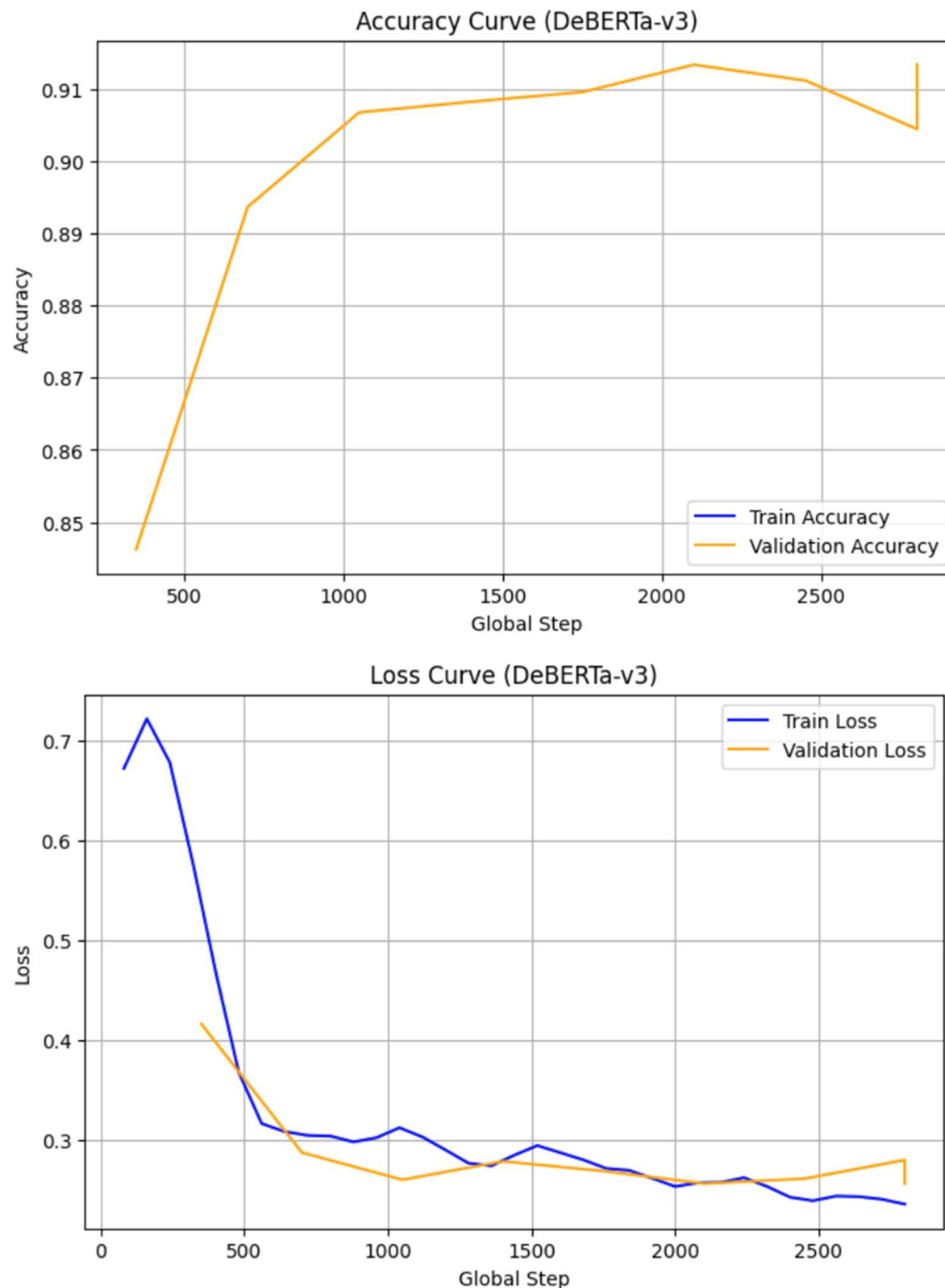


Figure 5.5. Training and validation accuracy and loss of the DeBERTa-v3 model.

The DeBERTa-v3 model reached a maximum test accuracy of 0.9164, which is the best performance obtained in this study shown in Figure 5.5. Its decoupled attention mechanism and strengthened context encoding remarkably increased classification performance.

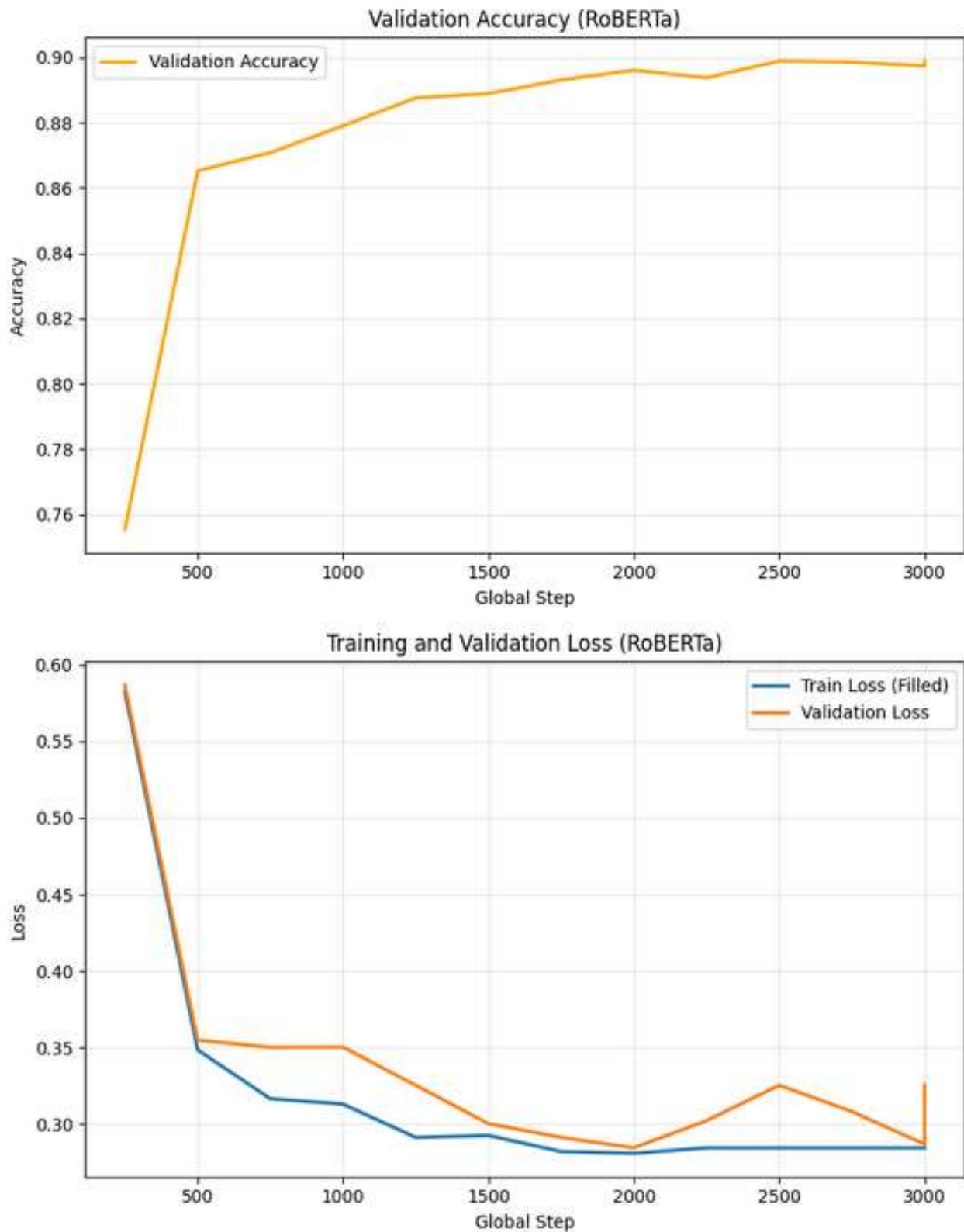


Figure 5.6. Training and validation accuracy and loss of the RoBERTa model.

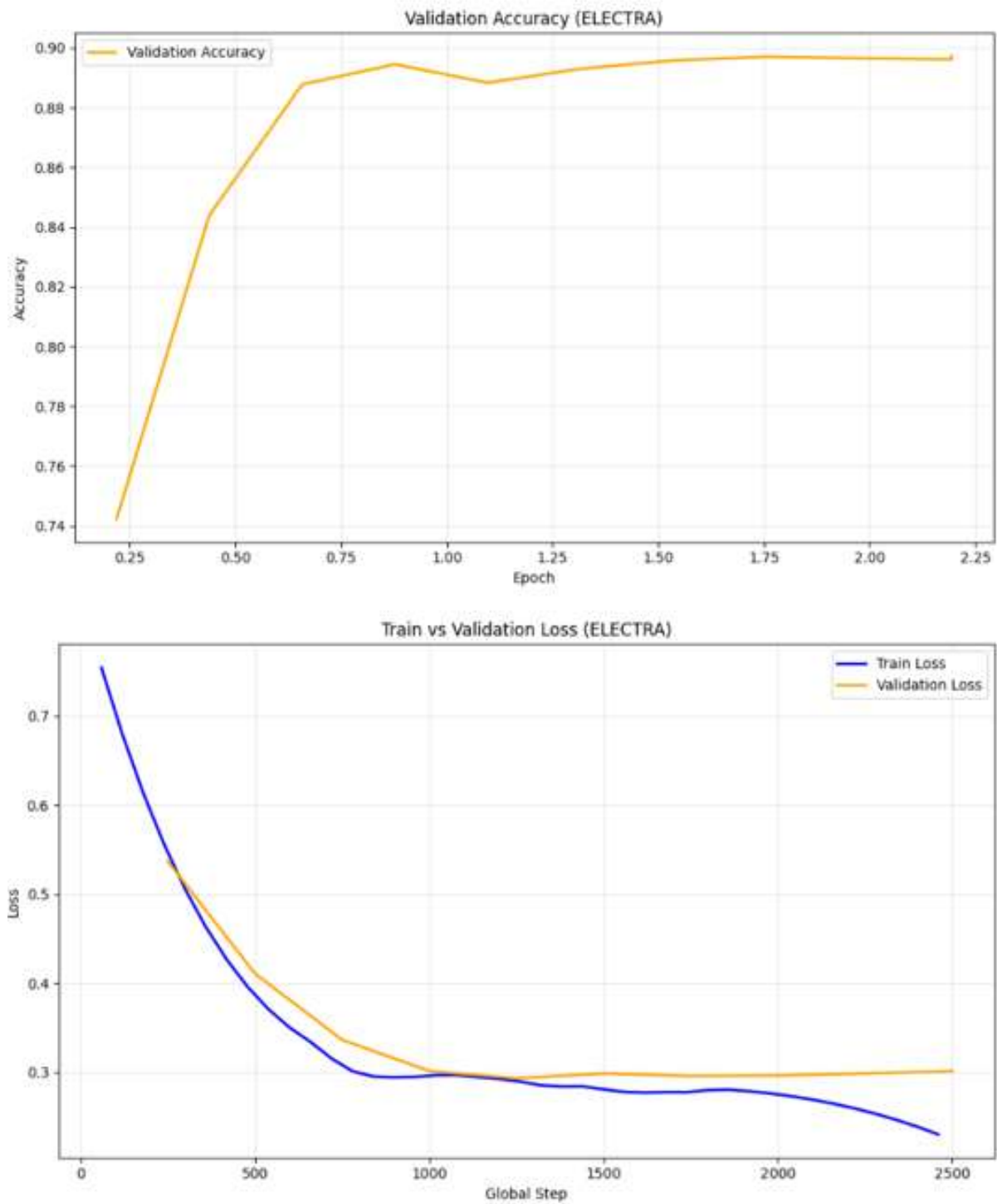


Figure 5.7. Training and validation accuracy and loss of the ELECTRA model.

5.3. Discussion of Findings

According to the experimental findings, transformer-based models are more effective indeed than traditional ML and DL models. Even though effective in managing the sparse TF-IDF features, linear models such as the logistic Regression and SVM are also it is not effective with deep contextual connections.

The semantic meaning of word sequences was inferred through CNN and Bi-LSTM-based models that were unable to fully capture the dependencies over a long distance but performed poorly as compared to transformers. The most successful model is DeBERTa-v3 whose accuracy = 0.9164 and training and validation Loss of DeBERTa-v3 is given in Figure 5.5. The outcome validates that disentangled attention and enhanced contextual embedding are useful in enhancing a superior answer prediction.

This suggests that the general performance outcome of all the models depicted in Figure 5.8 indicates that the use of pretrained language models is essential in achieving the high-performance of text classification attempts addressing complex scenarios especially when contextual detail in the model is a major factor of consideration.

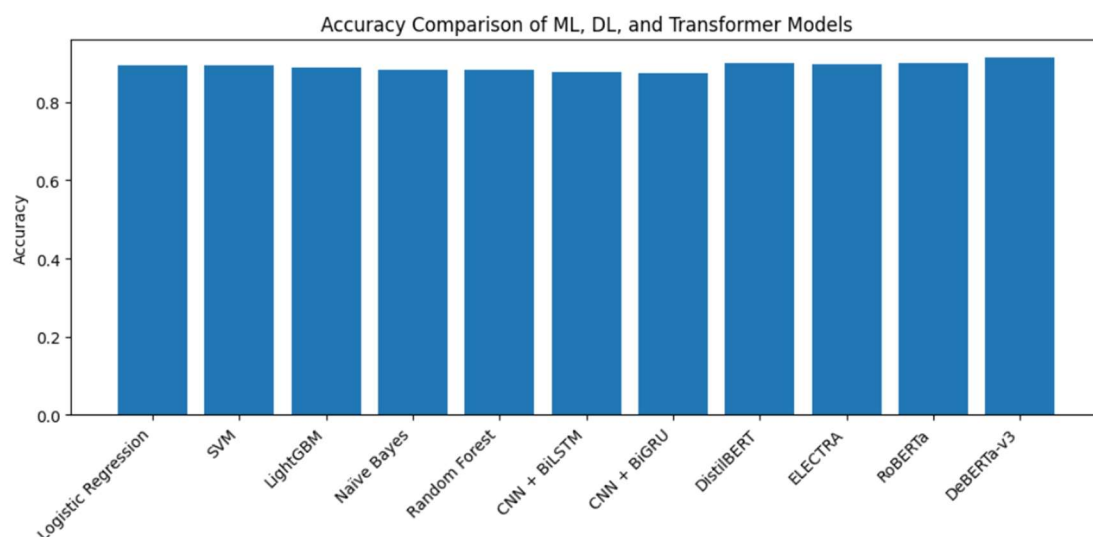


Figure 5.8. Accuracy comparison of machine learning, deep learning, and transformer-based models.

5.4. Limitations of the Proposed Work

In spite of good performance, there are still a number of limitations. To begin with, pre-trained transformer models are heavy in terms of the computation resources and cannot be applied in real-time. Secondly, the data does not vary with time, and therefore, it is impossible to use sequence-based temporal modeling. Thirdly, hyperparameter optimization of transformers is computationally expensive and hard to train. Finally, here we take into consideration only the supervised learning and even leave semi-supervised or unsupervised solutions to which can even enhance the generalization.

5.5. Summary of the Chapter

The chapter showed a detailed experiment of ML, DL and, LLM models in text classification. Every model was rated against a similar set of criteria in that way that afforded equal and proper comparison across the standard evaluation measurements. Baseline In standard ML models, LR and optimized SVM, high results of 89 percent accuracy were observed on high performance with high dimensional textual representations. Deep networks like Bi-LSTM and CNN-BiGRU also reached lower contextual learning and lower identification accuracy [?]88%, and thus they did not show much capability of learning long-range dependencies.

Transformers models far surpassed both the ML and DL models in terms of usage of their self-attention architectures and the provision of already trained contextual representations. They are in decreasing order: in the accuracy test, DeBERTa-v3 scored 91.64% and showed balanced classification of its results, as the analysis of the confusion matrices showed. The classical ML models namely SVM however, only confirmed the fact that they were able to perform well but transformers had better predictive abilities.

Overall, these findings are quite explicit that attention-based transformer models can better perform in the context of extracting semantics and context in text information. This proves the modeling approach that is proposed and adds to the implications of the findings and future work considered in the following chapter, particularly in optimizing the models and their scaling up to deploying them to the real world.

Chapter 6

Conclusion and Future Work

6.1. Conclusion

This thesis involved an assessment process to understand the performance of classical machine learning and deep learning structures together with the current transformer-based language models on sentiment categorization. Through the results of the experiments, it is seen that the contextual semantics enhances the performance of the model. Both Logistic Regression and Support Vectors showed an excel performance on TF-IDF representations, proving that the two can be used in high dimension and sparse textual applications. This was the case since they could not represent the deeper semantic relationships, as they simply stared at the surface. Deep learning models, CNN-BiGRU and Bi-LSTM, showed great improvements through the process of learning sequential and local text patterns. Despite having improved, they could not find long-range correlations, and thus, their accuracy was lower compared to the methods that were transformer-based. Transformer models can make informed guesses using self-attention processes of how the various sequences of components of an input sequence are related to each other. This is a plus in comparison with the traditional and the deep learning models. DeBERTa-v3 was the best compared to the other models as its accuracy is 0.9164. With this in mind, it is evident that disentangled attention and contextual encoding can be rather helpful. Based on the findings, to perform meaningful sentiment analysis, one has to apply transformer-based language models that have been pretrained to perform tasks which are, contextually, complicated. Based on the results of in depth study and significant volume of empirical research, transformer based architecture will be used in the case of sophisticated sentiment analysis solutions.

6.2. Future Work

This paper demonstrates that transformer-based models are more effective in classifying the sentiment, yet dozens of interesting fields of work can still be examined. Future studies can examine thorough sentiment analysis, such as aspect-based sentiment classification to capture more accurately the user perception on specific aspects of mobile apps. Moreover, it would be useful to add datasets that are bilingual and those that belong to other fields so that the model could be more general and applicable to as many different users as possible.

The further research can focus on the efficiency of models and model deployment with the concentration on lightweight or quantized transformers design to enable real-time sentiment analysis on resource-constrained systems. Custom further explainability methods such as attention visualization and post-hoc interpretation may help to make model predictions more

understandable and believable. Finally, a combination of sentiment analysis, topic modeling, and user behavior analysis can give us a clearer view of the current response of the users, which could allow us to make more informed choices regarding the way in which we can improve our mobile application and services.

References

- [1] Ballas, V., Michalakakis, K., Alexandridis, G., & Caridakis, G. (2024). Automating mobile app review user feedback with aspect-based sentiment analysis. *Lecture Notes in Computer Science*, 179–193.
- [2] Maroof, A., Wasi, S., Jami, S. I., & Siddiqui, M. S. (2024). Aspect-based sentiment analysis for the service industry. *IEEE Access*, 12, 109702–109713.
- [3] Adiningtyas, H., & Auliani, A. S. (2024). Sentiment analysis for mobile banking service quality measurement. *Procedia Computer Science*, 234, 40–50.
- [4] Sivakumar, M., & Uyyala, S. R. (2021). Aspect-based sentiment analysis of mobile phone reviews using LSTM and fuzzy logic. *International Journal of Data Science and Analytics*, 12(4).
- [5] Qahar, A., Ruldeviyani, Y., Mukharomah, U. N., Fidyawan, M. A., & Putra, R. (2024). Factor analysis influencing mobile JKN user experience using sentiment analysis. *IAES International Journal of Artificial Intelligence*, 13(2), 1782–1793.
- [6] Liu, B. (2020). *Sentiment analysis: Mining opinions, sentiments, and emotions* (2nd ed.). Cambridge University Press.
- [7] Thelwall, M., Buckley, K., & Paltoglou, G. (2012). Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1), 163–173.
- [8] Rana, A., & Singh, S. (2014). Comparative analysis of sentiment orientation using SVM and Naïve Bayes techniques. *International Journal of Computer Applications*, 99(16), 1–6.
- [9] Liu, T. Y., Yang, Y., Wan, H., Li, H., & Zhang, X. (2018). Opinion mining and sentiment analysis in social media: Challenges and solutions. In *Proc. IEEE Int. Conf. Data Mining Workshops* (pp. 19–26).
- [10] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2), 1–135.
- [11] Sammons, M., Christodoulopoulos, C., Kordjamshidi, P., Khashabi, D., Srikumar, V., & Roth, D. (2016). EDISON: Feature extraction for NLP, simplified. *Language Resources and Evaluation*, 4085–4092.
- [12] Christian, H., Agus, M. P., & Suhartono, D. (2016). Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF). *ComTech Computer Mathematics and Engineering Applications*, 7(4), 285.

- [13] Mujahid, M., et al. (2021). Sentiment analysis and topic modeling on tweets about online education during COVID-19. *Applied Sciences*, 11(18), 8438.
- [14] Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 1, 173–180.
- [15] Tripathy, A., Agrawal, A., & Rath, S. (2016). Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications*, 57, 117–126.
- [16] Goldberg, Y. (2016). A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57, 345–420.
- [17] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- [18] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proc. 2014 Conf. Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543).
- [19] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 135–146.
- [20] Peters, M., et al. (2018). Deep contextualized word representations. In *Proc. 2018 Conf. NAACL-HLT* (pp. 2227–2237).
- [21] Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 79–86).
- [22] Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML)* (pp. 137–142).
- [23] McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. In *Proceedings of the AAAI Workshop on Learning for Text Categorization* (pp. 41–48).
- [24] Mahto, A. K. (2021, September 7). Demystifying Naïve Bayes: Simple yet powerful for text classification. *Medium*. <https://tinyurl.com/j66fbv69> [Accessed: Aug. 15, 2025].
- [25] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Text classification and Naive Bayes. In *Introduction to Information Retrieval* (Chap. 13). Cambridge University Press.
- [26] ResearchGate. (n.d.). Decision tree used for sentiment classification. *ResearchGate*. <https://tinyurl.com/24vdbvp6> [Accessed: Aug. 15, 2025].

- [27] Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998, November). Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management* (pp. 148-155).
- [28] Medium. (n.d.). Sentiment Analysis Using SVM. *ScrapeHero*.
<https://tinyurl.com/3nhjmphu> [Accessed: Aug. 15, 2025].
- [29] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- [30] Jurafsky, D., & Martin, J. H. (2023). *Speech and language processing* (3rd ed., draft). Pearson.
- [31] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference on NAACL-HLT* (pp. 4171–4186).
- [32] Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1746–1751).
- [33] Data Science Today. (n.d.). Sentiment analysis: CNN vs LSTM. *Data Science Today*.
<https://tinyurl.com/bdejhhfk> [Accessed: Aug. 15, 2025].
- [34] Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *Proceedings of INTERSPEECH* (pp. 1045–1048).
- [35] GeeksforGeeks. (n.d.). Introduction to recurrent neural network. *GeeksforGeeks*.
<https://tinyurl.com/5a8cbmh6> [Accessed: Aug. 15, 2025].
- [36] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
- [37] LSTM. (n.d.). LSTM. *Dive into Deep Learning*. https://d2l.ai/chapter_recurrent-modern/lstm.html [Accessed: Aug. 15, 2025].
- [38] Cho, K., et al. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1724–1734).
- [39] ResearchGate. (n.d.). GRU model structure (GRU: Gated recurrent neural network). *ResearchGate*. <https://tinyurl.com/2ww2pz5s> [Accessed: Aug. 15, 2025].
- [40] Brown, T., et al. (2020). Language models are few-shot learners. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 33, 1877–1901.

- [41] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). *RoBERTa: A robustly optimized BERT pretraining approach*. arXiv:1907.11692.
- [42] DeLucia, S. A. (n.d.). Unleashing BERT: Transformer model for NLP. *Arize*. <https://arize.com/blog-course/unleashing-bert-transformer-model-nlp/> [Accessed: Aug. 15, 2025].
- [43] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8), 9.
- [44] Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). *ELECTRA: Pre-training text encoders as discriminators rather than generators*. In *International Conference on Learning Representations (ICLR)*. (arXiv:2003.10555).
- [45] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper, and lighter. *arXiv preprint arXiv:1910.01108*.
- [46] Yang, Z., et al. (2019). XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)* (pp. 5754–5764).
- [47] Jannah, N. Z. B., & Kusnawi, K. (2024). Comparison of Naïve Bayes and SVM in sentiment analysis of product reviews on marketplaces. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 8(2), 727–733.
- [48] Juliane, C. (2021, November). Implementation of naive bayes algorithm on sentiment analysis application. In *2nd International Seminar of Science and Applied Technology (ISSAT 2021)* (pp. 193-200). Atlantis Press.
- [49] Azeraf, E., Monfrini, E., & Pieczynski, W. (2021). Improving usual Naive Bayes classifier performances with Neural Naive Bayes based models. *arXiv preprint arXiv:2111.07307*.
- [50] Sumantiawan, D. I., Suseno, J. E., & Syafei, W. A. (2023). Sentiment analysis of customer reviews using Support Vector Machine and SMOTE-Tomek Links for identifying customer satisfaction. *Jurnal Sistem Informasi Bisnis*, 13(1), 1–9.
- [51] Wirawan, A., & Cahyono, H. D. (2023). Easy data augmentation in sentiment analysis of cyberbullying. In *Proceedings of the 6th International Conference on Information and Communications Technology (ICOLACT)*, Yogyakarta, Indonesia (pp. 443–447).
- [52] Özmen, C. G., & Gündüz, S. (2025). Comparison of machine learning models for sentiment analysis of big Turkish web-based data. *Applied Sciences*, 15(5), 2297.

- [53] Danyal, M., Khan, S., Khan, M., Ghaffar, M., Khan, B., & Arshad, M. (2023). Sentiment analysis based on performance of linear support vector machine and multinomial Naïve Bayes using movie reviews with baseline techniques. *Journal on Big Data*, 5, 1.
- [54] Khan, T. A., Sadiq, R., Shahid, Z., Alam, M. M., & Su'ud, M. M. (2024). Sentiment analysis using support vector machine and random forest. *Journal of Informatics and Web Engineering*, 3(1), 67–75.
- [55] Singhal, S., Majeed, H. L., Ibrahim, H. M., Jatana, N., Gupta, C., Kuma, A., & Hassen, O. A. (n.d.). Sentiment analysis on Amazon reviews of mobile phones using machine learning. *Technology*, 15, 19.
- [56] Ahammad, S., Sinthia, S. A., Ahmed, M., Hossain, M., Asif, N. A. A., & Ikram, N. A. (2024). Deep learning-based sentiment analysis of user generated reviews of various AI powered mobile applications. In *Proceedings of the 2024 International Conference on Inventive Computation Technologies (ICICT)* (pp. 505–512).
- [57] Samanmali, P. H. C., & Rupasingha, R. A. H. M. (2024). Sentiment analysis on Google Play Store app users' reviews based on deep learning approach. *Multimedia Tools and Applications*, 83(36), 84425–84453.
- [58] Amirkhalili, Y., & Wong, H. Y. (2025). Banking on feedback: Text analysis of mobile banking iOS and Google app reviews. *arXiv preprint arXiv:2503.11861*.
- [59] Zaman, N., Goldberg, D. M., Qiao, Z., & Abrahams, A. S. (2025). More than meets the eye: Feature concerns and suggestions in mobile XR app reviews. *Decision Support Systems*, 114475.
- [60] Ranjan, S., & Mishra, S. (2022). Perceiving university students' opinions from Google app reviews. *Concurrency and Computation: Practice and Experience*, 34(10), e6800.
- [61] Basri, N. F., & Utami, E. (2025). Application of Word2Vec and LSTM models in sentiment analysis of Mobile Legends user reviews. *Sistemasi: Jurnal Sistem Informasi*, 14(2), 856–871.
- [62] Eser, G., & Sahin, C. (2024). Sentiment analysis and rating prediction for app reviews using transformer-based models.
- [63] Motger, Q., Miaschi, A., Dell'Orletta, F., Franch, X., & Marco, J. (2024). T-FREX: A transformer-based feature extraction method from mobile app reviews. In *Proceedings of the 2024 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)* (pp. 227–238).

- [64] Nkhata, G., Gauch, S., Anjum, U., & Zhan, J. (2025). Fine-tuning BERT with Bidirectional LSTM for fine-grained movie reviews sentiment analysis. *arXiv preprint arXiv:2502.20682*.
- [65] Chi, D., Huang, T., Jia, Z., & Zhang, S. (2025). Research on sentiment analysis of hotel review text based on BERT-TCN-BiLSTM-attention model. *Array*, 25, 100378.
- [66] Ullah, W., Zhang, Z., & Stefanidis, K. (2023). Sentiment analysis of mobile apps using BERT. In *Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 66–78). Cham, Switzerland: Springer Nature.
- [67] Lee, M. A., Park, Y. J., Na, J. Y., & Sohn, C. B. (2023). Implementation of review sentiment analysis application using KoBERT, KoGPT-2, and KoBART optimized hyperparameters. *Journal of Digital Contents Society*, 24(11), 2831–2840.
- [68] Nugroho, K. S., Sukmadewa, A. Y., Wuswilahaken, D. W. H., Bachtiar, F. A., & Yudistira, N. (2021). BERT fine-tuning for sentiment analysis on Indonesian mobile apps reviews. In *Proceedings of the 6th International Conference on Sustainable Information Engineering and Technology* (pp. 258–264).
- [69] <https://www.kaggle.com/datasets/abiyyurasyiq/mobile-legends-google-play-reviews> last accessed 15/12/2025