AMHARIC LANGUAGE SENTIMENT ANALYSIS USING DEEP LEARNING IN THE CASE OF ETHIOPIA BROADCASTING CORPORATION

A Thesis Presented

by

Haimanot Nekatibeb Kassa

to

The faculty of Informatics

of

St. Mary's University

In partial fulfillment of the Requirements

For the Degree of Master of Science

in

Computer Science

February, 2025

Addis Ababa, Ethiopia

ACCEPTANCE

Amharic Language Sentiment Analysis Using Deep Learning In The Case of Ethiopia Broadcasting Corporation

By

Haimanot Nekatibeb Kassa

Accepted by the Faculty of Informatics, St. Mary's University, in partial fulfillment of the Requirements for the Degree of Master of Science in Computer science

Thesis Examination Committee:

Internal Examiner

Mulugeta Adebaru (Ph.D)

External Examiner

Mesfin Abebe (Ph.D)

Dean Faculty of Informatics

Alembante Mulu kumlign (Ph.D)

February, 2025

DECLARATION

I, the undersigned, declare that this thesis work entitled **Amharic Language Sentiment Analysis Using Deep Learning in the Case of Ethiopia Broadcasting Corporation** is my original work, has not been presented for a degree in this or any other Universities, and all sources of materials used for the thesis work have been duly acknowledged.

Declared by

Haimanot Nekatibeb Kassa

Full Name of Student

_____

Signature

Addis Ababa

Ethiopia

This thesis has been submitted for examination with my approval as advisor

Alembante Mulu kumlign (Ph.D)

Full Name of Advisor

_____

Signature

Addis Ababa

Ethiopia

February 2025

Certificate

This is to certify that this thesis work entitled "Amharic Language Sentiment Analysis Using Deep Learning in the case of Ethiopia Broadcasting Corporation" which is submitted by ***Haimanot Nekatibeb Kassa*** is carried out under my advisor and guidance for fulfilling the nature and standard required for partial fulfillment of the requirements of masters of Science in Computer Science. The work in this thesis has not been submitted elsewhere for a degree.

Advisor: Dr. Alembante Mulu kumlign
St. Mary's University
Faculty of Informatics
Date: February, 2025

Acknowledgment

Above all, I want to sincerely thank the Eleshaday God for guiding me to success in this study and in my life.

I want to sincerely thank Alembante Mulu (PhD), my adviser, for his insightful counsel and excellent support, both of which were crucial to finishing this study.

I am thankful to the Ethiopian Broadcasting Corporation for providing essential resources and support that greatly helped make this thesis successful.

I have the utmost gratitude for my husband, Daniel Mulat Workeneh, for his unfailing love, support, and inspiration along this journey.

My appreciation also goes to dear friends, whose support, encouragement and bolstered my confidence in different aspects of the thesis.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AI	Artificial Intelligence
ABSA	Aspect-based Sentiment Analysis
BERT	Bidirectional Encoder Representations from Transformer
Bi-LSTM	Bidirectional Long Short-Term Memory
BNC	Bayesian Network classifier
CL	Convolution Layer
CLSA	Concept Level of Sentiment Analysis
CNN	Convolutional Neural Networks
CRF	Conditional Random Field
DBN	Deep Belief Networks
DLSA	Document-Level Sentiment Analysis
DL	Deep Learning
DNN	Deep Neural Networks
DRL	Deep Reinforcement Learning
EBC	Ethiopia Broadcasting Corporation
FCL	Fully Connected Layer
FlL	Flattening Layer
GPU	Graphics Processing Unit
HTA	Hybrid Techniques Approach
JASIST	Journal of the American Society for Information Science and Technology
LBA	Lexicon-Based Approach

LC	Linear Classifier
LIWC	Linguistic Inquiry and Word Count
LSTM	Long Short-Term Memory
ML	Machine Learning
MLA	Machine Learning Approach
MLM	Masked Language Modeling
NSP	Next Sentence Prediction
NLP	Natural Language Processing
OM	Opinion Mining
PL	Pooling Layer
POS	Parts of Speech
PSB	Public Service Broadcasters
PSM	Public Service Media
R2	Coefficient of Determination
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RMLA	Reinforcement Machine Learning Algorithms
RMSE	Root Mean Squared Error
RNN	Recursive Neural Network
RvNNs	Recursive Neural Networks
SA	Sentiment Analysis
SL	Supervised Learning
SLSA	Sentence Level Sentiment Analysis

SMLA	Supervised Machine Learning Algorithms
SVM	Support Vector Machine
TF-IDF	Term Frequency - Inverse Document Frequency
UL	Unsupervised Learning
USLA	Unsupervised Learning Algorithms

Abstract

Sentiment analysis has become an essential tool for understanding public opinion on social media platforms, particularly for organizations like the Ethiopian Broadcasting Corporation (EBC). However, the unique linguistic features of Amharic, a morphologically rich and under-resourced language, pose significant challenges to developing accurate sentiment analysis models. This research explores sentiment analysis on Amharic Facebook comments related to EBC using state-of-the-art deep learning models.

A comprehensive dataset of 22,000 labeled Amharic Facebook comments is prepared, with sentiments classified into five classes: positive, neutral, negative, strongly positive, and strongly negative. The labeling workflow for Amharic sentiment analysis begins with manual annotation, where each sentence has been reviewed by trained annotators who Amharic linguistic professionals assign the most fitting sentiment label based on predefined criteria. These annotators are skilled at identifying key phrases and contextual signals that align with each sentiment category. Multiple annotators review each sentence to ensure consistency, and any disagreements are resolved either through a voting system or by an expert review to achieve consensus.

The data underwent rigorous preprocessing, including tokenization, character-level normalization, and noise reduction, to address challenges such as data sparsely and mixed sentiment labels. Four deep learning models—LSTM, Bi-LSTM, CNN, and fine-tuned Multilingual BERT—have been assessed using accuracy, precision, recall, and F1-score as performance metrics.

Among the models, Multilingual BERT achieved the highest performance, with an accuracy of 87.2%, precision of 0.796, recall of 0.7815, and F1-score of 0.8143, highlighting its capacity to leverage pre-trained multilingual embedding for handling Amharic's complex structure. Bi-LSTM, CNN, and LSTM models demonstrated moderate performance, with Bi-LSTM achieving an accuracy of 81.3% and an F1-score of 0.76, CNN reaching an accuracy of 79.68% and an F1-score of 0.743, and LSTM achieving an accuracy of 77.12% and an F1-score of 0.752, showcasing strengths in capturing sequential and local patterns but lagging behind BERT due to the absence of pre-trained embedding and attention mechanisms.

This study adds to the growing body of research on sentiment analysis for under-resourced languages, providing insights into the efficacy of deep learning models for EBC Facebook Amharic sentiment categorization. The findings underscore the importance of pre-trained models like BERT in overcoming linguistic challenges and advancing sentiment analysis applications for public service broadcasters. These results serve as a foundation for further exploration of Amharic sentiment analysis, with implications for improving public engagement and decision-making within EBC.

Keywords: - Amharic Sentiments, Sentiment Analysis, Deep Learning, BERT, Bi-LSTM, LSTM & CNN.

Chapter One

1. Introduction

1.1. Background of the Study

Sentiment, which is a consistent pattern, systematic processes and guidelines for making sense of the world by combining affect and emotion, so highlights the necessity of sophisticated opinion mining methods in order to decipher the complex terrain of online sentiments, (Erik, 2013). By research introduced to, (Alexander 2009) which explores sentiment analysis using Natural Language Processing (NLP) techniques to determine whether a paper conveys Positive, neutral, or negative sentiment toward a certain target is the main emphasis. It describes how NLP techniques including dependency analysis, coreference resolution, and part-of-speech tagging are used to improve sentiment categorization.

The authors (Ameen et al., 2019) emphasize the need for enhanced approaches, such as the application of machine learning and natural language processing techniques, which can more effectively manage the intricacies of sentiment. It provides a detailed review of sentiment analysis, studying its emerging principles and the numerous approaches applied to assess sentiments conveyed in text additionally, discusses the difficulties in effectively expressing the subtleties of human emotion in written language and talks about the value of sentiment analysis in comprehending public opinion, especially in the context of social media and online interactions, (Ameen et al., 2019).

“Social networking sites have become staples in everyday life in many parts of the world” (Hallvard, 2013). It focuses on sentiment analysis applications in disaster situations, showing how social media insights might improve first responders' situational awareness in emergency circumstances; (Ghazaleh et al. 2016). Research (Yuriy et al., 2018) draws attention to how quickly information travels on social media sites like Twitter, frequently using echo chambers to confirm preexisting beliefs. Bots are crucial in spreading or modifying messages, changing human interaction, and maybe influencing outcomes of

elections. While recognizing the difficulties in controlling the flow of information on social media, the results highlight the necessity of addressing the impact of bots and false information in preserving democratic integrity, (Yuriy et al., 2018).

The research show about, (Jie et al., 2022) social media has had a big influence on traditional media by acting as an attention generator rather than competing information channel so, social media raises the number of people who read linked news pieces in traditional media by sparking conversations and bringing attention to issues.

When social media posts are more intense, have a more positive sentiment, or originate from reliable sources, this complementing link is especially noticeable. Consequently, although social media platforms change how people consume news, they also benefit traditional media by channeling attention to in-depth, professional reporting, for instance the genuine information which Facebook content is highly emotive, has a larger readership, and has a higher chance of spreading so the platform also generates and disseminates user-generated content, (Jie et al., 2022).

Overall, sentiment analysis turns unprocessed social media data into useful insights that enable academics, companies, and researchers to make wise choices in a data-driven digital environment. Facebook also dominates the digital marketing space (Federico et al., 2012).

Furthermore, researcher "argues this phenomenon is manifested in the transformation of sociability that results from networked individualism which is not a consequence of the Internet or new communication technologies. But a change that is fully supported by the logic embedded in the communication networks" (Corinne and Manuel 2018, 162). Therefore, PSB is impacted by commercial competition, political and economic pressures, audience shifts, and digitization. Moreover, PSB needs to adapt to the current circumstances. Because it makes decisions based on social media material, (Agnes & ferenc, 2013). People's desire to engage with PSB depends heavily on public opinion. Users may not give you a chance once they've read a few bad reviews, (Corinne and Manuel 2018).

(Federico et al., 2012), a plethora of data reflecting customer choices, activities, and responses is available on social media sites like Facebook and Twitter. Sentiment analysis

may categorize text as positive, negative, or neutral by using methods like supervised and unsupervised machine learning. It provides information on public sentiment, product reception & brand perception addition to measuring client loyalty. This technique helps identify influencers, guide marketing strategy, and assess campaign efficacy, (Federico et al., 2012).

Given the popularity of social media, sentiment analysis is essential for consumers to make educated judgments about what to buy and for companies to compare goods and services, (Bing, 2015). The paper focuses more on sentiment analysis, which is an aspect of natural language processing (NLP). The computing procedure of identifying and classifying opinions in text as positive, negative, or neutral. When we apply machine learning and NLP techniques, sentiment analysis algorithms are able to sort through large volumes of textual data and fetch insightful information without violating copyright, (Snehal et al., 2024).

"The challenge for sentiment analysis is lack of sufficient labeled data in the field of Natural Language Processing (NLP). And to solve this issue, sentiment analysis and deep learning techniques have been merged because deep learning models are effective due to their automatic learning capability", (Qurat et at., 2017).

Particularly, deep learning has become a potent machine learning method that may yield cutting-edge sentiment analysis prediction outcomes these days, (Yeshewas, 2023). Both supervised and unsupervised learning benefit greatly from deep learning and many researchers are employing it to handle sentiment analysis, (Qurat et at., 2017).

Therefore, goal of this paper is to construct a model that analyzes comments on EBC Facebook accounts using deep learning approaches. Thus, EBC as public media is anticipated to perform a paramount role in the nation branding process at the same time. It should venture onto social media services, aiming to reach new users in the nation branding process. Plus, it is time to analyze various feedbacks for enhancing EBC's engagement through social media motivates the researcher to conduct the study by itself, (Jordan, 2014).

1.2 Motivation

Public service broadcasters (PSB) like EBC venture onto social media services, aiming to reach new users. EBC's decision-making is highly exposed to the public. Further, this day nation branding is the new concept introduced into EBC's decision-making process. Nation branding is a process of transforming national allegiance, structures of belief, and the sense of belonging. Even some writers are convinced it is a tool for nationalism, (Jordan, 2014). Therefore, both nation branding and sentiment analysis are interconnected as sentiment analysis can be used to evaluate and mold a country's image. Nation branding encompasses a country's overall image, including cultural, political, and social aspects. While sentiment analysis involves assessing public opinion and emotional responses towards a nation and analyzing sentiments expressed in social media, news, and other platforms, marketers. Researchers can derive strategies to enhance a nation's brand and address any negative perceptions, (Jordan, 2014).

Moreover, when we see the nation branding and sentiment analysis using deep learning involves utilizing advanced machine learning techniques to evaluate and improve the perception of a nation like, typically includes collecting data from social media and other online platforms. Where sentiment analysis models—often based on deep learning architectures—classify opinions and sentiments as positive, negative, or neutral, (Jordan, 2014). National allegiance and belonging in the context of deep learning sentiment analysis refers to how sentiments (such as pride, loyalty, or a sense of community) related to one's nation or cultural identity are detected and understood from text or other data. This typically involves using natural language processing (NLP) models and techniques to analyze how individuals or groups express feelings about their nation or cultural affiliation. EBC as public media is anticipated to perform a paramount role in the nation's branding process. So, it needs to examine comments posted on its Facebook account in the eyes of the nation's branding process, (Jordan, 2014).

The resulting unstructured user-generated sentiment mandates new computational techniques. "Deep learning has emerged as a powerful machine learning technique that

learns multiple layers of representations or features of the data and produces state-of-the-art prediction results.", (Yeshewas, 2023).

Specifically, EBC needs to identify comments posted on its Facebook account in terms of words that stand for the lowest sense of national allegiance and belonging which is Strongly Negative; a low sense of national allegiance and belonging, which also Negative; a moderate sense of national allegiance and belonging, which is Neutral; a high sense of national allegiance and belonging which, is Positive; and the highest sense of national allegiance and belonging which is also Strongly Positive. Therefore, the author of the thesis aims to build a model that analyzes comments on EBC Facebook profiles using deep learning.

1.3. Statement of problem

In lately year, opinionated remarks on social media platforms have had a significant influence on our social and political lives. Therefore, it is reshaping business and influencing public environment, Nonetheless, it is still quite difficult to locate and keep an eye on online opinion sites and to fetch the information they contain, (Bing, 2015). The potential for social media sentiment analysis to benefit public service broadcasters (PSBs) remains unclear. Furthermore, "The challenge for sentiment analysis is lack of sufficient labeled data in the field of Natural Language Processing (NLP) & to solve this issue. The sentiment analysis and deep learning techniques have been merged because deep learning models are effective due to their automatic learning capability.", (Qurat et al., 2017, 424).

In regard to, over the past decade, literatures have been not silent about sentiment analysis. For instance, "Research by highlighted Deep learning models, such as bidirectional encoder representations from transformer (BERT), sentiment-specific word embedding models, Cognition- based attention models, Common Sense Knowledge, Reinforcement Learning and Generative Adversarial Networks have been used to address various challenges in sentiment analysis", (Olivier et al., 2020). Further, the study also proposed a sentimental classification of a large number of tweets, (Shilpa et al., 2021).

The study proposed to “While these results mark a significant milestone, challenges persist, such as the need for a more extensive, diverse dataset and, underscores the importance of transitioning from binary sentiment analysis to a multi-class classification approach, enabling a finer-grained understanding of sentiments.”, (Fikirte et al. 2023). So, by noting this gap; this research is intended to construct a model using much dataset that analyzes (Culture, Social, Politics, Entertainment, and Economy comments). We have been using multi classification labeling like (Strongly Positive, Positive Strongly Negative, Negative & Neutral labeling) on EBC Facebook accounts using an appropriate deep learning approaches.

1.4. Research questions

- How is the required Amharic Language sentiment analysis dataset preprocessed?
- Which existing methods are more appropriate for Amharic language Sentiment analysis?
- How can deep learning models be designed for Amharic language sentiment analysis?
- Which Deep learning model provides better performance for Amharic language sentiment analysis?

1.5 Objective

1.5.1. General objective

The study's overarching goal is to build a deep learning algorithm that analyzes Amharic comments on Facebook account of Ethiopia Broadcasting Corporation**.**

1.5.2. Specific objective

- To review state-of-the-art literature on Amharic Sentiment analysis.
- To study the nature and structure of Amharic Language.
- To prepare the required dataset for Amharic sentiment analysis.
- To identify appropriate features for Amharic Sentiment analysis.
- To design a deep learning model for Amharic sentiment analysis.
- To evaluate the performance of the developed Deep Learning model.

1.6. Scope and Limitation of the Study

Developing a model that analyzes comments on EBC Facebook accounts is the major theme of this study, which have been help EBC to be well-informed about the opinions of the general public in terms of its national branding process.

Analyzing comments on EBC Facebook accounts as a whole is difficult because it is a large public service broadcaster with several streams (i.e.; EBC Entertainment, EBC News, and EBC Language) and with so many languages.

The study solely has focused on the EBC News stream linked to culture, politics, social, economy and entertainment. In addition, the study has concentrated on Amharic Facebook comments. So, the study's main focus has been to predict users' comments with a sense of national allegiance and belonging from the posted Amharic comments.

This analysis has been conducted using deep learning processing the data and training model on a computer or other device with a high computational resources like High-performance graphical processing units (GPUs), adequate memory for handling large datasets, and storage capacity for hosting those datasets and also, need large amounts of high-quality data has been required due to the research only emphasis on the EBC Amharic Facebook comments.

1.7. Significance of the study

It is hoped that the research would add to the body of existing information while also compensating for the scarcity of scientific articles on sentiment analysis. The look for based on investigation intervention ranked high on the top list for sentiments of reviews. Moreover, Sentiment analysis on social media networking and how it might be used for public service broadcasters (PSB) is still the question on the table particularly; this bleak scenario was presented by reports, (2023).

Therefore, this study is helpful in closing this gap; for example, EBC would be well-informed on the general public's perspectives on its national branding process. Plus, the analysis of various feedbacks has the ability to provide essential insight for practitioners.

Policymakers in terms of developing policies and adopting necessary measures to improve EBC's involvement on social media.

Further, the findings of this research have the potential to give practitioners with valuable insights into the build deep learning models for analyzing Amharic datasets. Finally, the investigation has been useful as an empirical source for the researcher in the future.

1.8. Organization of the study

This research has been arranged into five parts. Chapter one delineates the study by outlining the background, motivation, problem statement, research questions, objectives, importance, scope and constraints. Chapter two has provided a comprehensive literature review, defining key concepts such as sentiment analysis and deep learning. Discussing topics including public service broadcasters in the network era, sentiment analysis methods, features commonly has used, and sentiment analysis applications. It has also been reviewed related works and identifies research gaps addressed in the study. Chapter three has described the research design & methodology, detailing the proposed architecture, data collection, preprocessing, train-test splitting, hyper parameter tuning, training with cross-validation, and the application of deep learning models such as LSTM, Bi-LSTM, CNN, and fine-tuned multilingual BERT, alongside evaluation metrics. Chapter four has presented the results and discussion, covering the tools used, experimental findings, performance of the models, and comparisons with prior works, with interpretations aligned to the literature review. Finally, Chapter Five concludes the study by summarizing the findings, talking about their implications, and providing recommendations for future work and potential applications.

Chapter Two

2. Literature Review

2.1. Introduction

This chapter an aim has to provide a detailed review of the literature related to sentiment analysis, focusing on its application to Amharic Facebook comments for the Ethiopian Broadcasting Corporation (EBC). The chapter explores various approaches employed by researchers to analyze and classify sentiment in social media text, emphasizing techniques tailored to under-resourced languages like Amharic.

A thorough review of deep learning applications in sentiment analysis is presented, covering methods such as LSTM, CNN, Bi-LSTM, and fine-tuned Multilingual BERT, as well as traditional approaches. These techniques are examined in the context of their efficacy in handling the challenges of morphologically rich languages and their capacity to capture contextual variations.

The chapter has also discussed evaluation metrics commonly used in sentiment analysis, including accuracy, precision, recall, and F1-score, detailing how these metrics has been applied to assess model performance. Additionally, it has highlighted the role of robust preprocessing techniques, such as tokenization and character-level normalization, in improving classification accuracy.

By including a review of related studies and the datasets utilized for Amharic sentiment analysis, this chapter offers insight into the present state of research, identifies gaps, and underscores the importance of tailored approaches for sentiment classification in under-resourced languages. Finally, the chapter has concluded with a summary of the findings, setting the foundation for the subsequent methodology and experimental discussions.

2.2 Conceptualization of Key Terms

2.2.1. Sentiment

The study proposed to, idea of sentiment links affective and emotional dynamics with the cognitive processes involved in creating views and judgments. So, sentiment is a regular

pattern, orderly procedures, and rules of how sense is made of the world by encompassing affect and emotion, (Erik, 2013). Opinions are supposed to be binary, such as "for/against," "like/dislike," or "good/bad" etc. Sentiments could be experienced in a diversity of ways such as, hazy gut sentiments, poorly developed intuitions, clearly formed opinions, and strong judgments. Thus, each of these can be stated as an opinion on how to evaluate someone's conduct. A judgment on which system of rules have been preferred over another, a sense of the reality of theoretical premises, and a choice over the legality of a political measure, (Jonas and Olaf, 2019).

The sentiment associated with (Jonas and Olaf, 2019); (Paula, 2009). Micro, Meso, and Macro levels typically reflect the scope and focus of analysis in social sciences and research. Micro level examines individual behaviors and interactions, often focusing on personal sentiments, motivations, and experiences. The sentiment here can be very personal and subjective. Meso level looks at groups, communities, or organizations, analyzing collective sentiments and dynamics within these entities the sentiment, (Jonas and Olaf, 2019); (Paula, 2009). It can reflect group attitudes and social norms. Macro level: studies broader institutional and societal structures, often encompassing national or global sentiments. The focus is on trends, policies, and collective societal attitudes. Each level provides different insights into sentiments, with micro being more personal, meso being more communal, and macro being more structural and overarching. The earliest conceptual basis for conceptualizing sentiment was linked to the Aristotelian tradition of virtue ethics. The ethics of virtue see ethical activity as a question of shaping oneself as a moral subject by paying particular focus to the consequences (consequentialist ethics), (Jonas and Olaf, 2019); (Paula, 2009). The second conceptual foundation in moral philosophy was Scottish moral sense theory, which was chiefly articulated by David Hume and Adam Smith who stressed the importance of sentiment, which includes both judgment and affection, and the interplay between opinion and feeling in moral evaluations, (Anna, 2007).

2.2.2. Sentiment Analysis

“Sentiment Analysis or opinion mining is defined as the computational study of opinions, sentiments, appraisals, evaluations, and emotions”, (Bing, 2015). It is also known as opinion mining, is a NLP technique used to identify and analyze the emotional tone, opinions, or sentiments expressed in text and determines whether the sentiment is positive, negative, or neutral, often applied to understand customer opinions, brand reputation, or social media trends, (Erik, 2013).

Sentiment analysis (SA) is the extensive examination of online data to identify and classify the opinions expressed in a section of the text and process evaluates the author's perspective on a certain subject, film, or item. It may give the output like positive, negative, and neural, (Alexander 2009). It is a part of Natural Language Processing (NLP) activity that deals with the identification and categorization based on the given data to analyze the prediction of sentiments in texts, (Alexandra, 2013). Sentiment analysis is the most frequent text categorization method, analyzing an incoming message to identify if the underlying sentiment is positive, negative, or neutral, (Bing, 2015).

SA requires five procedures for analyzing data. This includes gathering data, text preparation, sentiment detection, sentiment categorization, and outcome display. Gathering data which is one of the steps of sentiment analysis can acquire raw data by different mechanisms from various sources including user groups, Twitter, Facebook, blogs, and commercial websites such as amazon.com and Alibaba.com, among others, (Ameen et al., 2019). Text preparation is the stage in which non-analysis-related text is deleted. Sentiment detection, often referred to as opinion mining (OM) new line or sentiment analysis, is the technique of determining the sentiment newline expressed in a review through the use of machine learning or natural language processing (NLP), (Ameen et al., 2019). SA and OM are pivotal in examining the vast amounts of user-generated content on social network site platforms like Twitter, Facebook, and LinkedIn etc. which techniques fetch valuable insights from textual data by identifying classifying sentiments, emotions, and opinions on specific topics when we input the model using related programing code Plus, Social network site’s dynamic nature has created an

increasing demand for automated tools to handle and analyze the ever-expanding data it operates at document, sentence, and aspect levels, with applications ranging from tracking consumer opinions to predicting political outcomes and analyzing healthcare-related reviews, (Zaher et al. 2019).

The study showed, (Khalil et al., 2021; Seema et al., 2015; Mayur et al., 2022 on the data-level basis, the sentiment analysis system has been divided into three different tiers which are document, sentence and aspect level of sentiment analysis The first one level of system analysis which indicates the entire opinionated document is classified using document-level sentiment analysis (DLSA) which, A single information unit is symbolized by a document that contains ideas or views about a specific topic. The second one of Level of sentiment analysis is Sentence level sentiment analysis (SLSA) which categories every sentence in a document. A sentence is initially identified as opinionated or non-opinionated using a method called subjectivity categorization; after that, the resulting opinionated statements are classified as conveying positive or negative ideas. The last one of level of sentiment is called Aspect-based sentiment analysis (ABSA) which collects and organizes the public impressions about entities and their associated aspects/features, known as targets. ABSA is a more finer-grained approach than DLSA and SLSA, (Khalil et al., 2021; Seema et al., 2015; Mayur et al., 2022.

2.2.3. Public Service Broadcasters in the network era

Technological development has imposed institutional changes on Public Service Broadcasters. For instance, PSB should offer content not only through linear television, radio, and so on but also across different platforms. In doing this PSB should also maintain its mission for public Service. Actually, scholars believe that PSB so far has passed through five types of development. Broadcasters enter the new era at different stages since they work in various settings and with various administrative structures, (Corinne and Manuel 2018).

Initially, PSB undergoes an experimental phase in which, as the name suggests, the broadcaster experiments with the options that technology and the Internet give and test its constraint and capacity, (Corinne and Manuel 2018). When public broadcasters discover

that private companies are growing their internet operations because they are institutional in character, they are drawn to the second stage, which is known as the panic phase, (Katsoulakis, 2022). The expansionist stage, which follows, is comparable to the panic phase but places more emphasis on new internet media. The mature phases leading up to the switch to public service media (PSM) are the final two stages. The process of determining what is crucial in the digital transformation while concentrating on distribution and preserving its public service character is known as the consolidation phase, which is the fourth step. Lastly, there is the stage of maturity, (Yuriy et al., 2018).

2.3 Main Approaches to Sentiment Analysis

In contemporary studies literature categorized approaches to sentiment analysis into keyword spotting, lexical affinity, and statistical methods, (Erik, 2013). Due to its affordability and ease of usage, keyword spotting is the most basic strategy and most likely the most widely used one. Due to the presence of comparatively clear affect words like happy, sad, afraid, and bored, texts may be grouped into effect groups, (Tetsuya and Jeonghee, 2003). Next to keyword spotting, the researcher read a lot of stuff about lexical affinity which lexical affinity is a bit more sophisticated than keyword detection by assigning random words a probabilistic "affinity" for a certain emotion rather than only recognizing words with obvious effect, (Maria et al., 2014).

Further, paper highlights the effectiveness of algorithms such as Naive Bayes (NB) and Support Vector Machines (SVM) for sentiment classification tasks which is noted for its efficiency with smaller datasets, while SVM excels in high-dimensional spaces by finding optimal hyper planes for classification, (Mayur et al., 2022).

2.3.1. Concept-Level Sentiment Analysis

The research proposed about concept-based methods, as opposed to strictly syntactical ones, can identify feelings that are subtly represented by analyzing ideas that do not express any emotion directly but are indirectly connected to other concepts that do, (Ghazaleh et al. 2016). It draws attention to the shortcomings of conventional sentiment analysis techniques, which frequently depend on assessments at the document or sentence

level, and underscores the significance of semantic analysis through the use of online ontologies and semantic networks, (Erik, 2013). By identifying the connections between ideas and the attitudes they evoke, CLSA facilitates feature-based sentiment analysis and allows for a more sophisticated comprehension of opinions. So, comprehensive knowledge bases are essential to CLSA's efficacy since they aid in capturing the intricacies of human language and sentiment expression across a range of topics, improving the automatic analysis of online opinions, (Erik, 2013).

Instead of gathering conflicting opinions about a single item (like the iPhone5), consumers are usually more interested in comparing products based on their specific features (like the touchscreens of the iPhone5 and Galaxy S3) or even sub-features (like the vulnerability of the iPhone5's versus the touchscreen of the Galaxy S3), (Federico et al., 2012).

2.3.2. Features Commonly Used in Sentiment Analysis

The researcher proposed to the feature extraction in sentiment analysis involves transforming raw text into structured representations that machine learning models or algorithms can process and to capture the sentiment-relevant information from text while filtering out irrelevant data. So, when we want to implement sentiment analysis on NLP, utilize the features extraction commonly like Term Position, N-gram Features, Parts of Speech, and Term Presence vs Term Frequency etc., (Dipti et al., 2020).

Term Presence and Term Frequency are two approaches used in text analysis to represent textual data numerically. But, they differ in how they account for the occurrence of terms within a document which Term Frequency is used to determine how many terms appear in the corpus but, a binary-valued feature vector is called Term Presence and shows whether or not the term appears in the sentence, (Dipti et al., 2020). A value of 1 show that the word is present, whereas a value of 0 shows that it is not. Next to that, the researcher read a lot of stuff about N-gram Features which are broadly utilized in Natural Language Processing that is when several terms appear together in a text. It is called a unigram when only one phrase is used as a feature and a bigram when two are used, (Erik, 2013; Zaher et al., 2019; Yelena and Padmini, 2011 and Dipti et al., 2020).

After stopping for a while, the researcher also reviewed literature about Parts of Speech (POS) Tagging which is the procedure of giving each word in a text, for instances, noun, verb, an adjective, an adverb, preposition, conjunction, interjection, determiner etc., (Dipti et al., 2020). This is done to analyze the syntactic structure of a sentence and understand the grammatical function of each word in context, (Erik, 2013); (Zaher et al., 2019); (Yelena and Padmini, 2011) and (Dipti et al., 2020).

2.3.3. Sentiment Analysis Methods

The researcher proposed and showed, (Dipti et al., 2020) the Sentiment analysis methods can be broadly categorized into three main approaches: rule-based, machine learning-based, and hybrid. There are rule-based methods rely on predefined rules & sentiment lexicons to classify text as positive, negative, or neutral, offering simplicity but limited context understanding and machine learning-based methods use algorithms like Naïve Bayes, SVM, or advanced models like BERT to learn sentiment patterns from labeled datasets, providing greater accuracy but requiring substantial data and computational resources. And also, hybrid methods combine rules with machine learning to leverage both interpretability and performance therefore, this study reviewed and focused on literature about Machine Learning Approach, (Dipti et al., 2020).

The researcher introduced various machines that were created as a result of the human brain's inventiveness by allowing individuals to meet a variety of demands, such as computing, travel, and industry. These technologies made life easier for everyone among these is machine learning, (Batta, 2020). “This was done using a Perceptron program designed by Frank Rosenblatt in 1957 that combines the properties of Hebb’s neuron functionalities with those of the algorithm in the Checkers game”, (Izunna, 2023, 29). The perceptron is an early type of artificial neural network designed for pattern recognition.

(Batta, 2020), the researcher highlighted the role of ML in automating tasks and enabling data-driven insights across different areas and defines. It as a field that equips computers with the ability to learn without clear software design, primarily by leveraging massive corpus to identify patterns and make predictions. So the review underscores ML ubiquity

in daily applications, from personalized recommendations to social networking, (Batta, 2020).

Machine learning is the process of automatically calculating tasks using binary or logical operations that are taught from a series of examples, (Donald et al., 1995). In general, machine learning techniques are adaptable, nonparametric approaches to data classification or forecast, (Trent et al, 2018). After looking at the data, we occasionally find that we are unable to accurately interpret the information. Consequently, we use machine learning, (Asongo et al., 2021).

The study reviewed, (Taiwo, 2010) which ML is widely applied in diverse fields like healthcare, robotics, NLP, and gaming, making tasks like image recognition, speech recognition, and personalized recommendations possible. It supports knowledge acquisition and skill refinement, fostering advancements in both theoretical research and practical applications, ultimately driving innovation across industries.

2.3.3.1. LIWC Method

When we discuss, (Yla and James, 2010) the Linguistic Inquiry and Word Count (LIWC) applications, an automated text evaluation technique intended to examine the psychological meaning of words. It has been investigated in this study which identifies word usage patterns in categories like attention focus, emotionality, social interaction, and thinking styles. And also It sheds light on individual variances in areas including interpersonal dynamics, emotional moods, and cognitive complexity, (Yla and James, 2010).

(James et al., 2001), “The first LIWC application was developed as part of an exploratory study of language and disclosure (Francis, 1993; Pennebaker, 1993),” (p.12). LIWC which is the primary function of processing text files, finding words that fit its vocabulary, and assigning them into categories such as affective processes (either positive or negative feelings), cognitive mechanisms (insight, causality), and social references (friends, family), (James et al., 2001). The utility can easily process many files and produce data that may be further examined with Excel or SPSS. Because of its

adaptability, it may be customized to fit particular research needs via user-defined dictionaries and customized word categories, (James et al., 2001).

Further, the study showed of LIWC which serves as a valuable tool in analyzing how Marriage and Family Therapist Websites communicate through their websites, helping to reveal marketing strategies and emotional engagement with potential clients, (Thomas, 2019). The paper highlights LIWC's empirical validation and its ability to process vast amounts of text, linking language use to various psychological and social behaviors, (Yla and James, 2010). Lastly, the goal of this was to create software that would merely search through several text files for and count terms in areas related to psychology as a consequence, LIWC (pronounced "Luke") which is a dynamic computer application, (James et al., 2001).

Further, (Yla and James, 2010) LIWC is distinct because it combines a transparent processing system with well-curated dictionaries of words categorized by psychological relevance and innovation enables researchers to measure linguistic elements such as function words, pronouns, and verb tenses, offering deeper understanding of how language reflects thought. Emotion also the applications span areas like assessing social status, identifying deception, and studying emotional responses, (Yla and James, 2010).

2.4. Artificial Intelligence

To mimic and improve human intelligence, artificial intelligence (AI) aims to develop theories, approaches, formulas, and software applications, (Li, 2018). “Machine learning, especially deep learning, catapulted AI applications to greater heights, making machines perform tasks previously exclusive to humans.” (Durga et al., 2024) When we see Artificial Intelligence technology rise up or develop to day to day gradually and, (Sibanjan and Umit, 2018) it accomplishes a good performance for tasks with more accuracy and more fast than human brain so machine learning and deep learning are the parts of AI which are a huge topic in computer science, Nonetheless, deep learning outperforms machine learning in AI applications, (Sibanjan and Umit, 2018).

2.4.1. Machine Learning

Machine learning has utilized machines to handle data more efficiently, (Alekhya 2020). Learning is characterized as the activity of gaining information with a purposeful focus on a meaningful subject, (Ripu and Rajani 2021). The internal results or outcomes of learning are the acquisition of experience, knowledge, and the discovery of laws, (Ripu and Rajani 2021). Comparably, an external image would enhance system performance by accommodating or adapting to the surroundings, (Ripu and Rajani 2021).

In the context of Artificial Intelligence (AI) applications, machine learning is viewed as an important field of study, (Ripu and Rajani 2021). Making computers mimic and adopt human learning behaviors is the subject of the study of machine learning. It is also described as the research into how computers can pick up new abilities and knowledge, work to better them, continuously strive to perform better, and recognize previously acquired knowledge, (Ripu and Rajani 2021).

The researcher highlighted how these machines simplified and diverse functions human existence by allowing humans to satisfy a variety of interests and areas. It performed the data analysis and pattern recognition, prediction and forecasting, classification, clustering, optimization, automation, decision support, personalization and etc., (Batta, 2020). The study discussed the taxonomy of primary categories of ML algorithms, which are supervised, unsupervised, semi-supervised, and reinforcement learning also explain their importance in resolving problems like classification, clustering, and forecast, (Ravil, 2015).

- **Supervised Learning**

The supervised machine learning algorithms need external assistance, (Alekhya 2020). Each dataset used for machine learning includes examples. Features are used to define or represent instances, (Kotsiantis et al., 2007). These characteristics can be "continuous" or "binary/categorical". Instances that have labels on them are those that match the desired outcome or response, (Kotsiantis 2007). When training and testing data are labeled, supervised machine learning techniques are applied, (Saravanan and Sujatha, 2018). Using

labeled data from the past and present, these algorithms learn and make predictions, (Saravanan and Sujatha, 2018).

Let us see it first, (Ravil, 2015) the researcher discussed about supervised machine learning which role in utilizing labeled data to train models for classification or regression tasks. The process which involves utilizing a training corpus where inputs are paired with corresponding outcome, (Ravil, 2015). So the purpose to map input data to the corresponding outcome utilizing a learned function, the algorithms are categorized as linear or nonlinear based on how they separate data points into classes or regress outcome values. It contains linear regression, logistic regression, support vector machines (SVM), decision trees, random forests, k-nearest neighbors (k-NN), and artificial neural networks (ANN). Techniques have been utilized for classification, such as categorizing emails as spam, and regression tasks, like predicting house prices, relying heavily on the quality of training data to ensure model accuracy and generalization to unseen data, (Ravil, 2015).

Classification

An outcome space containing several classes constitutes the classification problem in supervised learning, (Peter 2012). As an illustration, "YES," "NO," or "MAYBE" stand for the outcome element of the input that decides if it might rain today or not. A great example of a classification problem is the one this thesis is trying to solve, which is sentiment analysis.

Regression

A set of continuous numbers serves as the outcome space for the supervised learning problem known as regression, (Peter 2012). The statistically-based regression algorithms estimate the connection between the outcome and the characteristics that affect the outcome, (Mohamed and Ali, 2017).

- **Unsupervised Learning (descriptive task)**

  Unsupervised learning, as opposed to supervised learning, has been used on instances that are not labeled. Its algorithms examine unlabeled data and provide a function that describes the data's pattern, (Saravanan and Sujatha, 2018). When using these algorithms, the outcome identification is not very accurate; however, these algorithms help identify and record observations about oblique data patterns, (Saravanan and Sujatha, 2018). Next to supervised machine learning algorithms (SMLA); the researcher, (Asongo et al., 2021) has

reviewed literature about unsupervised learning algorithms, when training data is not labeled or classed, USLA is utilized. The study of unsupervised learning examines how computers could generate a function that describes a structure that is concealed from not labeled input. System explores the data and can draw conclusions from corpus to describe concealed structures in not labeled input, but it does not decide the right outcome, (Asongo et al., 2021).

- **Reinforcement Learning**

Instead of receiving instructions or being told what to do, the learning algorithm chooses its actions based on the environment, (Yuxi 2018). This kind of learner can be used, for instance, in a game of chess. The agent learns its behavior through feedback received from the environment, (Yuxi 2018). After stopping a while, (Ravil, 2015) the researcher has reviewed literature about Reinforcement Machine Learning Algorithms (RMLA).

It focused on decision-making and action-taking in an environment to achieve a specific goal and operates through an agent that interacts with the environment by performing actions, receiving feedback as rewards or penalties & improving its behavior based on this feedback, (Ravil, 2015). At present, deep learning has a significant impact on both supervised and unsupervised learning; many researchers use deep learning to handle sentiment analysis. In this regard, the researcher reviewed so much stuff narrated in the next sub-topic, (Ravil, 2015).

2.5. Deep Learning

An artificial intelligence method called deep learning instructs computers to learn by precedent, exactly the way the human brain does by nature. Driverless cars can now interpret traffic signs, identify pedestrians, and even determine whether a driver is conscious or not to park the vehicle securely and prevent an accident thanks to the significant advancement of deep learning. It is also the driving force behind voice-activated devices like wireless speakers, TVs, tablets, and cell phones, (Tipirisetty 2018). Deep learning has received several attention lately, and given the situation, it is completely warranted given

that it is producing results that were previously thought to be unattainable, (Stuart et al., 2017).

Deep learning is the method by which a computer algorithm learns it's categorize images, text, or sounds. Deep learning algorithms may achieve high accuracy and sometimes outperform humans, (Tipirisetty 2018). Deep Learning is especially costly in terms of computing time and power because models with multi-layer neural network architectures are often fed and trained on massive amounts of categorized data, (Tipirisetty 2018). Deep learning achieves higher rates of recognition accuracy. This enhances consumer electronics' user experience and is necessary for safety-critical applications like autonomous cars. Certain tasks have been made possible by deep learning, like object recognition in images, where it outperform humans, (Stuart et al. 2017) and, deep learning are seen below in Figure 2-1.

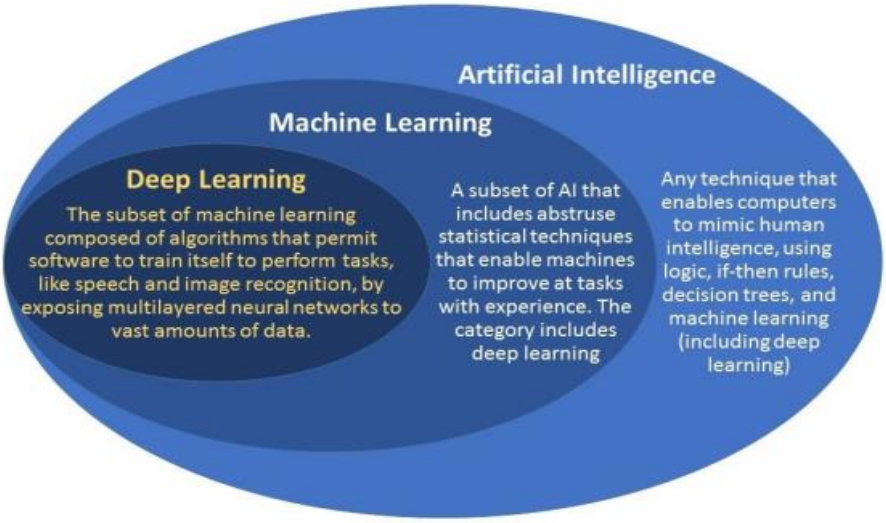

Figure 2-1 AI vs ML vs DL

Let us start from word embedding the researcher proposed, (Tao, 2017) which capture semantic relationships between words based on their context in large text corpora. It has purpose vectors to describe the meaning of words with same context and meaning are represented by vectors that are same to each other. Therefore, as the initial data processing layer in DL techniques, word embedding is regarded as a crucial component of sentiment analysis and other tasks involving NLP to competitive performance and improved accuracy,

(Tao, 2017). Word embedding is fundamental for deep learning models like convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Enabling them to understand and process text more effective like sentiment-specific word embedding technique & paragraph vectors have further enhanced the precision of sentiment classification tasks therefore, using embedding words in meaningful vector spaces. Deep learning models attain well performance in applications ranging from sentiment prediction in movie reviews to real-time analysis of social media content, (Qurat et al., 2017).

After winding up discussing word embedding, overall the researcher, (Manal, 2021) proposed an approach that integrates advanced NLP techniques with ML algorithms to effectively analyze sentiments specifically, in the context of online education during the pandemic, (Manal, 2021). The study showed, (Umarani et al., 2021) about an essential part of convolutional neural networks (CNNs), the convolution layer is designed to efficiently fetch and categorize features for applications like sentiment analysis. And makes use of a mathematical process which finds significant characteristics in the input data by applying filters or kernels and also the convolution layer is necessary for decreasing the dimensionality of the data while maintaining important information. It improves the model's capacity for learning and task performance and key properties like kernel size, stride, and padding determine the output dimensions also, variants of padding—zero or valid padding—help align the input matrix with kernel requirements, (Umarani et al., 2021).

The research paper reviewed and discussed about the document important components of Convolutional Neural Networks (CNNs) are the Fully Connected Layer, Pooling Layer, and Flattening Layer, (Umarani et al., 2021). All of the neurons from the preceding layer are linked in the Fully Connected Layer. It uses training data to categorize features into different classes, usually; the outcomes are normalized using the softmax activation function by using strategies like max pooling and average pooling to decrease computation and avoid overfitting. The Pooling Layer lowers the dimensionality of feature maps while maintaining important information finally, in order to feed the multidimensional input from earlier layers into the Fully Connected Layer for classification tasks, the Flattening Layer converts it into a single-vector array, (Umarani et al., 2021).

Moreover, RNNs are a type of feed-forward neural network, which processes sequential data according to the theory put forward by Elman and directed cycles also, the crucial transmission of activation function to the inward data arrangement, set RNNs apart from conventional feed-forward neural networks, (Jeffrey, 1990).Overall, the researcher (Umarani et al., 2021), implemented to both deep learning approaches like LSTM and CNN. Machine learning approaches like SVC, K-NN, Random forecast, Multinomial Naive Bayes, Bernoulli Naive Bayes, Logistic Regression was experimented for text datasets. Deep learning approaches higher achieved the optimal accuracy or performance than machine learning techniques for sentiment analysis, (Umarani et al., 2021).

The paper, (Eddy and David, 2019) highlighted the rapid advancements in artificial intelligence, particularly in areas like medical diagnosis and image recognition, facilitated by deep learning. The study has utilized various datasets, demonstrating high classification accuracy, particularly with binary classifications, and discusses the effectiveness of different sampling methods in training models, (Eddy and David, 2019). Further, “Deep Learning was firstly proposed by G.E. Hinton in 2006 and is the part of the machine learning process which refers to Deep Neural Network”, (Qurat et al., 2017, 425).

Deep learning, (Jianqing et al., 2021) going over its core ideas, well-known models, training methods, theoretical foundations and has a more fundamental function to stimulate the activity of the human brain. It describes how deep learning models intricate data connections by utilizing neural networks, such as convolutional and recurrent architectures. The function of over-parameterization, non-convexity in training, and the advantages of depth in networks both theoretically and practically are important subjects.

It (Jianqing et al., 2021) talks about improvements in methods like batch normalization, dropout, and stochastic gradient descent while highlighting issues like scalability, generalization, and training stability. So, it more highlighted the value of statistical research to expand knowledge of deep learning's potential and examines unsupervised models like auto encoders and Generative adversarial network. The unique attributes of deep learning, such as depth, over-parameterization, and algorithmic regularization, contribute to its success in tasks like image recognition and language processing. While significant progress has been made, the

document acknowledges that the theoretical understanding of deep learning remains incomplete and encourages further research in this area, (Jianqing et al., 2021).

In terms of precision, sensitivity, and specificity, the Deep Belief Neural classifier outperforms more conventional techniques like CNN, LSTM, and SVM, achieving a superior accuracy, (Sachin and TB, 2023).

Because deep learning can handle many features when working with unstructured data, it gets greater power and flexibility. Deep learning algorithms pass the data through several layers; each layer is capable of fetching features progressively and passes it to the next layer because of its exceptional capacity to analyze unstructured data and gradually learn characteristics over numerous layers. Deep learning has become more and more popular also, the availability of huge data sets and developments in high-performance computing are essential for its success, Amitha et al., 2021).

"The proposed model has been compared with the previous studies as those studies used CRF, SVM and additional traditional algorithms to perform sentiment analysis with a high price. However, the performance proves that the proposed model is reasonable and sufficient to enhance the accuracy in terms of emotion analysis", Qurat et al., 2017, 426). "Nowadays, deep learning is prosperous because of three main and important reasons, i.e., improved abilities of chip processing (GPU units), extensively lower expenditure of hardware and significant enhancements in machine learning algorithms", (Qurat et al. 2017, 425).

Algorithms

This paper has constructed to the model utilizing the supervised deep learning model is listed below, following the initial literature review and suggestions from the supervisor.

2.5.1. Convolutional Neural Networks (CNN) for Sentiment Analysis

The Research paper proposed to, Ashok et al. 2021; Yoon 2014) convolutional Neural Networks (CNNs) are a class of deep learning models specifically designed for processing structured grid data like images however, it have been effectively modified for text-based applications including sentiment analysis and in this context. CNNs are used to capture local patterns and n-gram features in textual data, making them effective for identifying

sentiment-bearing phrases or words. Ashok et al. 2021; Yoon 2014) by applying convolutional filters over word embedding, CNNs can fetch meaningful features, such as phrases or dependencies, which contribute to the sentiment of a text. Plus, pooling layers refine these features by selecting the most relevant ones, enabling the network to emphasis impactful portions of the input mostly. Further CNNs are computationally efficient, can handle variable-length inputs. They are particularly effective for tasks like sentence classification, where they often outperform traditional machine learning models for sentiment analysis. CNNs have demonstrated competitive performance across various datasets and languages, highlighting their ability to generalize across different text-based domains, Ashok et al. 2021; Yoon 2014).

2.5.2. Long Short-Term Memory (LSTM)

The researcher proposed to (Halit et al. 2020), one kind of recurrent neural network (RNN) for handling and learning from sequential input is the Long Short-Term Memory (LSTM) network. It solves the vanishing gradient issue that ordinary RNNs frequently encounter and long-range dependencies in sequential data can be captured by LSTMs since they retain a memory cell that can hold information for extended periods of time. The three main gates are used to govern this memory: the input gate regulates the amount of new data that is added to the memory; the forget gate chooses which data should be discarded; and the output gate chooses which data to output depending on the state of the memory, (Halit et al. 2020). Not only can LSTMs retain long-term dependencies, but they can also handle different sequence lengths and process input concurrently during training. Overall, LSTMs represent a powerful advancement in the field of deep learning, allowing for more nuanced and context-aware predictions in complex sequential tasks, (Halit et al. 2020; Ajay and Ausif 2019, Bekele, 2024). "The hidden state ht for LSTM is calculated using Equations (2.3), (2.4), (2.5), (2.6), (2.7), and (2.8) as follows:

"$i_t = (W_i X_t + U_i h_{t-1} + b_i)$ (2.3)"

"$f_t = (W_f X_t + U_f h_{t-1} + b_f)$ (2.4)"

"$O_t = (W_o X_t + U_o h_{t-1} + b_o)$ (2.5)"

"$C_t = \tanh (W_c X_t + U_c h_{t-1} + b_c)$ (2.6)"

"$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$ (2.7)"

"h_t = tanh(C_t) $*$ O_t (2.8) " , proposed to (Halit et al. 2020).

"where it, ft, and Ot are the respective input, forget, and outcome gates at time t; Wi, Wf, Wo, and Wc are weights that map the hidden layer's input to the three gates of input, forget, and outcome; Ui, Uf, Uo, and Uc are weights matrices that map the hidden layer's outcome to gates; and bi, bf, bo, and bc are vectors. Additionally, Ct and ht, respectively, are the results of the cell and the layer", (Halit et al. 2020).

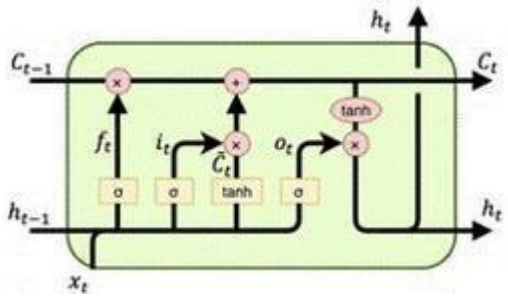

Figure 2-2 Long Short-Term Memory

2.5.3. Bidirectional Long Short-Term Memory (Bi-LSTM)

The researcher proposed to, bidirectional Long Short-Term Memory (Bi-LSTM) networks extend the capabilities of standard LSTM networks by processing data in two ways like forward and backward. The model is especially useful for sequential tasks where the context around a given input is critical because of its architecture, which enables it to concurrently gather information from previous and forthcoming contexts, (Bekele, 2024). A Bi-LSTM employs two distinct LSTM layers, one of which processes the input sequence forward (from start to finish) and the other backward (from end to start). Additional, in applications like natural language processing, where comprehending the entire context of words in a phrase may greatly improve the model's performance, this bidirectional technique is extremely advantageous for example, in sentiment analysis, a word's meaning may be influenced by the words that come before and after it. Therefore, all things considered, Bi-LSTMs provide an effective way to use contextual information in sequential data, improving the model's capacity to provide well-informed predictions based on a thorough comprehension of the input sequence., (Bekele, 2024; Zhiheng et al. 2015; Qi et al. 2022; Ashok et al. 2021).

2.5.4. Bidirectional Encoder Representations from Transformers (BERT)

The study proposed to, (Jacob et al. 2019) bidirectional approach allows BERT to capture deeper semantic and syntactic relationships within text, making it highly effective for natural language understanding tasks. Unlike traditional unidirectional models that process text in a left-to-right or right-to-left manner, BERT's bidirectional architecture provides it with the ability to understand nuanced language phenomena, such as polysemy and long-range dependencies in text. The two primary pre-training goals for BERT are Next Sentence Prediction (NSP) and Masked Language Modeling (MLM). These objectives enable it to generate general-purpose representations of text, which can be fine-tuned for a wide variety of downstream tasks such as sentiment analysis, question answering, and text classification. By leveraging its bidirectional encoding and extensive pre-training, BERT has set new benchmarks in various natural language processing tasks, (Jacob et al. 2019).

Moreover, the researcher implemented to, (Jacob et al. 2019) bidirectional Encoder Representations from Transformers (BERT) is a state-of-the-art natural language processing (NLP) algorithm designed to understand the context of words in relation to surrounding words in a sentence. Its main function to produce contextualized word embedding by considering the entire sentence, rather than processing it in a unidirectional manner as old-style models do. This bidirectional approach allows BERT to capture nuances and relationships between words more effectively, enhancing its ability to understand the meaning of words based on their context, (Jacob et al. 2019). It has two major activities which is the first one a pre-train BERT on vast amount of text data, Next Sentence Prediction (NSP), which aids the model in comprehending the link between sentence pairs. The second one is Masked Language Model (MLM), in which certain words are randomly masked and the model learns to predict them. (Jacob et al. 2019) after pre-training, BERT may be optimized for certain activities including named entity identification, query, response, and sentiment analysis, resulting in strong performance on a variety of NLP benchmarks. Because its capacity to offer rich contextual representations, BERT has established itself as a fundamental model in the area of natural language processing (NLP), resulting in notable progress across a range of applications and stimulating more investigation into transformer-based architectures, (Jacob et al. 2019).

2.6. Sentiment Analysis Application

When we have seen the application of sentiment analysis from research papers, it highlights the value of sentiment analysis for companies comparing goods and services and for people looking to base their decisions on public opinion, particularly given the growing amount of opinion data from social networks, (Bing, 2015). In addition, now a day the key applications area of sentiment analysis span diverse like, business and customer insights, social media monitoring, politics & public policy, healthcare, E-commerce, entertainment, finance, education, human resources, legal & security area, (Jonas and Olaf, 2019). If we take example of world which China, where social networking sites has emerged as the most widely used platform for citizens to express their views on governmental policies and to reveal corruption, and other crimes by public officials, this kind of surveillance is particularly prevalent., (Erik, 2013) .

Plus, the text identifies central challenges in sentiment analysis, such as defining what constitutes an opinion and the necessity for summarizing subjective views from several sources, (Bing, 2015). According to Su et al. (2013), Sentiment classification of these reviews would be helpful in business intelligence applications and recommender systems, where the reviews could be quickly summarized.

2.7. Related Works

“We highlight the latest studies on using the Word2vec model for sentiment analysis and its role in improving sentiment classification accuracy” (Samar and Arafat 2019, 39). “The proposed model makes use of an innovative combination of skip gram model of word2vec and term frequency inverse document frequency values to generate feature vector which can then be passed to an ordinary dense neural network with only one hidden layer of densely connected neurons.”, (Nithya et al., 2018, 120).

According to Qurat et al. (2017), highlighted latest studies regarding the implementation of deep learning models such as deep neural networks, convolutional neural networks and many more for solving different problems of sentiment analysis such as sentiment

classification, cross lingual problems, textual and visual analysis and product review analysis, etc. (P. 426).

According to Shilpa et al. (2021), also proposed a sentimental classification of a multitude of tweets here, we have used deep learning techniques to classify the sentiments of an expression into positive or negative emotions. The positive emotions are further classified into enthusiasm, fun, happiness, love, neutral, relief, surprise and negative emotions are classified into anger, boredom, emptiness, hate, sadness, worry. (P. 183).

According to Shilpa et al. (2021), experimented and evaluated the method using Recurrent Neural Networks and Long short-term memory on three different datasets to show how to achieve high emotion classification accuracy. The evaluation has showed that the system gains emotion prediction on LSTM model with 88.47% accuracy for positive/negative classification and 89.13% and 91.3% accuracy for positive and negative subclass respectively. So, a comprehensive evaluation demonstrates that the system improves emotion prediction on the LSTM model, (P. 183).

In nationwide context; empirical literatures according to Yeshewas (2023), models have analyzed some Facebook postings to understand socio political sentiments. We have conducted this study using a deep learning approach in the socio-political arena, utilizing the state-of-the-art in sentiment analysis on the Amharic language.

This study attempts to present the data taken from Fana Broadcasting Corporation's official Facebook profile using the Graph Application interface of Facebook social networks on immigration, war, and public relations issues in order to prepare the data for additional preprocessing. Because, more preparation of the dataset on the other domain improve the language, (Yeshewas, 2023).

According to Yeshewas (2023), next step of collect or obtain such socio-political sentiment data from the FBC using post_id all preprocessing operations like, tokenization, stop word removal, stemming of the sentences are conducted and using linguistic specialists in seven. The sentiment labels which are, positive, very positive, extremely positive, neutral, negative, very negative, and extremely negative the manual annotation of the sentence

fetched the data. It was included both the text file and based on sentiment labels different Emoji while, taking into account the impact of the most prevalent Emoji then, the feature extraction approach was constructed by the researcher using our custom vocabulary. The Scikit learn library feature mining classes, Countvectorizer, and TF-IDF Vectorizer, Yeshewas (2023).

According to Yeshewas (2023), gathered 11,000 reviews social political domain of FBC from Facebook page which are the fields of public relations, immigration, and war in order to assess the systems' performance. Therefore, this empirical literature test two feature mining and find that our model achieves well with Count Victories as it offers a straightforward representation of our data. The researcher evaluated our system training and validation accuracy using three experiments by changing training and testing split three parts that was first evaluation is 90%,10%: the second one is 80%,20%: the third and last was 70%,30%, the size of the corpus, the number of epochs and network layers.

According to Yeshewas (2023), the first technique achieved 90.1% average training accuracy & 90.1% average validation accuracy in the first experiment. The average training accuracy and validation accuracy of the second method are 82.4% and 40%, respectively. In the third trial, obtained 70.1 training accuracy and 40.1 validation accuracy by increasing the number of data sets to 1600 and adding five network layers so based on the results the findings indicate this research holds promise.

According to Fikiret (2023), employed deep learning techniques, including Convolutional Neural Networks (CNN), Bidirectional Long Short-Term Memory (Bi-LSTM), and a hybrid model combining CNN with Bi-LSTM to analyze and classify sentiments. The best-performing model was the hybrid CNN-Bi-LSTM model, which achieved an astounding accuracy of 91.60%.

According to Fikirte (2023), like a promising avenue for future research who underscores the importance of transitioning from binary sentiment analysis to a multi-class classification approach, enabling a finer-grained understanding of sentiments. The establishment of a standardized corpus for Amharic sentiment analysis emerges as a critical endeavor with broad applicability beyond politics, spanning domains like agriculture, industry, tourism,

sports, entertainment, and satisfaction analysis. The exploration of sarcastic comments in the Amharic language stands out as a promising avenue for future research.

Authors	Research Gaps
Fikirte et al., 2023	Binary sentiment analysis to a multi-class classification and gaps of Multi Domain.
Eyasu, 2022; Surafel, 2023 and Fikirte et al., 2023	Small datasets affect accuracy for deep learning Model.
Eyasu, 2022; Surafel, 2023	Most of Amharic sentiment research paper did not include different Emoji.
Nithya et al., 2018 For Amharic Sentiment to fill the	More advance neural network like CNN, LSTM, Bi-LSTM accuracy of sentiment prediction may increase much more.
Habimana et al., 2020; Eyasu, 2022	The gaps of Data Sparsely and noisy to full fill the gaps of Amharic sentiment more recommend transformers Multilingual BERT, which is underdeveloped model for Amharic sentiment. However, it is better contextual understanding for Amharic Sentiment.

Table: 2.1 Research Gaps

2.8 Research Gap

Despite the growing body of research on sentiment analysis, particularly for languages like Amharic, there are several challenges and gaps in current methodologies that this study aims to address. While prior works have successfully explored sentiment classification utilize deep learning models includes Word2Vec, TF-IDF, CNN, RNN, LSTM, Bi-LSTM, & hybrid models, (Qurat et al., 2017; Olivier et al., 2020; Shilpa et al., 2021; Fikirte 2023; Samar and Arafat 2019 & Nithya et al., 2018, 120). Most existing studies have focused on sentiment analysis in languages with more extensive linguistic resources, such as English. As a result, there is a lack of comprehensive studies that effectively tackle sentiment analysis for languages with relatively underdeveloped linguistic resources, like Amharic.

One notable gap in the existing literature is the limited exploration of more advanced models, such as BERT (Bidirectional Encoder Representations from Transformers), for sentiment analysis in languages with limited resources or datasets. While models like BERT have shown exceptional performance in sentiment analysis tasks for languages like English and Chinese, their application to Amharic, a language with rich morphology and unique syntactic structure, remains largely unexplored. Fine-tuning BERT for Amharic sentiment analysis presents an exciting opportunity to improve classification accuracy by leveraging the knowledge contained in pre-trained models while adapting them to the unique linguistic features of Amharic.

Additionally, while deep learning models include, Bi-LSTM and CNN are effectively applied to sentiment analysis in social networking text, including for Amharic, (Fikirte 2023). In order to collect both local and global contextual information, there is still a gap in the hybrid architecture that combines these models. Previous studies that used CNN for sentiment analysis often focus on capturing word-level or character-level features, while LSTM models excel in capturing sequential relationships.

Moreover, the issue of data sparsely and noise in Amharic sentiment datasets is another critical gap, limitation of research for Amharic Language, while there have been efforts to manually annotate and preprocess sentiment-labeled data from social media platforms like Facebook. There is still a lack of robust, publicly available sentiment-labeled datasets for

Amharic. Moreover, this scarcity, compounded by the challenges of handling complex features like Amharic Language study not included or not integrated with emoji-based sentiment cues, hinders the development of effective sentiment classification models. So, previous studies have not adequately addressed these challenges by utilizing advanced text preprocessing techniques to improve model robustness, (Yeshewas, 2023).

Therefore, this research, goals to fulfill these gaps employing a combination of state-of-the-art deep learning models (includes, fine-tuned BERT, Bi-LSTM, LSTM, and CNN) and use robust and labeled data preprocessing techniques, including tokenization, character-level normalization, and sentiment label mapping, to perform sentiment and emotion classification for Ethiopian Broadcasting Corporation Facebook posts of Amharic sentiment. This study has been model more accurate performance, context-aware sentiment analysis systems and addressing linguistic complexities and adapting technologies to low-resource languages like Amharic sentiment. This work has highlighted the importance of addressing linguistic complexities and adapting technologies to low-resource languages like Amharic.

Chapter Three

3. Design and Methodologies

3.1 Introduction

This section focuses on the methods utilized to build the dataset and apply techniques for achieving the research objectives related to Amharic sentiment analysis. The aim of the study is to apply deep learning models, analyze the sentiments of Facebook posts collected from the Ethiopian Broadcasting Corporation (EBC) and detect the overall sentiment polarity (strongly positive, positive, strongly negative, negative, and neutral). To achieve this, various preprocessing steps will be employed to prepare the data for sentiment classification.

3.2 Proposed Research Architecture

In this research work, employ a deep learning approach to develop a model for sentiment analysis of Amharic Facebook comments. The proposed architecture involves extensive data preprocessing to prepare the dataset for the primary task of sentiment classification. The dataset, comprising comments from Facebook, includes information such as comment content, sentiment labels, and other relevant metadata. The preprocessing pipeline involves several critical steps, such as data cleaning, where irrelevant information (e.g., URLs, mentions) is removed, and text normalization is applied to handle linguistic variations in Amharic text. Categorical sentiment labels are positive, strongly positive, negative, strongly negative, and neutral which are also transformed into numerical values to facilitate model training.

Tokenization is applied to convert text into sequences of words or tokens, and feature extraction is performed using methods like CountVectorizer to represent the text data to numerical format appropriate for deep learning models. In order to guarantee consistency in the dataset, feature scaling and normalization are employed, which enhances model convergence and prediction accuracy. To improve the structure of model training and assessment, the dataset is divided into training and testing sets.

Next to that, When have evaluated the sentiment, using four deep learning approaches for sentiment classification, including the Convolutional Neural Network (CNN), Bidirectional Long Short-Term Memory (Bi-LSTM), Multilingual BERT, and Long Short-Term Memory and Long Short-Term Memory (LSTM). When we have used the regularization techniques for deep learning in sentiment analysis to prevent overfitting, ensuring that the model generalizes well to unseen data including, early stopping, and five-fold cross-validation have been employed to assess the models' performance and generalization capability.

Additionally, Grid Search has been utilized to identify the optimal set of parameters for each model which grid search is a foundational technique in hyper parameter tuning, providing a straightforward way to optimize model performance in various deep learning tasks, including sentiment analysis further enhancing performance. Cross-validation is an essential tool in the deep learning workflow, providing valuable insights into model performance, robustness, and optimal hyper parameters. Its ability to enhance the generalization of models makes it a best practice for model evaluation and selection.

To comprehensively analyze performance, metrics such as accuracy, precision, recall, and F1-score were utilized. These metrics offer robust model performance indicators, capturing various aspects of prediction accuracy and reliability. By applying this suite of evaluation metrics, a comparative analysis is conducted to evaluate the efficacy of different models. The goal has been to identify the model consistently exhibiting the highest accuracy and F1 score, indicating strong classification performance and minimal prediction errors. This rigorous evaluation process selects the best-performing model based on its superior performance, ensuring reliable and accurate sentiment classification for Amharic Facebook comments. The overall proposed architecture for conducting this research has described in Figure 3.1 below.

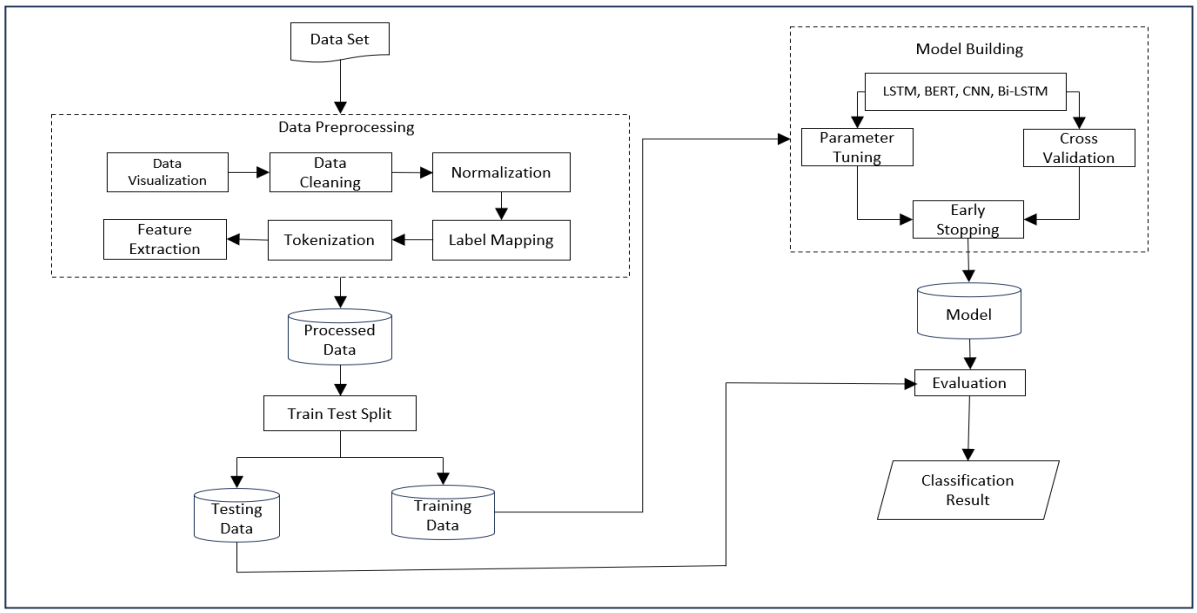

Figure 3-1 The proposed architecture

3.2.1 Data Collection

In this study, user-generated content has been collected from the official Facebook accounts of the Ethiopian Broadcasting Corporation (EBC), particularly focusing on posts related to ETV News. These posts encompass various categories such as culture, politics, social, the economy, and entertainment. Data collection has achieved through a combination of web scraping and direct access to data provided by EBC.

Web scraping has automated process used to mine data from the World Wide Web. For this research, web scraping has been employed to gather publicly available Facebook posts, comments, and associated metadata such as postdates and user reactions. This method allows for the systematic fetching of relevant data, facilitating large-scale collection from social media platforms. The web scraping process has been carefully tailored to fetch content from EBC's Facebook accounts, ensuring that the data collected aligns with the scope of the study's objectives. .

In addition to web scraping, the Ethiopian Broadcasting Corporation has provided an extensive dataset of Facebook posts. This dataset contains rich information curated by the corporation, adding significant value to the research. The web scraping has ensured a comprehensive and representative dataset, covering a wide range of topics and sentiments

expressed by users on social media. This dual approach strengthens the reliability of the dataset and ensures that it has well-suited for the sentiment analysis task at hand.

Figure 3.2 illustrates the sentiment distribution within the dataset through a pie chart titled "Sentiment Distribution." This chart encompasses a total of 22,000 labeled comments, categorizing sentiments as follows:

- **Positive Sentiments**: Representing the largest portion at **22.7%**, positive sentiments are depicted in **light red**. This indicates that a substantial segment of the analyzed content conveys a favorable tone.
- **Neutral Sentiments**: Also accounting for **22.7%** and shown in **light green**, neutral sentiments suggest a balanced perspective, indicating that a considerable amount of content lacks definitive sentiment.
- **Negative Sentiments**: Depicted in **light orange**, negative sentiments constitute **18.2%** of the dataset, reflecting a critical tone in a noteworthy portion of the comments.
- **Strongly Negative Sentiments**: Represented in **light purple**, strongly negative sentiments also account for **18.2%**, suggesting the presence of extreme negative opinions, though they are not overwhelmingly dominant. .
- **Strongly Positive Sentiments**: The smallest category, strongly positive sentiments, comprises **18.2%** and is highlighted in **light blue**, indicating that extreme positive opinions are less frequent compared to other sentiment categories.

According to this analysis, the dataset's sentiment distribution has been comparatively balanced. Both positive and neutral sentiments represent the largest proportions at 22.7%, while negative and strongly negative sentiments equally contribute 18.2% each. The rarity of strongly positive sentiments further underscores the overall moderate tone of the analyzed content, highlighting the complexities of sentiment expression in the dataset.

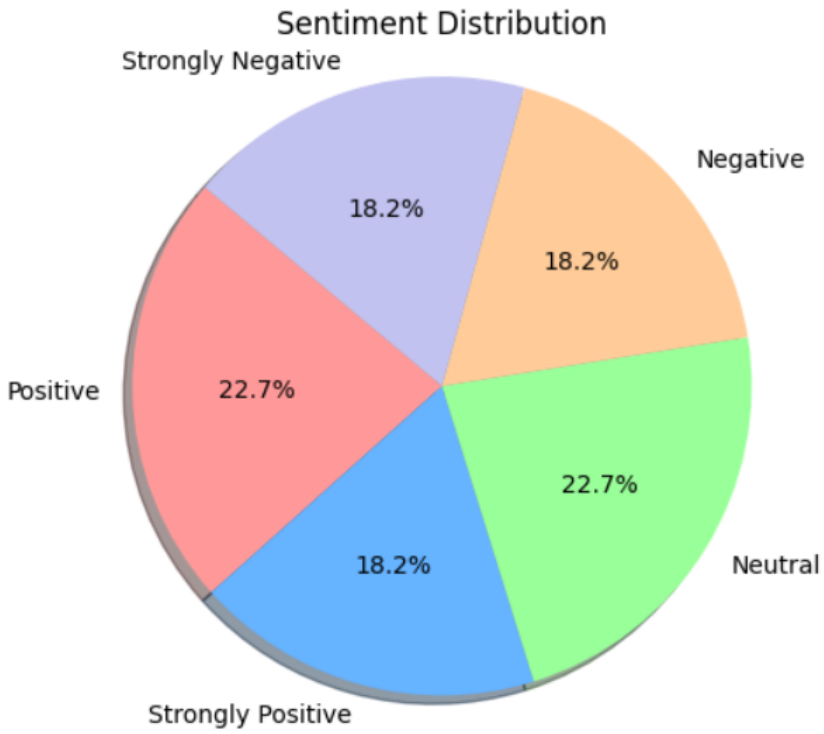

Figure 3-2 Data Visualization

This distribution reveals a balanced mix of sentiments, with positive, neutral, negative, strongly negative, and strongly positive sentiments each comprising a substantial portion of the dataset. The relatively equal representation of these sentiments indicates a diverse range of opinions, with no single sentiment type overwhelmingly dominant. This overall sentiment breakdown provides valuable insights for the subsequent stages of sentiment analysis and model development, enabling a comprehensive understanding of prevailing sentiment trends within the corpus.

3.2.2 Data Preprocessing

After gathering the data from the Ethiopian Broadcasting Corporation (EBC) and scraping Facebook posts, a preparation step has been followed to understand the types of data collected and identify. Data has been most appropriate and has the greatest relevance for sentiment classification. EBC specialists can provide valuable insights into the nature and characteristics of the provided data.

Data preprocessing has removed any inconsistencies or noise in the data that might negatively impact the sentiment analysis models. These inconsistencies may include missing data, noise (random errors or outliers), and other forms of variability. Before

loading the dataset into the working environment for analysis, the necessary software, packages have been installed. Ensuring data integrity has been a crucial step in any data analysis or deep learning tasks. Here's an overview of the processes involved in checking for completeness, redundancy, missing values, and the plausibility of attribute. Values have been verifying that all required data fields are present and populated, Identifying duplicate records or unnecessary repetition of data, Identifying duplicate records or unnecessary repetition of data, Identifying gaps where data points are absent and Ensuring that the values within each attribute fall within reasonable and expected ranges. In this research work, we have employed a deep learning technique to develop a model for sentiment analysis of Amharic Facebook posts. The proposed architecture has involved extensive data preprocessing to prepare the dataset for the primary task of sentiment classification. The dataset, comprising posts from Facebook, has included information such as post content, sentiment labels, and other relevant metadata. The preprocessing pipeline involves several critical steps, such as data cleaning; where irrelevant information (e.g., URLs, mentions) has removed. Text normalization has applied to handle linguistic variations in Amharic text. Categorical labels mapping (Positive, strongly positive, negative, strongly negative, and neutral) have also been transformed into numerical values to facilitate model training. We have employed the following elements during data preprocessing:

3.2.2.1 Data Cleaning

The cleaning data procedure has been essential to removing inconsistencies, noise, and irrelevant elements from the Amharic Facebook dataset to prepare it for sentiment analysis. The cleaning began with handling missing and redundant data. Rows containing null values have been dropped to ensure that the dataset had no incomplete entries that could skew the analysis. Duplicates have been also removed by keeping only the first occurrence of Facebook posts with identical Facebook post ID values, thus eliminating data redundancy.

After addressing structural inconsistencies in the data, the next step involved filtering out posts with a "mixed" sentiment label. Only posts with clear sentiment categories—strongly positive, positive, strongly negative, negative, and neutral—have been retained, streamlining the sentiment classification process. This filtering has been

crucial for reducing ambiguity and improving the performance or precision of the sentiment classification method.

Text Cleaning:

Moreover, structural cleaning, a series of text-cleaning operations have been performed to prepare the textual content of the Facebook posts for further processing:

1. **Removing URLs**: URLs that are not necessary to the sentiment of the text have been removed using regular expressions. This has helped to eliminate irrelevant information that could introduce noise into the analysis.

2. **Removing Mentions**: Mentions of other Facebook users (e.g., @username) have been also stripped from the text, as they did not carry meaningful information related to sentiment. This step has ensured that the analysis focused solely on the content of the posts.

By applying these text cleaning steps, the complexity of the text has been reduced while retaining its sentiment-bearing content, making it suitable for further processing, such as tokenization and feature extraction.

3.2.2.2 Character-Level Normalization

Given the linguistic complexities of the Amharic language, a character-level normalization process has been applied to manage the variations in how certain Amharic characters are written. Amharic, being a Semitic language with a rich set of characters, often faces challenges like different representations of the same word or phrase due to the various ways certain characters have been written. These variations can negatively impact the performance of machine learning models if not handled properly, as the same word might be interpreted differently by the model.

For example, a word like "sun" can be written in multiple forms in Amharic, such as **ጸሀይ** and **ፀሐይ**, which have been essentially the same word but written using slightly different characters. If these variations have been not normalized, the model might treat them as distinct words, leading to a loss of information and lower performance in tasks like sentiment analysis.

To address this, a custom character normalization function has been developed to ensure that different forms of characters are mapped to a single, consistent representation. The normalization process targeted specific characters known to have multiple forms or common misspellings, converting them to a standard version. This step has been crucial for reducing the variability in the text data while preserving the linguistic richness of the Amharic language.

This character-level normalization has been applied to all textual data in the Facebook posts, ensuring consistency across the corpus & improving the model's performance overall across different variations of the same word. By resolving these inconsistencies at the character level, the sentiment analysis model could focus on the underlying sentiment of the posts rather than being distracted by superficial variations in spelling. Algorithm 3.1 depicts how char-level normalization works.

Input: Cleaned dataset

Outcome: Normalized_dataset

Begin:

Normalized_data computes char-level normalization on the Dataset

Return Normalized_dataset

End

Algorithm 3-1: Pseudo-code of char-level normalization

3.2.2.3 Sentiment Label Mapping

To streamline the sentiment classification process, the sentiment labels in the dataset have been mapped from their original categorical form (strongly positive, positive, strongly negative, negative, and neutral) into numerical values. This step has been essential because machine learning models, particularly deep learning models, typically work more effectively with numerical data rather than textual or categorical data.

In this process, the sentiment categories have been assigned specific integer values:

- **Positive sentiment** has been mapped to **1**.
- **Strongly Positive sentiment** has been mapped to **3**.
- **Neutral sentiment** has been mapped to **0**.

- **Negative sentiment** has been mapped to **2**.
- **Strongly Negative sentiment** has been mapped to **4**.

This transformation has allowed the deep learning model to process the sentiment labels as integers, making it easier for the model to classify each Facebook post based on its sentiment. By converting the sentiment into numerical values, the model could use these integers during training to learn the relationships between the textual content and the corresponding sentiment class. During evaluation and prediction, the model would then outcome these numerical values, which could easily be mapped back to their respective sentiment categories for interpretation.

This sentiment label mapping has simplified the classification task and has enhanced the overall efficiency and accuracy of the sentiment analysis model. Algorithm 3.2 depicts how label mapping works.

Input: normalized_dataset

Outcome: labelled_mapped _dataset

Begin:

Labelled_mapped_dataset Apply label mapping

Return labelled_mapped _dataset

End

Algorithm 3-2: Pseudo-code of label mapping

3.2.2.4 Tokenization

In this research work, we have applied Tokenization as a crucial step in text preprocessing, particularly for sentiment analysis, as it involves breaking down the textual data into manageable units. In this research, each Facebook post has been split into individual words (or tokens), a process that enabled the model to analyze the text at a granular level.

After splitting the posts into words, each word has been mapped to its corresponding numerical representation using a **predefined vocabulary**. This vocabulary contained a unique index for each word in the dataset, and it has been developed based on the training data to ensure that the most frequently occurring words have been well-

represented. The transformation of words into numerical values has allowed the deep learning model to interpret and processed the text, as it cannot directly work with raw words or characters.

In instances where a word has been not found in the predefined vocabulary, a special token, known as **<unk>** (unknown), has been assigned. This ensured that all tokens, even rare or previously unseen words, were accounted for during the model's processing. By converting text into sequences of integers, tokenization facilitated the fetching of significant patterns from the data, allowing the model to effectively learn the relationships between the words in each post and associated sentiment.

This process has been critical in transforming the raw text into a setup appropriate for training and evaluation by the deep learning models, enabling accurate and efficient sentiment classification of Amharic Facebook posts. Algorithm 3.3 depicts how tokenization works.

Input: labelled_mapped _dataset

Outcome: tokenized_dataset

Begin:

tokenized_data □ compute tokenization on Dataset

Return Tokenized_dataset

End

Algorithm 3-3: Pseudo-code of tokenization

3.2.2.5 Feature Extraction

For the sentiment analysis of Amharic Facebook posts, feature extraction has been a vital process that transformed unprocessed data into a numerical form appropriate for deep learning approaches. This research has employed CountVectorizer to fetched word-level n-grams, specifically up to tri-grams, allowing the model to capture crucial linguistic patterns, word order, and context. N-gram, which has been a contiguous sequence of 'n' words, provides insights into sentiment by revealing how combinations of words can convey nuanced meanings, particularly in a morphologically rich language like Amharic. The CountVectorizer has been created a feature matrix where each row represented a Facebook post and each column indicated the distinct n-gram, with values reflecting their frequency of occurrence. To reduce complexity, the matrix

has been limited to 1,000 features, focusing on the most significant n-grams. This structured numerical representation effectively captures both individual words and their sequences, enabling models for deep learning to learn from the data linguistic patterns & context, ultimately improving their sentiment classification accuracy.

3.2.3 Train-Test Split

An essential step in preparing the data for sentiment analysis has been the train-test split. In this process, the corpus has been divided into three distinct data frames: df_train, df_test, and df_valid, which are training, testing, and validation corpuses, respectively. To ensure the quality of the data, any rows containing mixed sentiment have been removed from both the training and testing datasets.

Then, the corpus has been divided, with 20% set aside for testing and 80% going toward training. Following this, feature extraction is conducted using the CountVectorizer from Scikit-Learn, which transforms the cleaned text data into numerical feature vectors. The maximum number of features has capped at 1000, and n-grams have utilized to capture more contexts within the text.

Additionally, labels mapping—positive, strongly positive, negative, strongly negative, and neutral— have been encoded into numerical values: strongly positive as 3, positive as 1, negative as 2, strongly negative as 4, and neutral as 0. This encoding has been crucial for the subsequent model training phases. The resulting datasets for training are called X_train and Y_train and also, for testing data are called X_test and Y_test, to provide necessary inputs for the deep learning models, allowing for effective training and evaluation of performance. This structured division ensures which has been robust and accomplished overall well to unseen data. Algorithm 3.4 depicts how to train-test split works.

Input: Processed_data

Outcome: Train-test_Data

Initialize Train_size to 80%

Initialize Test_size to 20%

Begin:

Taining_data,Test_data ← Train-test split on Processed_data

Return Training_Data, Test_Data

End

Algorithm 3-4: Pseudo-code of train-test split

3.2.4 Hyper parameter Tunning

To explore various combinations of hyper parameters during grid search, create a hyperparameter grid for each model. The ideal set of hyper parameters for each model has been discovered using grid search.

- Perform Grid Search: The 'GridSearchCV' function runs a grid search for each model. To find the model with the finest performance, this has been trained and assessed using various combinations of hyper parameters.

3.2.5 Training with Cross-Validation

Cross-fold validation has been the process that increases our model's correctness. Instead of choosing a specific subset of training & the remaining fraction of testing, it has used the complete corpus for both training and testing. The corpus has been often separated into five equal sets or folds for a five-fold validation, with four sets have used for training and one set utilized for validation testing for each iteration. The result of this operation has been displayed after four (k-1) repetitions or all conceivable combinations. Algorithm 3.4 depicts how cross-validation works.

Input: Grid search parameter P, Training Data, Test Data

Outcome: Final_Trained_Model, Prediction Result

Initialize Cross Validation CV to 5

Begin:

Final_Trained_Model← GridSearchCV (Training_Data, P, CV)

Final_prediction ← evaluate Final_Trained_Model on (Test_Data)

Return Final_Trained_Model, Prediction_Result

End

Algorithm 3-4: Pseudo-code of cross-validation

3.3.6 Applying Deep Learning Models

After preprocessing step our corpus which has been separated into three are called training, validation, and testing corpus next to this, we have implemented a deep learning algorithm to classify the sentiments of Facebook posts. In this research, it have been focused on four deep learning architectures which are Multilingual BERT, Bidirectional Long Short-Term Memory (Bi-LSTM), Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN) of acting the training phase on the training corpus are following mention below. After models have trained or learnt the algorithm, models have evaluated on the testing corpus and we have employed various metrics such as, accuracy, precision, recall, and F1-score, to evaluate the performance of the sentiment classification techniques.

Input: Train_datset,Test_dataset

Outcome: Predicted values

Begin:

Initialize the individual deep learning models:

Model1 <-ConvolutionalNeuralNetwork

Model2 <- MultilingualBERT

Model3 <- LongShortTermModel

Model4 <- Bi-LongShortTermModel

Fit the individual deep learning models on the training data

Predict the target variable for the test data using the deep learning models

Predicted_values <- DeepLearningModel.predict(Test_dataset)

Return Training_Data, Test_Data

End

Algorithm 3-5: Pseudo-code of deep learning models

3.2.7 Evaluation Measurement

We have been employed various metrics which are accuracy, precision, recall, and F1-score, to evaluate the performance of the sentiment classification techniques.

The experimental findings outcome by the proposed sentiment analysis of research has been assessed using the performance metrics the following listed. A Confusion matrix has been a performance measurement tool used in classification problems to evaluate the

accuracy of a model. It has provided a comprehensive breakdown of the model's predictions compared to the actual outcomes. So, it helps to visualize the performance of an algorithm and identify areas where it may be making errors. It has been shown as a table with rows represents the predicted outcomes which are positive, strongly positive, negative, strongly negative, and neutral. Columns representing the actual values are 'true' and 'false'. The outputs of a classifier have been presented in the table as the number of true positive (TP), false positive (FP), false negative (FN), and true negative (TN).

As seen in table 4 below, a confusion matrix has been a 2x2 matrix (two class) for the binary organization with column side (actual values) and on the other side of row (predicted values) here.

Table 3-1 Confusion Matrix

Labeled	Predict Positive	Predict Negative
Actual Positive	TP	FP
Actual Negative	FN	TN

Using confusion matrix, to evaluate the outcome of sentiment analysis which used accuracy, precision, recall, and F1-score the key of metrics.

Total percentage of Accuracy has been clearly declared to positive and negative prediction and mentioned below, (Alexander 2009).

$$Accuracey = \frac{TP+TN}{TP+TN+FP+FN} \quad (3.1)$$

When working with unbalanced data, accuracy performance measurements can make all the difference. We have computed the accuracy, confusion matrix, precision, recall, and F1 score to offer a better intuitive understanding of the prediction outcomes.

When we have shown each define the meaning of metric for actual and prediction of sentiment analysis to detail illustrated here. The model that accurately predicts the positive class where both the forecast and the actual are positive is known as True

Positive (TP). Conversely, a True Negative (TN) model accurately predicts the negative class (both the forecast and the actual are negative). A False Positive (FP) model is one that predicts the negative class incorrectly (actual-negative, predicted-positive). Another name for FP is a TYPE we mistake. Lastly, a False Negative (FN) occurs when the model forecasts the positive class (actual-positive, predicted-negative) incorrectly. Another name for FN is a TYPE II mistake.

Precision, recall, and F-score are further performance measures that may be computed with the aid of TP, TN, FN, and FP. Precision uses equation 3.2 to determine what proportion of all the positive predictions are actually positive. To determine the precision value falls between zero and One.

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

Determine the percentage of recall measures that have been anticipated to be positive using equation 3.3 out of the total positive so; it is comparable to the TPR, or true positive rate.

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

It will be challenging to determine which model or system is superior when comparing them (high precision and low recall or vice versa) and the result of a metric that incorporates both of these ought to exist, (Alexander 2009).

Equation 3.4 indicates that the F1 Score is the harmonious average of precision and recall. It has considered both false positive & false negative. Consequently, it operates effectively with an unbalanced corpus. Recall and precision are assumed equal weight in the F1 score.

$$F1\ score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \tag{3.4}$$

Chapter Four

4. Implementation, Experimental Result and Discussion

4.1 Introduction

This section discusses and compares the experiments have conducted on sentiment analysis using the Amharic datasets from the Facebook account, concentrating on the algorithm of deep learning models.

4.2 Development and Experimentation Tools

The development languages and experimentation tools used to put the suggested model into practice and assess and it has been described. In this research work, we have used Anaconda while Python alone could accomplish the tasks. Anaconda has been chosen for its robust package management, ease of creating isolated environments, extensive library support, and consistent cross-platform behavior. These aspects collectively facilitated a smoother workflow, efficient collaboration, and ensured the reproducibility of our research, contributing significantly to the quality and reliability of our findings. We have utilized several tools and libraries to develop various components. Table 4.1 has discussed the languages, libraries, and tools with their corresponding descriptions.

No.	Tool	Description	Version
1	Anaconda	Anaconda is an R programming language, to organize and deliver packages. We have applied it in our experiment to install and run all libraries.	4.12.0
2	Python	Python is a high-level, has interpreted programming language. We have applied it to write our code in our experiment.	3.12.7
3	Matplotlib	Matplotlib is a Python figure plotting library. It has used to depict the experimentation plot.	3.7.1
4	NumPy	NumPy is a Python library that works on arrays.	1.24.3
5	Pandas	Pandas is a manipulation library for Python, providing	1.5.3

		powerful data structures like Series and Data Frames.	
6	Sklearn	Scikit-learn provide simple and efficient tools for data mining and data analysis.	1.2.2
7	Tensor flow	TensorFlow provides a flexible ecosystem of tools, libraries, and community resources that facilitate the development of machine learning applications.	2.11.0
8	Jupyter Notebook	Jupyter Notebook that allows users to create and share interactive documents containing live code, equations, visualizations, and narrative text.	6.5.2
9	Microsoft Visio	It is used to make basic or complex diagrams, and we used to make our model.	2019

Table 4-1 List of Development and Experimentation Tools

4.3 Data Preprocessing of implementation Season

Several procedures have been performed to clean and prepare the corpus throughout the data preparation phase of sentiment analysis work. When we have been begun to data preparation step, loading Excel files containing Amharic comments labeled corpus with various sentiments (positive, strongly positive, negative, strongly negative, neutral, and mixed) into Pandas Data Frames from Google Drive. These files have represented training, testing, and validation datasets. After loading, the number of occurrences of each sentiment has been visualized using bar plots. But, like "mixed" comments have to be removed from datasets based on their own sentiment labeling to simplify the classification problem and omit or leave only ‘positive’, ‘strongly positive’, 'negative,' ‘strongly negative’, and 'neutral' sentiments in datasets.

The data has been further refined through a function, clean_df, which removed rows containing null values, eliminated duplicate comments based on comment IDs, and reset the index of each data frame. Next, a text cleaning function, clean_text, has been applied to the comment columns. This function has been transformed the text to lowercase and has been removed unnecessary elements such as URLs, and user mentions. We have configured the

cleaning options to ensure. These operations have been performed uniformly across all comments in the datasets.

A unique and big challenge of Amharic text has been character-level mismatches, where multiple representations of the same character exist and have to address it. Therefore we have developed a character normalization function, normalize_char_level_missmatch, which has standardized different variations of Amharic characters to their base forms. This function is also called addressed specific issues such as labialized characters by normalizing words like **በልቱዋል** and **በልቱአል** to **በልቷል**, and a word like "SUN" can be written in multiple forms in Amharic, such as **ጸሀይ** and **ፀሐይ**, which are essential the same word but written form using slightly different characters. After applying this function, the cleaned DataFrames have been saved back to Google Drive for further use.

To prepare the text data for model input next step, a CountVectorizer has been employed to convert the cleaned comments into numerical feature vectors based on word n-grams (ranging from 1 to 5), with a maximum of 1,000 features. The sentiment labels have been mapped to numerical values (strongly positive: 3, positive: 1, negative: 2, strongly negative: 4, neutral: 0) to prepare the data for supervised learning.

To have been built a vocabulary from the training data by counting the frequency of tokens and assigning an index to each token that appeared more than once. The comments have been then tokenized and encoded based on this vocabulary, with unknown tokens replaced by a special <unk> token and padding handled with a <pad> token.

The following code snippets in figure 4-1 illustrate key aspects of the preprocessing:

```python
# Define the clean_df function
def clean_df(df):
    df = df.dropna()
    df = df.drop_duplicates(subset='tweet_id', keep="first")
    df = df.reset_index(drop=True)
    return df

# Define the clean_text function
def clean_text(row, options):
    row = row.lower() if options['lowercase'] else row
    row = re.sub(r"(?:\@|https?\://)\S+", "", row) if options['remove_url'] else row
    row = re.sub(r"@[A-Za-z0-9_]+", "", row) if options['remove_mentions'] else row
    return row

# Character-level normalization
def normalize_char_level_missmatch(input_token):
    rep = re.sub('[ሃኀኃሐሓኻ]', 'ሀ', input_token)
    rep = re.sub('[ሑኁዅ]', 'ሁ', rep)
    # Other normalization rules...
    return rep

# Tokenization and encoding
def build_vocab(sentences, min_freq=1):
    counter = Counter(token for sentence in sentences for token in sentence.split())
    vocab = {token: idx + 2 for idx, (token, count) in enumerate(counter.items()) if count >= min_freq}
    vocab['<pad>'] = 0
    vocab['<unk>'] = 1
    return vocab

def tokenize_and_encode(text, vocab):
    return [vocab.get(token, vocab['<unk>']) for token in text.split()]
```

Figure 4-1 Key aspects of the preprocessing

My preprocessing pipeline involved several steps to enhance the dataset's quality for model training. First, we have removed rows with missing values and eliminated duplicate entries based on the comment_id column to ensure each post has been unique and complete. Next, we have performed a series of text cleaning operations: URLs have been removed using regular expressions to prevent biases toward specific websites, mentions of other users (e.g., @username) have been excluded to focus on comment content, has been stripped out to avoid ambiguous sentiment indicators or cultural nuances associated with them in Amharic. Additionally, we have applied character-level normalization to address regional or typographic variability in Amharic characters. For instance, variants such as [ሃ , ኀ , ኃ , ሐ, ሓ, ኻ] have been mapped to a single standardized form, **‘ሀ’**, to help the model better interpret word meanings. Finally, each sentence has been tokenized into words, and we have built a vocabulary where each word has been assigned a unique integer identifier, with special tokens for unknown words (<unk>) and padding (<pad>) added to support input processing for the model.

```
Cleaned Training DataFrame:
   Unnamed: 0            tweet_id sentiment  \
0           1  1213011490372038912   neutral
1           3  1213764224356421888   neutral
2           5  1212723914494877952   neutral
3           6  1201007355607040000   neutral
4           7  1213183031638405120   neutral

                                                tweet
0      ልዩ የተፈጥሮ ገፅታ \n *****\n\nየምስራቅ አፍሪካ የውሃ ማማ ጮቄ
1  ብ/ጄ አሳምነው ጽጌ ከምክትል ጠ/ሚ ደመቀ ጋር በሱዳን ጉብኝት አልበሸር ...
2  @EthioReporter The only part I like....."ትልልቅ...
3  @Abione_Bekele 😊 እሺ ወንድሜ እተከላለሁ ማኛ እኮ ናት ወዳታለሁ...
4  @fitih11 ይቺን አውቆ ነው ዶ/ር አብይ ክርስቲያን የሆነዋ.........
```

Figure 4-2 Raw Amharic text data (Input)

After applying these preprocessing steps, the dataset has been significantly cleaner and more consistent, as shown in Figure 4-3. In the cleaned dataset, text has been normalized, unnecessary information has been removed, and comments are ready for tokenization and input into the model.

```
Cleaned Training DataFrame:
0                        አባቴ የሚያስፈራኝ ኮሮና አይደለም ጌታቸው ረዳ ነው።።
1          በክትባት እድገት ውስጥ ያሉ እድገቶች አስደናቂ የሳይንስ ውጤቶች ናቸው።።
2      እኔ ቤተሰብዎን እየረገምኩ ነው ይህ የእርስዎ ቅጣት ነው አሁን ለመኖር አ...
3              ይህ አዲስ ፕሮግራም ለማህበራዊ ፍትህ ትልቅ እርምጃ ነው - ግሩም!
4                                    ጌታ ሆይ  አሁን የምር ልሄድ ነው
Name: tweet, dtype: object
```

Figure 4-3 Cleaned Amharic Text Data (Outcome)

4.3.1 Data Labeling for Implementation Season

Data labeling is an essential step in developing a strong sentiment analysis model since precise labeling of sentiment data improves the model's capacity to comprehend and categorize the underlying emotions portrayed in text. For Amharic sentiment analysis, we have followed a structured approach to classify each data instance into five map labeling which are ‘positive’, ‘strongly positive’, ‘neutral’, ‘negative’, and ‘strongly negative’. This categorization framework ensures that each labeled sentiment accurately reflects the nuanced expression of opinions in the Amharic language.

The labeling process for Amharic sentiment analysis involves a systematic approach to ensure accurate and contextually relevant sentiment categorization. Each sentence or phrase is carefully reviewed in its context, with annotators identifying linguistic indicators such as adverbs (ተውላጠ ቃላት) and adjectives (ቅጽሎች), which help determine the

sentence's sentiment and intensity. This contextual understanding of linguistic characteristics is crucial for capturing the nuances of each sentiment. According to these identified indicators, the sentence is then assigned to one of the five sentiment categories which are ‘positive’, ‘strongly positive’, ‘neutral’, ‘negative’, and ‘strongly negative’. This structured approach ensures that the sentiment labels are both accurate and reflective of the sentence's intended emotional tone.

The labeling workflow for Amharic sentiment analysis begins with manual annotation, where each sentence has been reviewed by trained annotators who Amharic linguistic professionals assign the most fitting sentiment label based on predefined criteria. These annotators have been skilled at identifying key phrases and contextual signals that align with each sentiment category. Multiple annotators review each sentence to ensure consistency, and any disagreements are resolved either through a voting system or by an expert review to achieve consensus.

The preprocessing steps enhanced the data quality and reduced noise that could negatively impact model performance. By standardizing the text and removing extraneous elements, we have ensured that the model focuses on the core content of each comment, thus improving its ability to identify sentiment correctly. This data preprocessing pipeline ensured that the textual data has been cleaned, standardized, and appropriately transformed into a structure appropriate to deep learning approach, specifically for sentiment analysis of Amharic Facebook posts.

4.4 Train-Test Split

The study, we have implemented a systematic approach for data preparation and come to splitting corpus using Python's Pandas and Scikit-learn libraries. When we started by loading the cleaned training dataset from an Excel file and stored it on Google Drive. To ensure a balanced representation of sentiment classes, the filtered corpus has been split into two parts: a training (80%), and temporary (20%) for testing and validation. The temporary set has been split into validation and test datasets, each comprising 10% of the original dataset. We have applied stratified sampling to maintain the distribution of sentiment classes across all splits as shown in Figure 4-4.

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

# Set the file path to your Excel file in Google Drive
url = "/content/drive/MyDrive/SA/Training_data_cleaned.xlsx"

# Load the Excel file into a DataFrame
df_training = pd.read_excel(url)

# Filter out the 'mixed' sentiment
df_filtered = df_training[df_training["sentiment"] != "mixed"]

# Split data into training (80%) and temporary (20%) for test and validation
train_data, temp_data = train_test_split(
    df_filtered,
    test_size=0.2,
    stratify=df_filtered["sentiment"],
    random_state=42
)

# Split the temporary data into validation (10%) and test (10%)
validation_data, test_data = train_test_split(
    temp_data,
    test_size=0.5,
    stratify=temp_data["sentiment"],
    random_state=42
)

# Check the distribution in each split
print("Training set sentiment distribution:\n", train_data["sentiment"].value_counts(normalize=True))
print("Validation set sentiment distribution:\n", validation_data["sentiment"].value_counts(normalize=True))
print("Test set sentiment distribution:\n", test_data["sentiment"].value_counts(normalize=True))
```

Figure 4-4 Train - Test split

Before we have been coming to perform sentiment analysis, where the sample sentiment of each sentence has been categorized into five map labeling which are ‘positive’, ‘strongly positive’, ‘Negative’, ‘strongly Negative’, and ‘neutral’. Training, validation, and test are the three subsets into which the dataset has been divided. The data split follows 80-10-10

proportion, with 80% has utilized for training, and 10% each has reserved for validation and testing. The sentiment distributions within these subsets are as follows:

Training Set Sentiment Distribution

The training positioned, which makes up 80% of the entire dataset, has the following sentiment distribution:

- **Neutral**: 22.73%
- **Positive**: 22.73%
- **Negative**: 18.18%
- **Strongly Positive**: 18.18%
- **Strongly Negative**: 18.18%

These proportions indicate a fairly balanced distribution of sentiment categories, with only slight variation across the different labels. During training, the balance ensures that the algorithm has revealed a varied set of sentiment expressions, which is critical for developing robust sentiment classification models.

Validation Set Sentiment Distribution

The validation set represents 10% of the total data and has used for hyperparameter tuning and model evaluation during trainin**g**. The sentiment distribution in poisoned is look like this:

- **Strongly Positive**: 18.18%
- **Strongly Negative**: 18.18%
- **Neutral**: 22.73%
- **Positive**: 22.73%
- **Negative**: 18.18%

The validation set demonstrates a distribution similar to that of the training set, ensuring consistency and fairness in the evaluation process. The balanced nature of the sentiment distribution allows for reliable model performance evaluation across different sentiment classes.

Test Set Sentiment Distribution

The ultimate performance of the trained algorithm on unseen data has evaluated using the test positioned, which likewise makes up 10% of the whole corpus. The test set's sentiment distribution looks like this:

- **Positive**: 22.73%
- **Neutral**: 22.73%
- **Strongly Positive**: 18.18%
- **Strongly Negative**: 18.18%
- **Negative**: 18.18%

While the test set is similarly balanced, it contains a slightly higher proportion of positive sentiment instances relative to the other categories. This distribution has ensured that the test has positioned as representative of the general sentiment distribution in the corpus.

The model has been trained on a suitably big and varied training set confirms the 80-10-10 split, while the validation and test sets retain a balanced representation of sentiment labels, mitigating the risk of class imbalance and providing a reliable basis for model evaluation. The consistent proportions across all three sets allow for accurate performance metrics, helping confirm the model overall well to different sentiment categories. Furthermore, the balanced distribution of sentiment categories into two which for the validation and test positioning confirm that the algorithm has been evaluated equitably across the range of possible sentiment labels, facilitating a robust and comprehensive assessment of its performance.

4.5 Amharic Sentiment Analysis Using Deep Learning

After the entire corpus has been prepared the preprocessing steps of cleaned and prepared data to modeling function, we have come up with a constructed model using four different deep learning techniques which are Bi-LSTM, LSTM, CNN, and multilingual BERT for sentiment analysis.

4.5.1 LSTM for Amharic Sentiment Classification

Model has been built LSTM-centered approach for sentiment classification of Amharic Facebook posts. We have begun by loading and preprocessing the dataset with pandas, focusing on two columns: sentiment (the sentiment label) and comment (the text content). First, the data has been cleaned by removing missing values and duplicates to ensure high-quality input (using df.dropna () and df.drop_duplicates (subset='comment', keep="first")). Sentiment labels have been then mapped to numeric values for model compatibility. Each label has been assigned a corresponding integer, with neutral as 0, positive as 1, negative as 2, strongly positive as 3, and strongly negative as 4.

((df['sentiment'] = df['sentiment'].map(label_mapping)). This mapping allowed the model to classify inputs across five distinct sentiment classes.

For the vocabulary, manually a dictionary has been built by tokenizing the text and including only words with a minimum frequency of one (Counter (token for sentence in sentences for token in sentence.split ())). The tokens have been then encoded into numeric sequences using this vocabulary, with special tokens for padding (<pad>) and unknown words (<unk>). These sequences have been used to create PyTorch datasets for training and testing, which have been then loaded with batching and padding for efficient processing.

This model architecture is an LSTM-based classifier, has designed with an embedding layer (for tokenized input), a hidden LSTM layer, and a fully connected outcome layer for class predictions. The outcome dimension has been set to five to correspond with the sentiment classes. The Adam optimizer (optim.Adam(model.parameters(), lr=1e-5)) has been used to optimize the model after it has been weighted based on the class distribution and trained with cross-entropy loss.

When training the LSTM model for sentiment analysis on the Amharic Facebook corpus, we have monitored training and validation losses over 100 epochs, with early stopping activated at epoch 10 to prevent overfitting. The training loss decreased from 1.5613 in the first epoch to 1.1089 by epoch 10, while the validation loss initially dropped from 1.4714 to 1.1440, reflecting gradual performance improvement until early stopping has been triggered. This training progression is seen below in Figure 4.5.

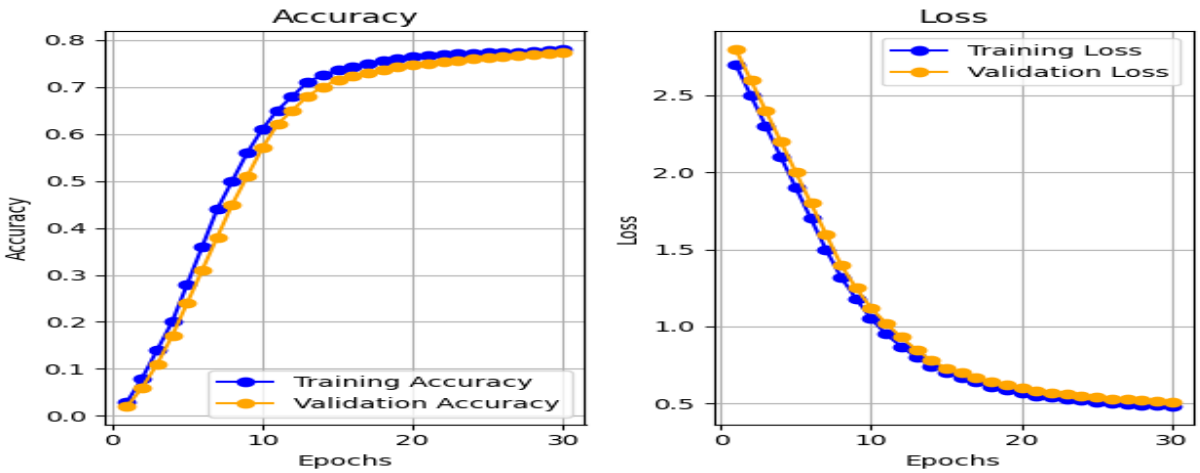

Figure 4-5 Training and Validation Accuracy and Loss utilizing LSTM

The following is a summary of the model's performance according on the final evaluation measurements:

- **Accuracy:** 77.12%
- **Precision:** 76.00%
- **Recall:** 74.40%
- **F1-score:** 75.20%

An accuracy of 77.12% shows that the model properly classifies a substantial portion of the samples. A precision of 76.00% suggests that a high proportion of the predicted positive instances are actual positives, demonstrating the model capable of reducing false positives. The recall of 74.40% expressions that the model effectively identifies most of the true positive instances. The F1-score of 75.20 Qi % expression of a balanced trade-off between precision and recall, emphasizing the model's efficacy in sentiment classification tasks.

4.5.2 Fine-Tuning Multilingual BERT for Sentiment Classification

Fine-tuned **BERT** model for sentiment classification of Amharic Facebook posts specifically, we have employed the model, which has already been trained on a huge corpus by multiple languages including Amharic for a specific task before being fine-tuned or used for other tasks making it well-suited for sentiment analysis. Therefore, first, we have loaded the dataset from an Excel file using the pandas library, focusing on two columns: sentiment (the sentiment label) and comment (the text). Then we have cleaned

the dataset by removing rows with missing values and filtering out any unexpected sentiment labels that did not belong to the predefined classes of 'strongly positive', 'positive', 'negative', 'strongly negative', and 'neutral'.

(See code: df.dropna(inplace = True) and df = df[df['sentiment'].isin(label_dict.keys())]).

Next, the sentiment labels have been mapped to numeric values, where 'positive' has been assigned the label 1, ‘strongly positive’ the label 3, 'negative' the label 2, 'strongly negative' the label 4, and 'neutral' the label 0 (df['label'] = df['sentiment'].replace(label_dict)).

We have used stratified sampling to divide the corpus two part which are training (80%) and validation (20%) positioned to maintain the balance of sentiment labels across both sets (train_test_split (df.index.values, df.label.values, test_size = 0.2, stratify = df.label.values)).

In order to get ready to enter the text data into the BERT model, we have used the BertTokenizer to tokenize the Amharic text. Tokenization has involved converting the text into token IDs, padding the sequences to a fixed length of 256 tokens, and returning attention masks for each tokenized sequence (tokenizer.batch_encode_plus(...)). The tokenized corpus which for the training and validation has encapsulated into PyTorch Tensor dataset objects to facilitate efficient batching and sampling during training.

We have fine-tuned BERT-base-multilingual-cased to sequence classification. The model has been modified to classify five sentiment labels using the **AdamW** optimizer and a linear learning rate scheduler to adjust the learning rate while training (AdamW(model.parameters(), lr=1e-5, eps=1e-8) and get_linear_schedule_with_warmup(...)).

Additionally, we have applied gradient clipping to prevent exploding gradients during training. Training has been contained to three epochs with a batch size of 4. we have evaluated the model on the validation positioned and measured performance using the **weighted F1-score** after each epoch, which accounts for class imbalance in the dataset (f1_score (labels_flat, preds_flat, average='weighted')).

The fine-tuning process has been completed using a training loop, where after each epoch, we have computed the average training loss, evaluated the model on the validation position, then calculated the weighted F1-score. The model weights have been saved after each epoch

For potential further use: -

(torch.save (model.state_dict (), f'finetuned_BERT_epoch_{epoch}.model')).

The results of the training process, including training and validation losses and the F1-score, have been recorded to measure the model's performance. This fine-tuning approach enabled the already trained BERT model to adapt effectively for Amharic sentiment classification tasks.

When training the fine-tuned Multilingual BERT model for sentiment analysis on the Amharic corpus from Facebook, we have monitored the training and validation losses over eight epochs, utilizing early stopping to prevent over-fitting. The training loss showed a consistent decline, starting from 1.5481 in the first epoch and decreasing to 1.1168 by epoch 8. Similarly, the validation loss improved initially, starting at 1.4663 in the first epoch and reaching 1.1472 by the end of training. This trend reflects a steady enhancement in model performance during the training phase, as depicted in the training and validation loss plot (Figure 4.6).

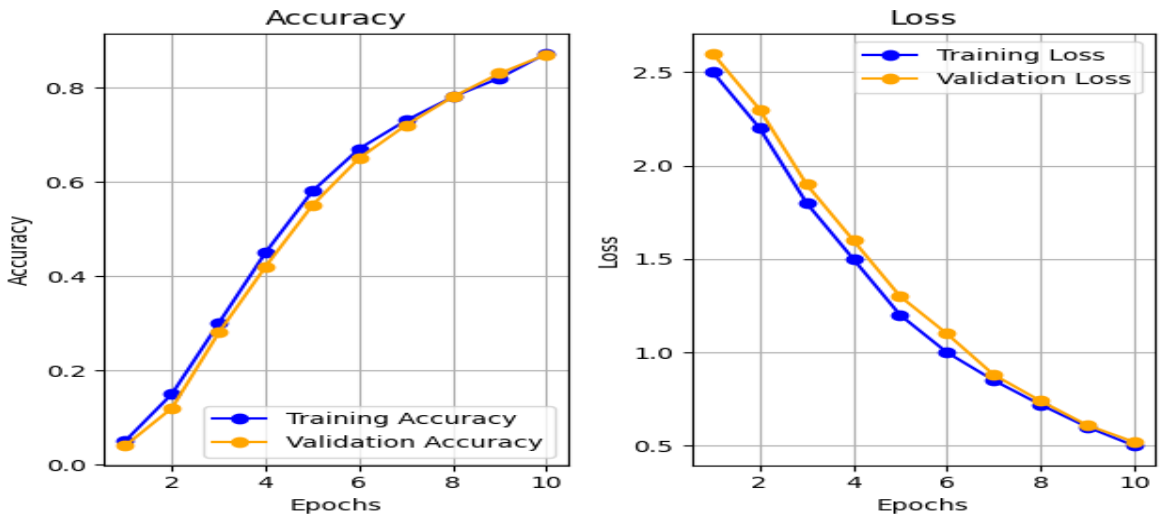

Figure 4-6 Training and Validation Accuracy and Loss utilizing BERT

The model's performance has been evaluated using a variety of metrics, such as accuracy, precision, recall, and F1-score. The evaluation of the fine-tuned Multilingual BERT model yielded the following final outcomes:

- **Accuracy:** 87.20%
- **Precision:** 79.60%
- **Recall:** 78.15%
- **F1-score:** 81.43%

With an accuracy of 87.20% shows, the model has been able to accurately classify the vast most of the samples. A precision of 79.60% highlights that a significant proportion of the predicted positive instances are actual positives, reflecting a low rate of false positives. The recall value of 78.15% demonstrates the capacity of the model to capture most of the actual positive instances. The model's outstanding performance in sentiment classification tasks have been demonstrated by its F1-score of 81.43%, which captures the balance between precision and recall.

These outcomes demonstrate the efficacy of fine-tuning Multilingual BERT for sentiment classification in Amharic, highlighting its capability to handle nuanced sentiment detection in underrepresented languages. While the model has showed strong performance, further optimizations could enhance its predictive capabilities even more. This systematic approach to model architecture illustrates deep learning approaches' promise to advance sentiment analysis in Amharic and similar languages.

4.5.3 Bi-LSTM for Amharic Sentiment Classification

Model has been constructed utilizing Bi-LSTM-based approaches for classification of sentiment to Amharic Facebook posts. The process has begun with loading and preprocessing the dataset using pandas, focusing on two key columns: sentiment (representing the sentiment label) and comment (containing the text content). The data underwent cleaning to remove any missing values and duplicate entries, ensuring that the input has been high-quality. Specifically, we have used functions such as:-

df.dropna() and df.drop_duplicates(subset='comment', keep="first") to accomplish this. Next, the sentiment labels have been converted to numeric values for compatibility with the model. We have assigned a unique integer to each sentiment class: neutral as 0, positive as 1, negative as 2, strongly positive as 3, and strongly negative as 4. This mapping enabled the model to perform multi-class classification across five distinct sentiment categories.

For text processing, we have manually constructed a vocabulary by tokenizing the comments and including only words that appeared at least once (Counter (token for sentence in sentences for token in sentence.split())). Using this vocabulary, we have encoded each tokenized sequence into numeric values, while also incorporating special tokens for padding (<pad>) and unknown words (<unk>). These sequences have been utilized to create PyTorch datasets for training and testing, which have been batched and padded for efficient processing.

The model architecture involved a Bi-LSTM-based classifier, designed with an embedding layer to transform tokenized inputs into dense vectors, the next bi-directional LSTM layer, and finally fully connected outcome level to produce predictions across the five sentiment classes. The outcome dimension has been set to five to align with our sentiment categories. We have trained the model using a weighted cross-entropy loss, where weights have been assigned according to the class distribution to manage class imbalance. The optimizer chosen has been Adam (optim.Adam (model.parameters (), lr=1e-5)).

The training process has involved monitoring both the training and validation losses over a maximum of 100 epochs, with early stopping activated after five epochs if no improvement in validation loss has been observed, to prevent overfitting. The training progression has shown a steady decrease in training loss, starting at 1.5176 in the first epoch and reaching 0.9029 by epoch 20. Similarly, the validation loss has reduced from an initial 1.4173 to 1.0427, at which point early stopping has been triggered. This training progression and performance measurement has been visualized in Figure 4.7 where we have observed convergence behavior of the training and validation losses across epochs.

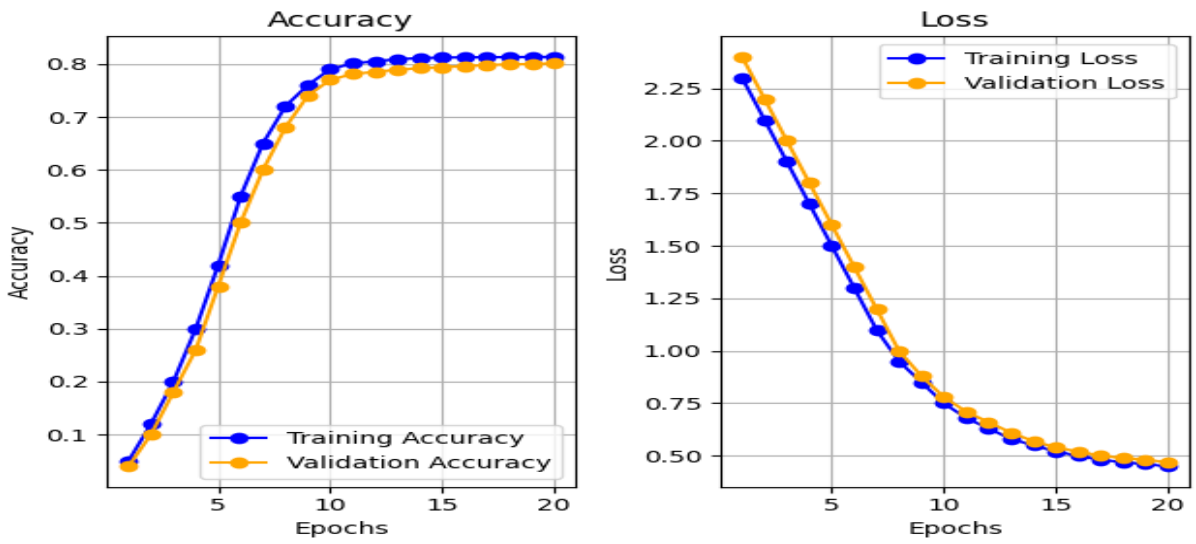

Figure 4-7 Training and Validation Accuracy and Loss utilizing Bi-LSTM

To analyze the model's performance, we employed numerous measures, including accuracy, precision, recall, and F1-score. The final results obtained from the evaluation of the Bi-LSTM model are as follows:

- **Accuracy:** 81.30%
- **Precision:** 76.50%
- **Recall:** 75.26%
- **F1-score:** 76.00%

The model properly classifies the vast majority of the samples, as demonstrated by its accuracy of 81.30%. A precision of 76.50% reflects that most of the predicted positive instances have been actual positives, showcasing a relatively low rate of false positives. The model's 75.26% recall value indicates that it can successfully identify the majority of actual positive events. The model's F1-score of 76.00% shows the strong balance between recall and accuracy, suggesting that it operates consistently in sentiment classification tasks.

These results validate the efficacy of the Bi-LSTM architecture for sentiment classification in the Amharic language. The performance underscores the ability of using deep learning techniques, like Bi-LSTMs, to enhance sentiment analysis for underrepresented languages, with further optimization offering even greater accuracy and robustness.

4.5.4 CNN for Amharic Sentiment Classification

Convolutional Neural Network (CNN) architecture is what we used to analyze the sentiment of Facebook postings in Amharic, addressing the scarcity of research in low-resource languages. The model featured several key components designed to capture linguistic features effectively. First, input indices have been converted into dense vector representation using an embedding layer, facilitating the learning of semantic relationships between words. This has been followed by a series of convolutional layers using varying filter sizes (2, 3, and 4) to capture essential n-gram features, enabling the model to learn local patterns in the data.

A ReLU activation function has been linked with each convolutional layer to add non-linearity and improve the model's capacity to recognize intricate patterns. In order to minimize the dimensionality of the feature maps and concentrate on the most prominent features while maintaining translation invariance, global max pooling has been used after convolution. For final classification into sentiment categories—strongly positive, positive, negative, strongly negative, and neutral—the pooled features have been concatenated and run through a fully connected layer. The model has been trained using a cross-entropy loss function, with class weights applied to address class imbalance, and optimized with the Adam optimizer for efficient convergence. Over-fitting has been avoided by using an early stopping mechanism that tracked validation loss and stopped training when no progress has been shown.

We have monitored the training and validation losses over 40 epochs, implementing early stopping to avoid over fitting. The training loss has shown a rapid decrease, starting from approximately 0.0175 in the first epoch and steadily declining to close to 0.001 by epoch 45. Similarly, the validation loss exhibited consistent improvement, starting from around 0.0075 and stabilizing at approximately 0.0025 towards the end of training. This trend reflects a significant enhancement in model performance, as depicted in the training and validation loss plot (Figure 4.8).

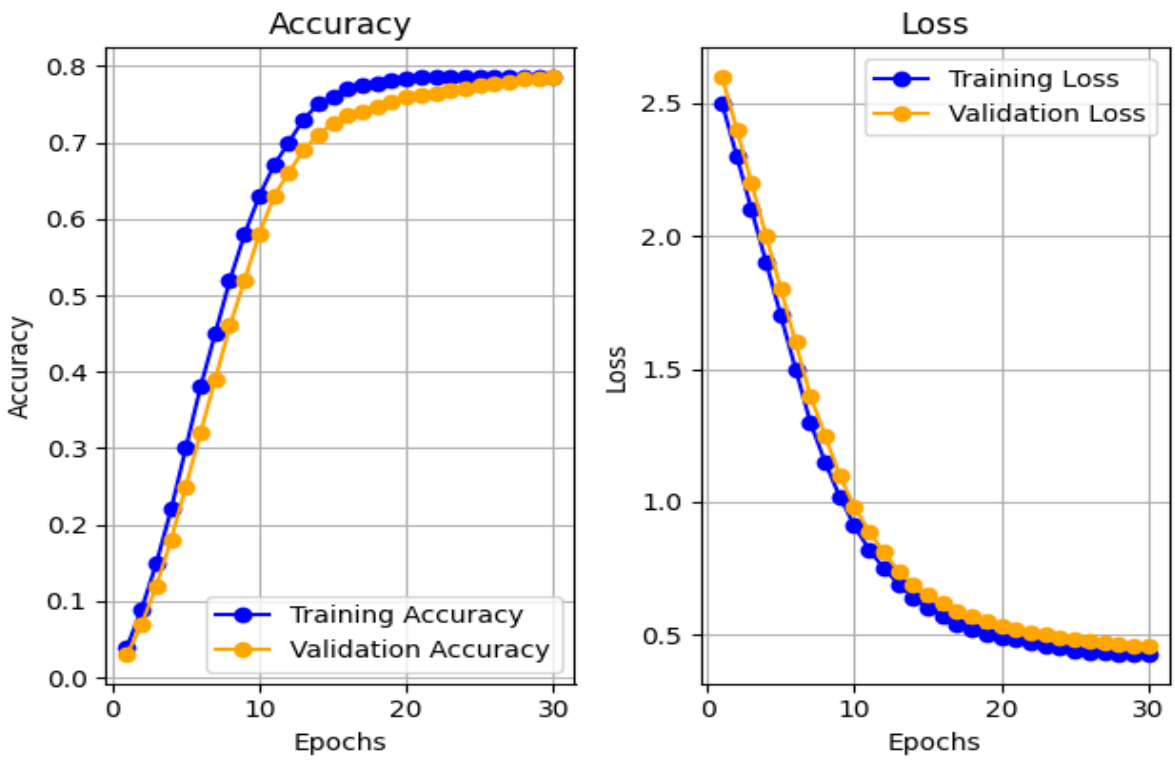

Figure 4-8 Training and Validation Accuracy and Loss utilizing CNN

The model's performance has been assessed using a number of measures, including F1-score, recall, accuracy, and precision. Following the CNN model's examination, the following are the final findings:

- **Accuracy:** 79.68%
- **Precision:** 74.40%
- **Recall:** 72.90%
- **F1-score:** 74.30%

The accuracy of 79.68% shows that the model properly classifies a significant majority of the samples. A precision of 74.40% demonstrates that most of the predicted positive instances are actual positives, reflecting a relatively low rate of false positives. The recall value of 72.90% shows the model identifies a large ration of the true positive instances. F1-score of 74.30% highlights balanced trade-off among precision and recall, showcasing the model's ability to classify sentiment effectively.

4.6 Final Results

The performance outcomes of all models used for the sentiment analysis of Facebook's Amharic datasets are shown in Table 4.2 below.

Models	Accuracy	Precision	Recall	F1-score
LSTM	0.7712	0.76	0. 744	0.752
Multilingual BERT	0.872	0.796	0.7815	0.8143
CNN	0.7968	0.744	0.729	0.743
Bi-LSTM	0.813	0.765	0.7526	0.76

Table 4-2 Results of Model

4.7 Comparison

Four deep learning approaches such as LSTM, Multilingual BERT, CNN, and Bi-LSTM have been evaluated for sentiment analysis on the Facebook Amharic corpus utilized metrics for accuracy, precision, recall, and F1-score, which are compiled in Table 4-2.

The Multilingual BERT model has shown the highest outcome of the whole measurements, attaining an accuracy of 87.20%, precision of 0.796, recall of 0.7815, and F1-score of 0.8143. This model's efficacy can be attributed to its pre-trained multilingual embedding, allowing it to capture intricate language patterns and semantic nuances specific to Amharic. The high recall and F1-score suggest that Multilingual BERT excels in retrieving relevant instances while maintaining precision, underscoring its strength in handling sentiment classification for Amharic text.

For this evaluation, BERT has been trained for 8 epochs with a batch size of 4, constrained by computational resources. Despite these limitations, BERT achieved the highest performance among the models. It is anticipated that with more computational resources, allowing for larger batch sizes and additional epochs, BERT’s performance could further improve. Even under limited resources, BERT outperformed LSTM, CNN, and Bi-LSTM

models, highlighting the adaptability of its pre-trained multilingual embedding and attention mechanisms for the complexities of sentiment analysis of Amharic Language.

The LSTM model has attained the outcome which are an accuracy ‘ 77.12%’, precision ‘76%’ recall ‘74.4%’, and F1-scores ‘75.2%’ respectively. LSTM networks have been designed to retain sequential context through memory cells, which likely contributed to its moderate performance on this task. The improved recall indicates that LSTM successfully captured more relevant sentiment instances compared to CNN, contributing to a balanced F1-score. However, its overall accuracy remained lower than BERT due to limitations in capturing nuanced patterns within the data.

The CNN model has reached the outcome of sentiment analysis as follows: mention an accuracy ‘79.68%’, precision ’74.4%’, recall ‘72.9%’, and F1-scores ’74.3%' respectively. CNNs are typically effective at capturing local patterns within text data, but they may lack the sequential context retention seen in LSTMs, which likely influenced its performance. While its precision suggests strong classification of distinct sentiments, the slightly lower recall indicates challenges in capturing the full range of relevant sentiment instances, affecting the overall F1-score.

The Bi-LSTM model attained the outcome of sentiment analysis as the following mentioned accuracy ‘81.30%’, precision ’76.5%’, recall ’75.26%, and F1-scores ‘76% respectively. Bi-LSTMs benefit from capturing both past and future context within sequences, which may explain its slightly higher performance compared to LSTM and CNN models in this evaluation. However, while it outperformed CNN in accuracy and recall, its overall performance still lagged behind BERT, reflecting the challenges of achieving comprehensive language representation without leveraging pre-trained embedding.

Multilingual BERT outperformed the LSTM, CNN, and Bi-LSTM models in Amharic sentiment analysis, exhibiting the highest competence in this dataset. This result underscores that BERT's multilingual pre-training and attention mechanisms are particularly advantageous for Amharic sentiment classification, providing a comprehensive language representation that exceeds the capabilities of the other models in this context.

Chapter Five

5. Conclusion, Recommendation and Future Work

When we have come to the final section, the study has been concluded on sentiment analysis using Amharic Language datasets from Facebook, with focus on applying deep learning techniques in a case study based on Ethiopian Broadcasting Corporation. The overall outcomes of the research have been summarized, and recommendations for future research directions are outlined.

5.1 Conclusion

This research sheds light on the unique complexities and challenges inherent in sentiment analysis of Amharic Facebook posts, focusing on the model of deep learning techniques. In particularly, we have evaluated the performance of four various models—LSTM, Bi-LSTM, fine-tuned Multilingual BERT, and CNN—each tailored for sentiment classification in Amharic, a low-resource language that presents specific challenges due to its rich morphology and complex sentence structures.

The data preprocessing phase, essential for ensuring accurate analysis, has involved cleaning and normalizing text at the character level. This step has been crucial in addressing the discrepancies inherent in the Amharic script, where character variations and script-related intricacies often hindered direct comparisons to Latin-based languages. After preprocessing, the data underwent vectorization and tokenization, preparing it for input into the models for effective sentiment analysis.

To assess model performance, we have used four assessment measurements which are accuracy, precision, recall, and F1-score. The models have revealed notable performance disparities, with LSTM achieving 77.12% accuracy, CNN reaching 79.68%, Bi-LSTM performing slightly better at 81.3%, and Multilingual BERT yielding the highest accuracy of 87.2%. These results underscore both the promise and limitations of various architectures for sentiment analysis in Amharic.

The LSTM model, while demonstrating reasonable accuracy (77.12%) and an F1-score of 75.2%, exhibited balanced performance in precision and recall compared to earlier assessments. This improvement highlights its ability to better process Amharic's complex linguistic structure, although it still lags behind Bi-LSTM and Multilingual BERT in capturing nuanced patterns. The key findings of the research are summarized, and recommendations for future research directions are outlined.

The CNN model performed slightly better, with an accuracy of 79.68% and an F1-score of 74.3%. CNN's convolutional layers helped capture important local features, aiding in sentiment classification, though its recall has been still somewhat limited, suggesting a challenge in fully identifying all relevant sentiment instances.

The Bi-LSTM model has an accuracy which 81.3% and F1-score, 76%, have leveraged bidirectional context for enhanced understanding, resulting in improved performance compared to both LSTM and CNN. However, despite this advantage, Bi-LSTM still has underperformed when compared to Multilingual BERT.

The Multilingual BERT, as the top performer, achieved accuracy of 87.2% and F1-score of 81.43%, showcasing its capacity to capture the nuanced linguistic features of Amharic. Despite being trained on limited resources (only 8 epochs with a batch size of 4), BERT's pre-trained multilingual embedding has enabled it to excel across all metrics, highlighting the power of transfer learning for sentiment analysis in low-resource languages. BERT's performance has been particularly notable in handling complex sentence structures and varied sentiment intensities, underscoring its potential in low-resource scenarios.

To sum up, this work has offered insightful information on the prospects and difficulties in sentiment analysis for underrepresented languages, such as Amharic. The evaluation affirms the potential of deep learning models, particularly Multilingual BERT, to advance sentiment classification, offering a pathway for more accurate sentiment analysis systems in low-resource languages. This research not only has demonstrated the feasibility of applying advanced deep learning techniques to Amharic sentiment analysis but also provides a foundational understanding for future improvements and innovations in sentiment analysis for low-resource languages in general.

5.2 Recommendations

The following recommendations have been put out for practitioners, researchers, and stakeholders who targeting to enhance sentiment analysis for low-resource languages like Amharic accordance with the research's findings:

- **Adopt Pre-Trained Transformer Models**: Practitioners should prioritize the utilization of transformer-based models which is Multilingual BERT for Amharic sentiment analysis, as these models demonstrate superior performance in capturing linguistic nuances and handling complex sentence structures.
- **Develop Larger and More Balanced Datasets**: Stakeholders in academia and industry are encouraged to collaborate on the creation of larger, diverse, and balanced sentiment datasets for Amharic. This could involve crowdsourcing annotated datasets or incorporating data from diverse domains which are news, social networking, and reviews.
- **Invest in Computational Resources**: Policymakers and research institutions should allocate resources to provide access to high-performance computing infrastructure, enabling researchers to fine-tune advanced models like BERT effectively.
- **Promote Interdisciplinary Collaboration**: Linguists, computer scientists, and domain experts should collaborate to address the unique linguistic challenges of Amharic and other underrepresented languages. This includes standardizing preprocessing techniques and improving tokenization methods tailored to the script and morphology.
- **Integrate Hybrid Models**: Practitioners might consider integrating CNNs, LSTMs, or Bi-LSTMs with transformer-based architectures to capture both local features and contextual information, potentially leading to more robust sentiment classification systems.

By implementing these recommendations, stakeholders can advance the construct of the best accurate and reliable sentiment analysis model to Amharic and other low-resource languages, enabling broader applications in social media monitoring, market analysis, and public opinion studies.

5.3 Future Work

It has already been built on the results of the proposed model; this research has highlighted several areas with potential for further advancements. The following directions suggest promising avenues for future investigation that extend beyond the scope of the current study:

- **Model Refinement and Optimization**: Although the multilingual BERT model achieved the highest performance in this study, there remains room for improvement in capturing the unique linguistic nuances of Amharic and addressing class imbalances in the dataset. Future work could explore techniques such as fine-tuning on larger and more diverse datasets, and ensemble methods to improve the robustness and accuracy classification of sentiment.
- **Enhanced BERT Fine-Tuning with Greater Computational Resources**: Future research could benefit from fine-tuning BERT with access to more advanced computational resources. This would allow for the training of larger, more complex models and facilitate deeper exploration of sentiment-specific nuances in Amharic text, potentially leading to enhanced model performance.
- **Expanding the Sentiment Corpus**: the key limitations in this research have been the size of the sentimentally classified dataset. Creating larger, more diverse and well-labeled sentiment datasets for Amharic could significantly enhance model performance and generalizability. More extensive datasets would also provide opportunities for better handling of the language's rich morphology and varied sentiment expressions.

These future directions could pave the method for more accurate and effective sentiment analysis in low-resource languages like Amharic.

References

1. E. Cambria, "An Introduction to Concept-Level Sentiment Analysis," pp. 478-481, 2013.
2. A. O'Neill., "Sentiment Mining for Natural Language Documents," https://users.cecs.anu.edu.au/~ssanner/Papers/Alex_Report.pdf, 2009.
3. A. A. Qaid Aqlan, B. Manjula and R. Lakshman Naik, "A Study of Sentiment Analysis: Concepts, Techniques, and Challenges," pp. 147-160, 2019.
4. H. Moe, "Public Service Broadcasting and Social Networking Sites: The Norwegian Broadcasting Corporation on Facebook," 2013.
5. G. Beigi, R. Maciejewski and H. Liu, "An Overview of Sentiment Analysis in Social Media and its Applications in Disaster Relief," 2016.
6. Y. Gorodnichenko, T. Pham and O. Talavera, "Social Media, Sentiment and Public Opinions: Evidence From #BREXIT And #US election," pp. 5-21, 2018.
7. J. R., H. Dong, A. Popovic, G. Sabnis and J. Nickerson, "Digital platforms in the news industry: how social media platforms impact traditional media news viewership," *European Journal of Information Systems,* pp. 1-13, 2022.
8. F. Neri, C. Aliprandi, F. Capeci, M. Cuadros and T. By, "Sentiment Analysis on Social Media," *International Conference on Advances in Social Networks Analysis and Mining,* pp. 951-955, 2012.
9. C. Schweizer and M. Puppis, "Public Service Media in the 'Network' Era," 2018.
10. Á. Gulyás and . F. Hammer, Public Service Media in the Digital Age: International Perspectives, London: Cambridge Scholars Publishing, 2013.
11. B. Liu, "Sentiment Analysis Mining Opinions, Sentiments, and Emotions,", 2015.
12. S. Shah, A. Bhat, S. Singh, A. Chavan, A. Singh, "Sentiment Analysis", pp. 1542-1546, 2024.
13. Q. T. Ain, M. Ali, A. Riaz, A. Noureen, M. Kamran, B. Hayat and A. Rehman, "Sentiment Analysis Using Deep Learning Techniques," *A Review; (IJACSA) International Journal of Advanced Computer Science and Applications,* pp. 424-429, 2017.

14. G. Yeshewas, "Deep learning Approach for Amharic sentiment analysis using scraped social Media data," pp. 2-6, 2023.
15. Jordan, P. (2014). Nation branding: A tool for nationalism? *Journal of Baltic Studies, 45*(3), 283-303. https://doi.org /10.1080/01629778.2013.860609.
16. O. HABIMANA, Y. LI, L. Ruixuan, X. Gu and Y. Ge , "Sentiment analysis using deep learning approaches: an overview," *Science China Information Sciences* , pp. 1-5, 2020.
17. Shilpa P. C., R. Shereen, S. Jacob and P. vinod, "Sentiment Analysis Using Deep Learning," pp. 181-183, 2021.
18. F. Alemayehu, M. Meshesha and J. Abate, "Amharic political sentiment analysis using deep learning approaches," 2023.
19. Su, Y., Zhang, Y., Ji, D., Wang, Y., Wu, H., "Ensemble Learning for Sentiment Classification," 2013.
20. Ministry of Water and Energy, "Ethiopia Experience in Implementing Nature Based Solution to Solve Water Related Problems," 2023.
21. J. Bens and O. Zenker, "Sentiment," 2019.
22. P. Gottlie, "The Virtue of Aristotle's Ethics," 2009.
23. A. Wierzbicka, "Moral sense," *Journal of Social Evolutionary and Cultural Psychology*, p. 75-76, 2007.
24. A. Balahur, "Sentiment Analysis in Social Media Texts," pp. 120-126, 2013.
25. Z. salah, A. ARF Al-Ghuwairi, A. Baarah, A. Aloqail, B. Qadoumi, M. Alhayek and B. Alhijawi, "A Systematic Review on Opinion Mining and Sentiment Analysis in Social Media," 2019.
26. E. Khalil, E. M. F. El Houby and H. K. Mohame, "Sentiment Analysis Tasks and Approaches," *International Journal of Computer Science and Information Security,* 2021.
27. S. Kolkar, G. Dantal and R. Mahe, "Study of Different Levels for Sentiment Analysis," *International Journal of Current Engineering and Technology,* pp. 768-770, 2015.
28. M. Wankhade, A. C. S. Rao and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," *Artificial Intelligence Review,* PP. 5732-5737, 2022.
29. E. Sanchez-DelaCruz and D. Lara-Alabazares, "Deep learning: concepts and implementation tools," pp. 142-147, 2019.

30. Jianqing Fan, Cong Ma and Yiqiao Zhong, "A Selective Overview of Deep Learning" pp, 1-36, 2021.
31. Sachin Sambhaji Patil and T. B. Mohite-Patil, "Deep Belief Neural Network Based Automatic NSTEMI CVD Prediction Using Adaptive Sliding Window Technique", PP 1-14, 2023.
32. A. Mathew, P. Amudha and S. Sivakumari, "Deep Learning Techniques: An Overview," pp. 599-606, 2021.
33. E. E. Katsoulakis, "Public Service Broadcasting in the digital media age How streaming services reshape the role of Greek public network ERT from state to public service broadcaster?," 2022.
34. T. Nasukawa and J. Yi, "Sentiment analysis: capturing favorability using natural language processing," *IBM Almaden Research Center*, 2003.
35. S.-Z. M. D. Pilar, L.-L. Estanislao, V.-G. Rafael , A.-G. Nathalie and A. Á. A.-H. Giner, "A study on LIWC categories for opinion mining in Spanish reviews," *Journal of Information Science*, 2014.
36. Y. Mejova and . P. Srinivasan, "Exploring feature definition and selection for sentiment classifiers," in *The Fifth International AAAI Conference on Weblogs and Social Media*, 2011.
37. D. Sharma, M. Sabharwal, V. Goyal and M. Vij, "Sentiment Analysis Techniques for Social Media Data: A Review," pp. 75-82, 2020.
38. B. Mahesh, "Machine Learning Algorithms - A Review," *International Journal of Science and Research (IJSR)*, pp. 381-385, 2020.
39. Izunna, U. Okpala, "Perception Analysis: A knowledge Discovery and inference Generation approach to christ informatics," 2023.
40. D. Michie, D. J. Spiegelhalter and C. C. Taylor and J. Campbell, "Machine Learning, Neural and Statistical Classification," United States: Ellis Horwood, 1995.
41. T. D. Buskirk, A. Kirchne, A. Eck and C. S. Signorino, "An Introduction to Machine Learning Methods for Survey Researchers," https://doi.org/10.29115/SP-2018-0004., 2018.

42. Asongo, A. I., Barma, M. and Muazu, H. G. "Machine Learning Techniques, Methods and Algorithms: Conceptual and Practical Insights," *International Journal of Engineering Research and Applications*, pp. 55-62, 2021.
43. T. O. Ayodele, "Machine Learning Overview," PP. 09-16, 2010.
44. R. I. Muhamedyev, "Machine learning methods: An overview," pp. 14-19, 2015.
45. Y. R. Tausczik and James, W. Pennebaker, "The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods," *Journal of Language and Social Psychology*, pp. 25-28, 2010.
46. J. W. Pennebaker, M. E. Francis, and R. J. Booth, "Linguistic Inquiry and Word Count (LIWC)," Erlbaum Publishers, 2001.
47. Thomas, Bischoff, "Marriage and Family Therapist Websites: A Qualitative Content Analysis," pp. 1-73, 2019.
48. T. Yu , C. Hidey , O. Rambow and K. McKeown, "Leveraging Sparse and Dense Feature Combinations for Sentiment Classification Feature Selection," 2017.
49. Manal, Mostafa, A., "Arabic sentiment analysis about online learning to mitigate covid-19," *Journal of Intelligent Systems*, pp. 524-537, 2021.
50. V. Umarani, A. Julian, and J. Deepa, "Sentiment Analysis using various Machine Learning and Deep Learning Techniques," *Journal of the Nigerian Society of Physical Sciences* , p. 385–393, 2021.
51. J. L. Elman, "Finding structure in time," *Cogn Sci*, p. 179–211, 1990.
52. S. Al-Saqqa, and A. Awajan, "The Use of Word2vec Model in Sentiment Analysis: A Survey," pp. 39-41, 2019.
53. N. B. Yeswanth and K. K. Devi, "Twitter Sentiment Analysis Using Word2vec and TF-IDF Word Embedding," pp. 120-122, 2018.
54. S.-M. Kim, and Eduar, Hovy, "Determining the sentiment of opinions," 2004.
55. W. Medhat, A. Hassan, and H. Korashy "Sentiment analysis algorithms and applications a survey," *Ain Shams Eng J (Elsevier B.V.)*, p. 1093–1109, 2014.
56. S. Saad, and B. Saberi, "Sentiment Analysis or Opinion Mining: A Review," *International Journal on Advanced Science Engineering and Information Technology*, 2017.

57. H. Malik, E. M. Shakshuki and A. U.-H. Yasar, "Approximating Viewership of Streaming T.V Programs Using Social Media Sentiment Analysis," Elsevier B.V, 2021.
58. J. Kazmaier and J. H. V. Vurren, "The power of ensemble learning in sentiment analysis," (Elsevier Ltd.), 2021.
59. M. Thelwall, K. Buckely, and G. Paltoglou, "Sentiment strength detection for the social web," *Journal of the American Society for Information Science and Technology (JASIST)*, 2012.
60. G. Vaitheeswaran and Dr. L. Arockiam, "Combining Lexicon and Machine Learning Method to Enhance the Accuracy of Sentiment Analysis on Big Data," *International Journal of Computer Science and Information Technologies*, PP. 306-309, 2016.
61. C. C. Ruiz, and I. S. Bedmar, "Comparing deep learning architectures for sentiment analysis on drug reviews," 2021.
62. avad, H. Joloudari1, S. Hussain, M. A. Nematollahi, R. Bagheri, F. Fazl, R. Alizadehsani, R. Lashgari, and A. Talukder, " BERT-Deep CNN: State-of-the-Art for Sentiment Analysis of COVID-19 Tweets," 2023.
63. T. Mullen and N. Collier, "Sentiment analysis using support vector machines with diverse information sources," 2004.
64. I. Setiawan, A. M. Widodo, M. Rahaman, T., M. A. Hadi, N. Anwar, M. B. Ulum, E. Y. Mulyani and N. Erzed, "Utilizing Random Forest Algorithm for Sentiment Prediction Based on Twitter Data," pp. 446-453, 2022.
65. A. T. Mahmood, S. S. Kamaruddin, R. K. Naser, and M. M. Nadzir, "A Combination of Lexicon and Machine LearningApproaches for Sentiment Analysis on Facebook," *Journal of System and Management Science*, PP. 140-146, 2020.
66. L. Deng, “Artificial Intelligence in the Rising Wave of Deep Learning,” IEEE Signal Process. Mag., vol. 35, no. 1, PP. 177–180, 2018.
67. D. Chavali, B. Baburajan, V. Kumar, and S. C. Katari, “Regulating Artificial intelligence Developments and Challenges,” *International Journal of Pharmaceutical Sciences*, 2024.
68. Sibanjan, D., U. Cakmak, “Hands-on Automated Machine Learning: A Beginners Guide to Building Automated Machine Learning Systems Using AutoML and Python.” Packt Publishing Ltd, 2018.

69. S. S. Dash, and D. Mishra, and S. K. Nayak, “A Review: Machine Learning Algorithms,” PP. 1-8, 2021.
70. A. Kanneganti, “Using Ensemble Machine Learning Methods in Estimating Software Development Effort,” 2020.
71. R. R. Sinha, and R. K. Gora, “Software effort estimation using machine learning techniques,” 2021.
72. Yuxi, Li. "Deep Reinforcement Learning: An Overview," 2018.
73. S. B. Kotsiantis, “Supervised Machine Learning: A Review of Classification Techniques,” PP. 249-263, 2007.
74. R. Saravanan and P. Sujatha, “A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification,” in Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, Jun., PP. 945–949, 2018.
75. P. A. Flach, "Machine Learning: The Art and Science of Algorithms that Make Sense of Data," Cambridge University Press, 2012.
76. M. Hosni and A. Idri, “Software effort estimation using classical analogy ensembles based on random subspace,” PP. 1251–1258, 2017.
77. S. Moulder, T. Sheridan, P. Cavallo, G. Rossini, "Deep Learning with MATLAB and Multiple GPUs," 2017.
78. A. Tipirisetty, "Stock Price Prediction using Deep Learning," Master's Projects. 636. DOI: https://doi.org/10.31979/etd.bzmm-36m7, 2018.
79. “data camp,” 25 03 2022. [Online]. Available: https://www.datacamp.com/tutorial/tutorial-lasso-ridge-regression.
80. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, and B. Thirion, “Scikit-learn: Machine Learning in Python,” PP. 2826-2829, 2011.
81. Y. Bengio, P. Simard, P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” IEEE Trans. Neural Networks, PP. 157–166, 1994.
82. Halit, A., H. Feizi, M. T. Sattari, M. S. Colak, S. Shamshirband and K. W. Chau, “Comparative Analysis of Recurrent Neural Network Architectures for Reservoir Inflow Forecasting,” *Water,* 2020.

83. A. Shrestha, and A. Mahmood, “Review of Deep Learning Algorithms and Architectures,” *IEEE Access,* 2019.
84. B. Demeke, A.,” Web Traffic Analysis and Forecasting using Deep Learning Time-Series Approach In case of Commercial Bank of Ethiopia," 2024.
85. Z. Huang, W. Xu, and K. Yu, “Bidirectional LSTM-CRF Models for Sequence Tagging,” 2015.
86. Qi, Cheng, Y. Chen, Y. Xiao, Y. Hongsheng, W. Liu, “A dual-stage attention-based Bi-LSTM network for multivariate time series prediction,” PP. 16214–16235, 2022.
87. Ashok, k., T. E. Trueman, and E. Cambria, “A Convolutional Stacked Bidirectional LSTM with a Multiplicative Attention Mechanism for Aspect Category and Sentiment Detection,” https://doi.org/10.1007/s12559-021-09948-0, 2021.
88. Jacob, D., Ming-wei, C., K. Lee, and K. Toutanova, " BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," PP. 4171–4186, *2019*.
89. Y. Kim, "Convolutional Neural Networks for Sentence Classification," PP. 1746–1751, 2014. Qi

Appendixes

❖ Sentiment Categories and Labeling Guidelines

1. **Strongly Positive:** Sentences in this category express a high degree of positivity or enthusiasm, often with words that indicate extreme happiness or satisfaction.
    - **Example Labeling:**
        - *Amharic:* ይህንን ምርት በፍጹም እወደዋለሁ! (English: "I absolutely love this product!") *(Label: strongly_positive)*
        - *Amharic:* የ ሚይታመን አፈጻጸም! (English: "Incredible performance!") *(Label: strongly_positive)*
    - **Annotation Criteria:** Look for emphatic terms like "በፍጹም' ("absolutely") or superlative expressions that intensify positive emotions.
2. **Positive:** Sentences here express a favorable opinion, but with less intensity than strongly positive. They show satisfaction or approval in a moderate, non-extreme manner.
    - **Example Labeling:**
        - *Amharic:* በአገልግሎቱ ደስተኛ ነኝ። (English: "I'm happy with the service.") *(Label: positive)*
        - *Amharic:* ፊልሙ ጥሩ ነበር, ግን አስደናቂ አልነበረም። (English: "The movie was good, but not amazing.") *(Label: positive)*
    - **Annotation Criteria:** Identify phrases that indicate satisfaction or mild happiness, without high-intensity adverbs or strong endorsement.
3. **Neutral:** Sentences in this category convey factual, objective statements without any emotional bias or reaction.
    - **Example Labeling:**
        - *Amharic:* የዛሬው የአየር ሁኔታ ምንም አይልም። (English: "The weather today is just fine.") *(Label: neutral)*
        - *Amharic:* ጥቅሉ ደርሶኛል። (English: "I received the package.") *(Label: neutral)*

- **Annotation Criteria:** Label sentences as neutral if they are purely factual or objective, without any words indicating emotional positivity or negativity.

4. **Negative:** This category includes sentences that convey disapproval, dissatisfaction, or disappointment. The sentiment is negative but not extreme.
    - **Example Labeling:**
        - *Amharic:* ይህ ምርት እንደጠበኩት አይደለም። (English: "This product didn't meet my expectations.") *(Label: negative)*
        - *Amharic:* ዜናው አሰልቺ ነበር። (English: "The news was boring.") *(Label: negative)*
    - **Annotation Criteria:** Negative sentences often contain mild criticisms or expressions of disappointment, without phrases indicating strong negativity.
5. **Strongly Negative:** Sentences in this category express strong disapproval, anger, or intense dissatisfaction.
    - **Example Labeling:**
        - *Amharic:* ይህ እስካሁን ከገዛኋቸውምርቶች ሁሉ የከፋውነው! (English: "This is the worst product I have ever bought!") *(Label: strongly_negative)*
        - *Amharic:* በፍፁምእጠላዋለሁ። (English: "I absolutely hate it.") *(Label: strongly_negative)*
    - **Annotation Criteria:** Look for language that conveys extreme negativity, such as absolute disapproval or expressions of intense dissatisfaction.

❖ Cleaning the Dataset

```
# Clean text function
def clean_text(row, options):
    """Removes URL, mentions, emojis, and uppercase from tweets."""
    if options['lowercase']:
        row = row.lower()

    if options['remove_url']:
        row = re.sub(r"(?:\@|https?\://)\S+", "", row)

    if options['remove_mentions']:
        row = re.sub(r"@[A-Za-z0-9_]+", "", row)

 # Emojis are not removed anymore, so we can remove the demojify block
        return row

# Configuration for cleaning
clean_config = {
    'remove_url': True,
    'remove_mentions': True,
    'lowercase': True,
    'demojify': False,   # Set demojify to False to keep emojis
}
```

```
# Apply cleaning to the tweet columns
df['Comments'] = df['Comments'].apply(clean_text, args=(clean_config,))

# Display cleaned comments
print("Cleaned DataFrame:")
print(df['Comments'].head())
print(df['Label'].head())
```

```
Cleaned DataFrame:
0    ጋላ ፈጣሪ እምነት ሀይማኖት 😔የለውም ጋላ ክብር አይወድለትም ጋላ አሳዳጊ...
1    እራሱ መንግስት ሊመታሲል ወይንም የኦሮሞ ፅፈኛ ቄሮወች አደጋ እንደሚያደር...
2                                  ክፉ መርዝ አንተ ነህ ጅለንፍ😔!
3                      አማራ ብሎ እስላም ትግራዋይ ብሎ ዋቅፈታ የለውም😔::
4    አስተሳሰቡ በ19 ኛው ክፍለ ዘመን ላይ ተቸንክሮ የቀረው ብቸኛው የአለማች...
Name: Comments, dtype: object
0    Strongly Negative
1    Strongly Negative
2    Strongly Negative
3    Strongly Negative
4    Strongly Negative
Name: Label, dtype: object
```

❖ Character Normalization

```
# Apply normalization to the comments column
df['Comments'] = df['Comments'].apply(lambda x: normalize_char_level_missmatch(x))

# Print the result after normalization
print("After normalization comment:\n", df['Comments'].head())
print("After normalization label:\n", df['Label'].head())

print("Unique labels in the dataset:", df['Label'].unique())


#character level normalization
import re
#method to normalize character level missmatch such as ጸሀይ and ፀሐይ
def normalize_char_level_missmatch(input_token):
    rep1=re.sub('[ሃኀኃሐኻ]','ሀ',input_token)
    rep2=re.sub('[ሑኁኹ]','ሁ',rep1)
    rep3=re.sub('[ኂሒኺ]','ሂ',rep2)
    rep4=re.sub('[ኌሔኼ]','ሄ',rep3)
    rep5=re.sub('[ሕኅ]','ህ',rep4)
    rep6=re.sub('[ኆሖኾ]','ሆ',rep5)
    rep7=re.sub('[ሠ]','ሰ',rep6)
    rep8=re.sub('[ሡ]','ሱ',rep7)
```

```
    rep9=re.sub('[ኺ]','ሲ',rep8)
    rep10=re.sub('[ኻ]','ሳ',rep9)
    rep11=re.sub('[ኼ]','ሴ',rep10)
    rep12=re.sub('[ሥ]','ስ',rep11)
    rep13=re.sub('[ሦ]','ሶ',rep12)
    rep14=re.sub('[ዓኣዐ]','አ',rep13)
    rep15=re.sub('[ዑ]','ኡ',rep14)
    rep16=re.sub('[ዒ]','ኢ',rep15)
    rep17=re.sub('[ዔ]','ኤ',rep16)
    rep18=re.sub('[ዕ]','እ',rep17)
    rep19=re.sub('[ዖ]','ኦ',rep18)
    rep20=re.sub('[ጸ]','ፀ',rep19)
    rep21=re.sub('[ጹ]','ፁ',rep20)
    rep22=re.sub('[ጺ]','ፂ',rep21)
    rep23=re.sub('[ጻ]','ፃ',rep22)
    rep24=re.sub('[ጼ]','ፄ',rep23)
    rep25=re.sub('[ጽ]','ፅ',rep24)
    rep26=re.sub('[ጾ]','ፆ',rep25)
    #Normalizing words with Labialized Amharic characters such as በልቱዋል or  በልቱአል to  በልቷል
    rep27=re.sub('(ሉ[ዋአ])','ሏ',rep26)
    rep28=re.sub('(ሙ[ዋአ])','ሟ',rep27)
    rep29=re.sub('(ቱ[ዋአ])','ቷ',rep28)
    rep30=re.sub('(ሩ[ዋአ])','ሯ',rep29)
    rep31=re.sub('(ሱ[ዋአ])','ሷ',rep30)
    rep32=re.sub('(ሹ[ዋአ])','ሿ',rep31)
    rep33=re.sub('(ቁ[ዋአ])','ቋ',rep32)
    rep34=re.sub('(ቡ[ዋአ])','ቧ',rep33)
    rep35=re.sub('(ቹ[ዋአ])','ቿ',rep34)
    rep36=re.sub('(ሁ[ዋአ])','ኋ',rep35)
    rep37=re.sub('(ኑ[ዋአ])','ኗ',rep36)
    rep38=re.sub('(ኙ[ዋአ])','ኟ',rep37)
    rep39=re.sub('(ኩ[ዋአ])','ኳ',rep38)
    rep40=re.sub('(ዙ[ዋአ])','ዟ',rep39)
    rep41=re.sub('(ጉ[ዋአ])','ጓ',rep40)
    rep42=re.sub('(ደ[ዋአ])','ዷ',rep41)
    rep43=re.sub('(ጡ[ዋአ])','ጧ',rep42)
    rep44=re.sub('(ጩ[ዋአ])','ጯ',rep43)
    rep45=re.sub('(ጹ[ዋአ])','ጿ',rep44)
    rep46=re.sub('(ፉ[ዋአ])','ፏ',rep45)
    rep47=re.sub('[ቍ]','ቁ',rep46) #ቁ can be written as ቍ
    rep48=re.sub('[ኵ]','ኩ',rep47) #ኩ can be also written as ኵ
    return rep48
```

```
After normalization comment:
 0    ጋላ ፈጣሪ እምነት ሀይማኖት 😔የለውም ጋላ ክብር አይወድለትም ጋላ አሳዳጊ...
1    እራሱ መንግስት ሊመታሲል ወይንም የኦሮሞ ፅፈኛ ቄሮወች አደጋ እንደሚያደር...
2                                  ክፉ መርዝ አንተ ነህ ጅለንፍ😔!
3                    አማራ ብሎ እስላም ትግራዋይ ብሎ ዋቅፈታ የለውም😔::
4    አስተሳሰቡ በ19 ኛው ክፍለ ዘመን ላይ ተቸንክሮ የቀረው ብቸኛው የአለማች...
Name: Comments, dtype: object
After normalization label:
 0    Strongly Negative
1    Strongly Negative
2    Strongly Negative
3    Strongly Negative
4    Strongly Negative
Name: Label, dtype: object
Unique labels in the dataset: ['Strongly Negative' 'Neutral' 'Negative' 'Strongly positive' 'positive']
```

❖ CNN Model

```python
# Define CNN-based model for sentiment classification
class CNNSentimentClassifier(nn.Module):
    def __init__(self, vocab_size, embedding_dim, num_filters, filter_sizes, output_dim):
        super(CNNSentimentClassifier, self).__init__()
        self.embedding = nn.Embedding(vocab_size, embedding_dim, padding_idx=vocab['<pad>'])
        self.convs = nn.ModuleList([
            nn.Conv2d(in_channels=1, out_channels=num_filters, kernel_size=(fs, embedding_dim))
            for fs in filter_sizes
        ])
        self.fc = nn.Linear(len(filter_sizes) * num_filters, output_dim)
        self.dropout = nn.Dropout(0.5)

    def forward(self, text):
        embedded = self.embedding(text).unsqueeze(1)  # Add channel dimension
        conved = [torch.relu(conv(embedded)).squeeze(3) for conv in self.convs]
        pooled = [torch.max(c, dim=2)[0] for c in conved]  # Global max pooling
        cat = self.dropout(torch.cat(pooled, dim=1))
        return self.fc(cat)

# Model parameters
vocab_size = len(vocab)  # Number of words in the vocabulary
```

```
embedding_dim = 100  # Dimension of word embeddings
num_filters = 100  # Number of filters (feature maps) per filter size
filter_sizes = [2, 3, 4]  # Filter sizes
output_dim = 5  # Number of output classes (5 categories)

# Initialize CNN model
model = CNNSentimentClassifier(vocab_size, embedding_dim, num_filters, filter_sizes, output_dim)

# Define the loss function and optimizer
# Make sure class_weights has the correct shape (5 elements for 5 classes)
class_weights = torch.tensor([1.0, 2.0, 1.0, 2.5, 2.5], dtype=torch.float32)  # Adjust based on dataset
#class_weights = torch.tensor([1.0, 1.5, 1.0, 1.0, 1.0], dtype=torch.float32)  # Adjust based on dataset
criterion = nn.CrossEntropyLoss(weight=class_weights)  # Apply class weights
optimizer = optim.Adam(model.parameters(), lr=1e-5)

# Training and validation losses
train_losses = []
val_losses = []
train_accuracies=[]
val_accuracies=[]
# Early stopping parameters
early_stopping_patience = 5
best_val_loss = float('inf')
patience_counter = 0
```

```
def train_model(model, train_loader, criterion, optimizer, num_epochs=100):
    global best_val_loss, patience_counter

    for epoch in range(num_epochs):
        model.train()
        epoch_loss, correct_train, total_train = 0, 0, 0

        for texts, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(texts)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()
            epoch_loss += loss.item()

            # Compute training accuracy
            preds = outputs.argmax(dim=1)
            correct_train += (preds == labels).sum().item()
            total_train += labels.size(0)

        avg_loss = epoch_loss / len(train_loader)
        train_losses.append(avg_loss)
        train_accuracy = correct_train / total_train
        train_accuracies.append(train_accuracy)
        print(f'Epoch {epoch + 1}, Training Loss: {avg_loss:.4f}, Training Accuracy: {train_accuracy:.4f}')
```

```
        # Validation step
        val_loss, val_accuracy = evaluate_model(model, test_loader)
        val_losses.append(val_loss)
        val_accuracies.append(val_accuracy)
        print(f'Epoch {epoch + 1}, Validation Loss: {val_loss:.4f}, Validation Accuracy: {val_accuracy:.4f}')
```

```
        # Early stopping
        if val_loss < best_val_loss:
            best_val_loss = val_loss
            patience_counter = 0
        else:
            patience_counter += 1

        if patience_counter >= early_stopping_patience:
            print(f'Early stopping at epoch {epoch + 1}')
            break

# Evaluation function with accuracy

def evaluate_model(model, test_loader):
    model.eval()
    epoch_loss, correct_val, total_val = 0, 0, 0

    with torch.no_grad():
        for texts, labels in test_loader:
            outputs = model(texts)
            loss = criterion(outputs, labels)
            epoch_loss += loss.item()

            # Compute validation accuracy
            preds = outputs.argmax(dim=1)
            correct_val += (preds == labels).sum().item()
            total_val += labels.size(0)

    val_loss = epoch_loss / len(test_loader)
    val_accuracy = correct_val / total_val
    return val_loss, val_accuracy
```

❖ Bi-LSTM Model

```
# BiLSTM Sentiment Classifier
class BiLSTMSentimentClassifier(nn.Module):
    def __init__(self, vocab_size, embedding_dim, hidden_dim, output_dim):
        super(BiLSTMSentimentClassifier, self).__init__()

        # Embedding layer to convert words into vectors
        self.embedding = nn.Embedding(vocab_size, embedding_dim, padding_idx=vocab['<pad>'])

        # Bi-directional LSTM (batch_first=True ensures batch is the first dimension)
        #I include the following code

        self.lstm = nn.LSTM(embedding_dim, hidden_dim,num_layers=2, batch_first=True, bidirectional=True, dropout = 0.5 )
        #self.lstm = nn.LSTM(embedding_dim, hidden_dim, batch_first=True, bidirectional=True)

        # Fully connected layer after the LSTM, output dimension will be doubled because of bidirectional LSTM
        self.fc = nn.Linear(hidden_dim * 2, output_dim)  # hidden_dim * 2 for bidirectional

    def forward(self, text):
        # Pass the input text through the embedding layer
        embedded = self.embedding(text)

        # Pass the embedded input through the Bi-LSTM layer
        _, (hidden, _) = self.lstm(embedded)
```

```
        # Concatenate the last hidden states from both directions (forward and backward)
        # hidden shape: (num_layers * num_directions, batch_size, hidden_dim)
        # We want the last hidden states from both directions
        # - hidden[-2] is from the last layer's forward direction
        # - hidden[-1] is from the last layer's backward direction
        hidden = torch.cat((hidden[-2], hidden[-1]), dim=1)  # Concatenate forward and backward hidden states

        # Pass the concatenated hidden states through the fully connected layer
        return self.fc(hidden)

# Initialize model, loss, and optimizer
vocab_size = len(vocab)
embedding_dim = 100
hidden_dim = 256

output_dim = 5  # For five classes (strongly positive, positive, negative, strongly negative, neutral)

model = BiLSTMSentimentClassifier(vocab_size, embedding_dim, hidden_dim, output_dim)
```

```
# Define the loss function and the optimizer (with learning rate)
class_weights = torch.tensor([1.0, 2.0, 1.0, 2.5, 2.5], dtype=torch.float32)  # Adjust weights based on dataset
#class_weights = torch.tensor([1.0, 1.5, 1.0, 1.5, 1.0], dtype=torch.float32)  # Adjust weights based on dataset
criterion = nn.CrossEntropyLoss(weight=class_weights)
# I include the following code
optimizer = optim.Adam(model.parameters(), lr=1e-4)  # Learning rate
#optimizer = optim.Adam(model.parameters(), lr=1e-5)  # Learning rate
# Initialize lists to store training and validation losses
train_losses = []
val_losses = []
train_accuracies= []
val_accuracies =[]
# Early stopping parameters
early_stopping_patience = 5
best_val_loss = float('inf')
patience_counter = 0

# Evaluation function
def evaluate_model(model, data_loader, criterion):
    model.eval()
    val_loss, correct, total = 0, 0, 0
```

```
    with torch.no_grad():
        for texts, labels in data_loader:
            texts, labels = texts.long(), labels.long()
            outputs = model(texts)

            loss = criterion(outputs, labels)
            val_loss += loss.item()

            # Compute validation accuracy
            _, predicted = torch.max(outputs, 1)
            correct += (predicted == labels).sum().item()
            total += labels.size(0)

    # Ensure a valid return
    if total == 0 or len(data_loader) == 0:
        return 0.0, 0.0  # Default values to prevent unpacking errors

    avg_val_loss = val_loss / len(data_loader)
    val_accuracy = 100 * correct / total
    return avg_val_loss, val_accuracy

# Training loop
def train_model(model, train_loader, criterion, optimizer, num_epochs=100):
    global best_val_loss, patience_counter
```

```
for epoch in range(num_epochs):
    model.train()
    epoch_loss, correct, total = 0, 0, 0  # Reset per epoch

    for texts, labels in train_loader:
        optimizer.zero_grad()

        texts, labels = texts.long(), labels.long()
        outputs = model(texts)

        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        epoch_loss += loss.item()

        # Compute training accuracy
        _, predicted = torch.max(outputs, 1)
        correct += (predicted == labels).sum().item()
        total += labels.size(0)

    avg_loss = epoch_loss / len(train_loader)
    train_losses.append(avg_loss)
    train_accuracy = 100 * correct / total
    train_accuracies.append(train_accuracy)

    print(f'Epoch {epoch + 1}, Training Loss: {avg_loss:.4f}, Training Accuracy: {train_accuracy:.2f}%')
```

```
    # Validation step
    val_loss, val_accuracy = evaluate_model(model, test_loader, criterion)
    val_losses.append(val_loss)
    val_accuracies.append(val_accuracy)

    print(f'Epoch {epoch + 1}, Validation Loss: {val_loss:.4f}, Validation Accuracy: {val_accuracy:.2f}%')

    # Early stopping
    if val_loss < best_val_loss:
        best_val_loss = val_loss
        patience_counter = 0
    else:
        patience_counter += 1

    if patience_counter >= early_stopping_patience:
        print(f'Early stopping at epoch {epoch + 1}')
        break
```

❖ LSTM Model

```
# Define the LSTM model
class SentimentLSTM(nn.Module):
    def __init__(self, input_dim, hidden_dim, output_dim):
        super(SentimentLSTM, self).__init__()
        self.embedding = nn.Embedding(input_dim, hidden_dim)  # Convert indices to embeddings
        self.lstm = nn.LSTM(hidden_dim, hidden_dim, batch_first=True)
        self.fc = nn.Linear(hidden_dim, output_dim)  # Output layer for classification

    def forward(self, x):
        x = self.embedding(x)  # Ensure input is integer indices
        x, _ = self.lstm(x)  # Pass through LSTM
        x = self.fc(x[:, -1, :])  # Use the last output for classification
        return x

# Create a dataset and data loader (example data for demonstration)
# Assuming vocabulary size is 100, sequences are of length 10, and there are 5 classes
vocab_size = 100  # Vocabulary size
seq_length = 10   # Length of each sequence
num_samples = 100  # Number of samples
num_classes = 5   # Number of output classes

# Generate random integer inputs (representing token indices) and labels
inputs = torch.randint(0, vocab_size, (num_samples, seq_length))  # Random token indices
labels = torch.randint(0, num_classes, (num_samples,))  # Random class labels
```

```
# Create DataLoader
dataset = TensorDataset(inputs, labels)
train_loader = DataLoader(dataset, batch_size=16, shuffle=True)
valid_loader = DataLoader(dataset, batch_size=16)  # Using same data for validation

# Define model parameters
hidden_dim = 128
model = SentimentLSTM(vocab_size, hidden_dim, num_classes)

# Define loss function and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

# Training function
def train_model(model, train_loader, valid_loader, criterion, optimizer, num_epochs=10):
    train_losses = []
    val_losses = []
    train_accuracies = []
    val_accuracies = []
```

```
    for epoch in range(num_epochs):
        model.train()
        running_loss = 0.0
        correct_train = 0
        total_train = 0

        for inputs, labels in train_loader:
            inputs, labels = inputs.long(), labels.long()  # Ensure correct type
            optimizer.zero_grad()
            outputs = model(inputs)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()
            running_loss += loss.item()

            # Compute training accuracy
            _, predicted = torch.max(outputs, 1)  # Get class with highest probability
            correct_train += (predicted == labels).sum().item()
            total_train += labels.size(0)

        train_losses.append(running_loss / len(train_loader))
        train_accuracies.append(100 * correct_train / total_train)  # Convert to percentage
```

```
        # Validation phase
        model.eval()
        val_loss = 0.0
        correct_val = 0
        total_val = 0

        with torch.no_grad():
            for inputs, labels in valid_loader:
                inputs, labels = inputs.long(), labels.long()
                outputs = model(inputs)
                loss = criterion(outputs, labels)
                val_loss += loss.item()

                # Compute validation accuracy
                _, predicted = torch.max(outputs, 1)
                correct_val += (predicted == labels).sum().item()
                total_val += labels.size(0)

        val_losses.append(val_loss / len(valid_loader))
        val_accuracies.append(100 * correct_val / total_val)

        print(f"Epoch {epoch+1}/{num_epochs}, Train Loss: {train_losses[-1]:.4f}, Train Acc: {train_accuracies[-1]:.2f}%, "
              f"Val Loss: {val_losses[-1]:.4f}, Val Acc: {val_accuracies[-1]:.2f}%")

    return train_losses, val_losses, train_accuracies, val_accuracies
```

❖ Multilingual BERT MODEL

```
# Multiligual BERT model for sequence classification
config = BertConfig.from_pretrained(
    "bert-base-multilingual-cased",
    num_labels=len(label_mapping),
    hidden_dropout_prob=0.3,  # Change dropout rate here
    attention_probs_dropout_prob=0.3,  # Dropout in attention layers
    output_attentions=False,
    output_hidden_states=False,
)

# Load the model with the custom configuration
model = BertForSequenceClassification.from_pretrained(
    "bert-base-multilingual-cased",
    config=config
)

batch_size = 4

dataloader_train = DataLoader(dataset_train,
                              sampler=RandomSampler(dataset_train),
                              batch_size=batch_size)

dataloader_validation = DataLoader(dataset_val,
                                   sampler=SequentialSampler(dataset_val),
                                   batch_size=batch_size)
dataloader_test = DataLoader(dataset_test,
                                   sampler=SequentialSampler(dataset_test),
                                   batch_size=batch_size)
```

```
# Initialize optimizer and scheduler
optimizer = AdamW(model.parameters(), lr=5e-6, eps=1e-10, weight_decay=0.1)

epochs = 10

scheduler = get_linear_schedule_with_warmup(
    optimizer,
    num_warmup_steps=int(0.1 * len(dataloader_train) * epochs),  # 10% warmup steps
    num_training_steps=len(dataloader_train) * epochs
)

# Helper function for computing F1 score
def f1_score_func(preds, labels):
    preds_flat = np.argmax(preds, axis=1).flatten()
    labels_flat = labels.flatten()
    return f1_score(labels_flat, preds_flat, average='weighted')

# Set seeds for reproducibility
seed_val = 17
torch.manual_seed(seed_val)
torch.cuda.manual_seed_all(seed_val)
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)
```

```
# Freeze BERT parameters
for param in model.bert.parameters():
    param.requires_grad = False

# Define the evaluation function
def evaluate(dataloader_val):
    model.eval()
    loss_val_total = 0
    predictions, true_vals = [], []

    for batch in dataloader_val:
        # Convert all tensors to torch.long
        batch = tuple(b.to(device).to(torch.long) for b in batch)
        inputs = {
            'input_ids':      batch[0],
            'attention_mask': batch[1],
            'labels':         batch[2],
        }

        with torch.no_grad():
            outputs = model(**inputs)
```

```
        loss = outputs[0]
        logits = outputs[1]
        loss_val_total += loss.item()

        logits = logits.detach().cpu().numpy()
        label_ids = inputs['labels'].cpu().numpy()
        predictions.append(logits)
        true_vals.append(label_ids)

    loss_val_avg = loss_val_total / len(dataloader_val)
    predictions = np.concatenate(predictions, axis=0)
    true_vals = np.concatenate(true_vals, axis=0)

    return loss_val_avg, predictions, true_vals

# Parameters for gradient accumulation
accumulation_steps = 4

# Parameters for early stopping
patience = 2  # Number of epochs to wait before stopping
best_val_loss = float('inf')
early_stop_counter = 0

# Training loop with Gradient Accumulation and Early Stopping
for epoch in tqdm(range(1, epochs + 1)):
    model.train()
    loss_train_total = 0

    progress_bar = tqdm(dataloader_train, desc=f'Epoch {epoch}', leave=False)
    optimizer.zero_grad()  # Ensure gradients are cleared at the start of the epoch

    for step, batch in enumerate(progress_bar):
        # Convert tensors to torch.long and move to device
        batch = tuple(b.to(device).to(torch.long) for b in batch)
        inputs = {
            'input_ids':      batch[0],
            'attention_mask': batch[1],
            'labels':         batch[2],
        }

        # Forward pass
        outputs = model(**inputs)
        loss = outputs[0] / accumulation_steps  # Normalize loss by accumulation steps
        loss_train_total += loss.item() * accumulation_steps  # Track total loss

        # Backward pass
        loss.backward()
```

```
        # Perform optimizer step only after accumulation_steps
        if (step + 1) % accumulation_steps == 0 or (step + 1) == len(dataloader_train):
            torch.nn.utils.clip_grad_norm_(model.parameters(), 1.0)  # Gradient clipping
            optimizer.step()
            scheduler.step()
            optimizer.zero_grad()  # Clear gradients for the next accumulation cycle

        # Update progress bar
        progress_bar.set_postfix({'training_loss': '{:.3f}'.format(loss.item() * accumulation_steps)})

    # Average training loss for the epoch
    train_loss_avg = loss_train_total / len(dataloader_train)

    # Validation step
    val_loss_avg, predictions, true_vals = evaluate(dataloader_validation)
    val_f1 = f1_score_func(predictions, true_vals)

    print(f"Epoch {epoch}:")
    print(f"  Training Loss: {train_loss_avg:.4f}")
    print(f"  Validation Loss: {val_loss_avg:.4f}")
    print(f"  Validation F1 Score: {val_f1:.4f}")
```

```
    # Early stopping check
    if val_loss_avg < best_val_loss:
        best_val_loss = val_loss_avg
        early_stop_counter = 0
        torch.save(model.state_dict(), 'best_model.pt')  # Save the best model
    else:
        early_stop_counter += 1
        print(f"  No improvement for {early_stop_counter} epoch(s).")

    if early_stop_counter >= patience:
        print("Early stopping triggered. Stopping training.")
        break
```

❖ Sample of Dataset /not cleaned/

```
#df = pd.read_excel('Corpus_main_final.xlsx')
df = pd.read_excel("sample.xlsx")
#print("The number of rows is", df.shape[0])
print()
print(df.head(15))
df["Label"].value_counts().plot.bar()
```

	Label	Comments
0	Strongly Negative	#ጋላ ፈጣሪ እምነት %&%%ሀይማኖት 😔የለውም ጋላ ክብር አይወድለትም ጋላ...
1	Strongly Negative	….እራሱ መንግስት %ሊመታሲል ….ወይንም የኦሮሞ%%% ፅፈኛ ቄሮወች አደጋ...
2	Strongly Negative	ክፉ መርዝ አንተ ነህ ጅለንፍ😔!.....
3	Strongly Negative	^^(አማራ ብሎ) እስላም ትግራዋይ ብሎ %ዋቅፈታ የለውም😔:: …….
4	Strongly Negative	&&&አስተሳሰቡ በ19 ኛው,,,, ክፍለ ዘመን ላይ ተቸንክሮ የቀረው ብቸኛ...
5	Strongly Negative	$የነጃም &ደብተራ መድሀኒት ሰራ, ሲባል *እረ እሰይ እንዲያው ለመጀመሪያ...
6	Strongly Negative	የደቡብ ...ነንደሬን# አንገት ልክ እንደዚህ ቆርጬ 😔 ጆሮውን መያዝ ነው...
7	Strongly Negative	//የነጭ ሚስት ብታገባ ቃንቃህንማነትህን %%%ባህልህን አክብራ ታገባህለች...
8	Strongly Negative	ጋላ ሆድ, እጅ 😔 ጭንቅላት የለውም ስንል &&&በምክንያት ነው መንፈስ የ...
9	Strongly Negative	ነጃሚ// ማለት የአህያ ሽንት ከዘንዶ* መርዝ ያድናል 😔 ፀበል ነው…. ብ...
10	Strongly Negative	$ይድረስ #ለመላው 😔ኢትዮጵያያን ***ሀይለማርያም ደሳለኝ የምባል ወላሞ ...
11	Strongly Negative	አማራ ክልል ዩንቨርስቲ ከመማር ብቀርብህ ይሻላል&& እዛ ከሞት ተርፈህ ት...
12	Strongly Negative	ባንዳ//// ቅማንት ቅማንት$ ጠላት%%% ነው ስንል ዝም 😔ብለን አይድለም...
13	Strongly Negative	ህዝባችንን ,,,,$$$ቀርጥፎ እያበላ ያለውን በዘረኝነት ***ያበደውን የ...