Addis Ababa University

School of Information Science

Exploring Deep Learning for Amharic Sentiment Analysis: In the Context of Political Discourse in Ethiopia

By

Name: Dereje Wegayehu

ID Number: GSE/5537/13

A Thesis Submitted to the School of Information Science in Partial Fulfilment for the Degree of Masters of Science in Information Science and Systems (Information Science).

Addis Ababa, Ethiopia

June 2023

ADDIS ABABA UNIVERSITY SCHOOL OF INFORMATION SCIENCE

Exploring Deep Learning for Amharic Sentiment Analysis: In the Context of Political Discourse in Ethiopia

Dereje Wegayehu

Name and Signature of Members of the Examining Board:

Name	Title	Signature	Date
Michael M	~~Chairperson~~ Examiner		14/08/2023
Solomon Tgeda (PhD)	Advisor		24/08/2023
Mandwossen M.	Examiner		24/08/2023

Declaration

I, Dereje Wegayehu, hereby affirm the following MSC thesis entitled " Exploring Deep Learning for Amharic Sentiment Analysis: In the Context of Political Discourse in Ethiopia" is a creation of mine. I affirm that I have not submitted this work anywhere. For any degree or Master, and it does not contain any plagiarism or violation of any other person's intellectual property rights. Every source of data and content utilized in this thesis has been appropriately cited and acknowledged. My research led to the work that is described in this thesis, and any assistance received during the research has been accordingly acknowledged. Furthermore, I confirm that this thesis is not based on data or information obtained from any confidential sources or materials direct copy, and I have complied with the ethical standards and regulations governing the conduct of research at Addis Ababa University. I understand that any violation of the above statements may result in disciplinaryaction and the cancellation of my degree.

_____

Date

I have given my permission as the university advisor to submit this thesis for examination.

Advisor

Acknowledgment

I would like to sincerely thank my thesis advisor, Dr. Solomon Teferra, for his supportive advice and assistance during my research. His expertise, encouragement, and patience have had a crucial role in forming the direction and scope of this thesis. Dr. Solomon Tefera's thoughtful and insightful feedback, as well as his unwavering dedication to my academic and personal growth, has had a key role in helping me to achieve my goals. I am grateful for the countless hours he spent reviewing and critiquing my work, providing me with constructive feedback and suggestions that have greatly improved the quality of my research. Additionally, I am appreciative of my family and friends' encouragement and assistance along this trying path. Throughout my academic career, their support and devotion have served as a constant source of inspiration and drive. Finally, I am grateful to the academic community at large for their contributions to the field of Information Science. It is through their hard work and dedication that I have been able to learn and grow, and I am honored to be a part of this community. I want to thank you all for your encouragement, advice, and support. This thesis would not have been feasible without you.

Dereje Wegayehu

Table of Contents

List of Table

List of Figure

List of Abbreviations

Adagrad	Adaptive Gradient Algorithm
Adam	Adaptive Moment Estimation
ANN	Artificial Neural Networks
API	Application Program Interface
AUC-ROC	Area Under the Curve of Receiver Operating Characteristic
Bi-LSTM	Bidirectional Long Short-Term Memory
BOW	Bag of Words
CBOW	Continuous Bag of Words
CNN	Convolutional Neural Network
GANs	generative adversarial networks
GloVe	Global Vectors for Word Representation
GRU	Gated recurrent unit
HAC	Hierarchical Agglomerative Clustering
HAL	Hyperspace Analogue to Language
LSA	Latent Semantic Analysis
LSTM	Long Short-Term Memory
ME	Maximum Entropy
NB	Naive Bayes
NLP	Natural Language Processing
ReLU	Rectified linear unit
RL	Reinforcement learning
RMSprop	Root Mean Square Propagation
RNN	Recurrent Neural Networks
RQ	Research questions
SGD	Stochastic Gradient Descent
SVM	Support Vector Machines
TF-IDF	Term frequency-inverse document frequency
Word2Vec	word to vector

Abstract

Deep learning, a potent machine learning technique, learns several layers of representations or characteristics of the data to produce cutting-edge prediction outputs. In addition to its popularity in a variety of fields of application, deep learning has been successfully used in sentiment analysis on a large scale in recent years. Nowadays, social media plays a significant role in swaying public opinion in favor of or against a government or organization. Therefore, an effective strategy is necessary to analyze the mood of any social media posting.

This research paper explores the application of deep learning that automatically extracts sentiments from Amharic text, Sentiment analysis for Amharic in the political domain. Social media platforms have become crucial for shaping public opinion and sentiment toward governments and organizations. A lot of research is being conducted for the development of sentiment analysis in different languages. However, sentiment analysis for under-resourced languages like Amharic has received little attention.

The study proposes a supervised deep-learning for sentiment classification and demonstrates its effectiveness in analyzing Amharic language data. Data was extracted from FANA broadcasting corporation and Ethiopian Prime Minister ABIY Ahmed Ali's official Facebook page and manually annotated into positive and negative classes. And this study examines the effectiveness of using a different training dataset and pre-trained word embedding and tokenizers experiment on purposes and improving the text categorization effectiveness of deep learning models.

The study compares the performance of five different deep learning models, including LSTM, CNN, RNN, GRU, and bi-LSTM, with various architectures and parameters. The experiments reveal that the GRU model achieved an accuracy of 82.49%, the LSTM model achieved the highest accuracy of 79.41% while the CNN model attained an accuracy of 77% the bi-LSTM model obtained an accuracy of 74.2% and the RNN model achieved an accuracy of 73.3%. These results suggest that using a large training dataset, pre-trained embedding, pre-trained Tokenizer, and Adams optimizer can significantly increase text categorization performance using deep learning models.

Keywords – Sentiment analysis, Amharic language, deep learning, politics.

Chapter One

1. Introduction

1.1. Background

Sentiment analysis is the process of controlling opinions, feelings, and subjective content (Wankhade et al., 2022). As it looks at various reviews and tweets, Sentiment analysis helps understand data about public opinions. It is a tried-and-true technique for forecasting several significant events, such as the box office performance of films and general elections (Schmidt et al., 2022). Public reviews, which are available on a variety of websites like Amazon, are utilized for assessing a certain entity, such as a person, a product, or a location. One can classify opinions as negative, positive, or neutral.

The sentiment analysis's objective is to automatically pinpoint the expressive tone of user assessments (Yenkikar et al., 2022). According to (Katrekar, 2019), sentiment analysis can be conducted at different levels: document level, sentence level, and aspect level. At the document level, the sentiment of the entire document is classified as positive, negative, or neutral. Similarly, at the sentence level, each sentence is categorized accordingly. However, at the aspect level, the analysis focuses on identifying the sentiment towards specific features or aspects within the document or sentence.

This study focuses on sentiment analysis with a deep learning method; Deep learning algorithms are capable of automatically learning hierarchical representations of data, which allows them to capture intricate relationships and dependencies within the text. Sentiment analysis involves understanding the subjective opinions, emotions, and attitudes expressed in text, which often require a deep understanding of context, semantics, and even subtle nuances. Deep learning models excel in this regard, as they can learn intricate features and representations of the text through multiple layers of processing.

The demand for sentiment analysis has grown as a result of the rising requirement for analyzing and organizing unstructured data that is obtained from social media in the form of hidden information (Alsayat, 2022). People now primarily convey their daily activities, reactions, and emotions through social media platforms. The most popular type of social media is blogs and microblogs (Marouf et al., 2022). Users share ideas, discussions, and thoughts on blogs, which are informal websites on the internet (Fraumann & Colavizza, 2022). Microblogs, on the other hand, are smaller blogs with brief posts up to a few signs (Ali et al., 2020). Each of them has the option to list entries

in reverse chronological order, with the most current news at the top. They are resources that let people discuss and comment on information that other users have shared. They stand out for their modernism and liveliness.

As of December 2021, there were 21,147,255 Internet users or 17.7% of the Ethiopian population. 6.3% of Ethiopia's population of 7,535,700 users of Facebook in January 2022 (*10 Top Ethiopia Facebook Pages 2023 — AllaboutETHIO*, n.d.) . The majority of the citizens who use the internet participate in many topics and provide input in their mother tongue on those that have been brought up by their friends, blogs, media, or newspapers. However, there hasn't been enough work done on deep learning-based sentiment mining for the political domain utilizing opinionatedAmharic text.

1.2. Statement of the Problem

The widespread adoption of social media has transformed people's lives, particularly in Ethiopia, where it serves as a vital platform for political activism and information exchange. As social media usage continues to increase, it becomes crucial for legislators, businesses, and community service providers to recognize the significance of gathering and analyzing data from these platforms. Such analysis can provide valuable insights into public opinion regarding services, products, rules, and regulations, enabling decision-makers to make informed choices aligned with the needs and desires of the general public. Integrating social media data analysis into decision-making processes can bridge the gap between people and decision-makers, leading to a more responsive and effective governance system.

However, analyzing sentiments, feedback, and expressions written or expressed in natural languages, especially for under-resourced languages like Amharic, poses a challenge. Understanding and interpreting sentiments and feedback shared on social media, particularly in specialized domains or topics, require specialized knowledge and context understanding. While previous attempts have been made to develop Amharic sentiment analysis models have been developed by researchers utilizing machine learning methods like Support Vector Machines (SVM), Naive Bayes (NB), and Convolutional Neural Networks (Kindeneh, 2020; Neshir et al., 2021; Saron & Zelelew, 2021) despite prior attempts to develop emotion analysis models and opinion-mining techniques for the Amharic dialect.

The context of the text is one of the major variables that affect sentiment analysis. When utilized in a different context, words, and phrases that could have a positive or bad

connotation in one area could have a different sentiment orientation (Liu et al., 2020).

The focus of the analysis of sentiment has shifted to context-dependent variation in sentiment orientation. To increase the quality of sentiment analysis, researchers are using increasingly advanced algorithms and approaches to recognize and examine the context in which words and phrases are employed. Sentiment analysis can offer a more sophisticated assessment of the text's emotional intensity by taking context into account. Words are powerful, but their power is not solely derived from their definitions. When employed in several phrase situations, the same word may elicit various emotions.

The feature engineering process is not something that traditional machine learning algorithms can do on their own, and thus, may struggle with understanding the nuances of language. However, deep learning algorithms are capable of performing semantic composition, it entails creating a text unit's vector of representation quickly and in a low-dimensional space by merging its finer-grained parts or entities. As a result, the algorithm is better able to discern the sentiment of the text by understanding the meaning behind words and how they relate to one another in a sentence.

To retrieve contextual information from feature sequences in foreign languages, some academics have investigated sentiment analysis utilizing deep learning approaches (Balakrishnan et al., 2022; Guo, 2022; Iqbal et al., 2022; Kaur et al., 2021; Reviews et al., 2022). The results of this research, however, cannot be immediately transferred to the Amharic dialect because of the distinctive features that set it apart from the other investigated languages. Furthermore, The Amharic language lacks sufficient context-based sentiment analysis research employing deep learning. To best capture contextual information for sentiment analysis of Amharic texts,

This research aims to explore the application of deep learning algorithms for sentiment analysis in Amharic texts by focusing on the retrieval of contextual information from feature sequences. Amharic is a complex language with unique linguistic characteristics, and sentiment analysis in this language presents specific challenges that require specialized approaches.

The study will investigate various deep learning algorithms and techniques that can effectively capture the contextual information present in Amharic texts. By leveraging deep learning techniques, the research aims to enhance the efficiency of sentiment analysis in Amharic texts. Traditional sentiment analysis methods often struggle to capture the nuances and intricacies of sentiment expressed in Amharic due to the language's rich morphology, syntax, and context-dependent nature. Deep learning

algorithms, with their capacity to learn complex representations and handle large-scale datasets, offer promising solutions to overcome these challenges.

The investigation will involve training deep learning models on a substantial collection of Amharic texts annotated with sentiment labels. These models will learn to extract relevant features from the input sequences and capture the contextual information crucial for accurate sentiment analysis. The research will compare and evaluate the performance of different deep learning architectures in terms of their efficiency in sentiment analysis tasks, such as sentiment classification or sentiment polarity detection.

The findings of this research are expected to contribute to the advancement of sentiment analysis in Amharic texts by providing insights into the effectiveness of various deep learning approaches. By identifying the most suitable algorithms and techniques, the research aims to enhance the accuracy and efficiency of sentiment analysis applications in the Amharic language.

1.3. Research Question

The research study will address the following research questions (RQ).

RQ1: Which deep learning models are most effective for Amharic sentiment analysis?

RQ2: Which deep learning model performs the best when analyzing sentiment in Amharic, compare to other state-of-the-art models?

RQ3: What effects do the amount of the training data, Transfer Learning, and optimization strategies have on the Amharic SA models?

1.4. The Objective of the Study

1.4.1. General Objective

This research is aimed to investigate the application of DNN approaches for the development of Amharic sentiment analysis political texts.

1.4.2. Specific Objectives

The following particular goals are the focus of this research work.

a) Collect Amharic social media reviews.
b) Annotating Amharic reviews.
c) Develop an Amharic sentiment analysis model using supervised learning of different DNN approaches.
d) Compare the performance of the models developed.
e) Interpreting the results of the comparison and drawing conclusions.

1.5. Scope

The scope of this thesis is to create an analysis model for Amharic comments that are only written in Ethiopic characters (Fidel). The study is limited to textual data, and other forms of media such as audio, images, and videos are excluded. The study's concentration is on sentiment analysis; it does not cover other aspects of natural language processing. The study will also not consider idiomatic expressions, symbolic language, and sarcasm, as these are complex forms of language that require a deeper level of linguistic analysis.

The sentiment polarity will be limited to two labels, positive and negative and using only supervised learning techniques, and the dataset will be manually labeled by human annotators. Standard assessment metrics will be used to assess the model's performance.

1.6. Methodology

The methodology for examining the viability of deep learning-based sentiment analysis on Amharic comments within the context of Ethiopian political debate entails several crucial steps.

Introduction: We will give a summary of the research topic and its importance in the project's introduction. We will also outline the main research question that we seek to answer through this study. Additionally, we will specify the general and specific objectives of the research, which will guide our investigation and analysis. In addition, we will emphasize the study's scope, which will outline the confines of the investigation.

Literature Review: is a critical aspect of any research study, as it allows researchers to build upon existing knowledge and gain insights into the theoretical background of their field. In this work, Amharic sentiment analysis which examines sentiment in the Amharic language is the main topic of interest. The literature review will examine the particular difficulties involved in analyzing sentiment in Amharic, including word representations and data accessibility, to get a thorough understanding of this subject. The review will especially concentrate on works relating to Amharic sentiment analysis in addition to looking at research on sentiment analysis in other foreign languages. To create a model that can successfully assess sentiment in Amharic, the finest methodologies and tools will lastly be examined, picked, implemented, and/or updated. The literature review will act as the research study's basis, offering a thorough understanding of the subject and directing the creation of an efficient model.

Data Collection: The datasets (comments) for the experiment were automatically

gathered from the official Facebook pages of Dr. Abiy Ahmed Ali and the FANA television company. Data was gathered using a Python scraper script and Facebook's online comment exporter tool. The reason for selecting this tool was its ability to gather large amounts of comments efficiently and quickly. The selection of FANA broadcasting corporation and Dr. Abiy Ahmed Ali official Facebook page as sources for data collection was based on their popularity and relevance to the research question.

Analyze and Design: In this project, creating a sentiment analysis model for Amharic comments in the context of Ethiopian political debate is the main goal. A deep learning strategy is used to do this, with various well-known algorithms; including Long Short-Term Memory (LSTM), Bi-Directional LSTM Network (Bi-LSTM), Convolutional Neural Network (CNN), Basic Recurrent Neural Network (RNN) model, and Gated Recurrent Unit (GRU) is used to experiment with various architectures and hyperparameters to discover the optimum model for the given problem. The deep learning models will be trained on an appropriate dataset, and they will be modified to achieve the best level of accuracy.

Evaluation: The effectiveness of an experiment's evaluation metrics determines whether it is a success. In this experiment, many metrics will be used to assess the models' performance, among which accuracy is the primary one. An important evaluation indicator, accuracy assesses how well the model correctly predicted the output. The performance of the model improves with accuracy. In addition to accuracy, the performance of the models may also be assessed using metrics like precision, recall, F1-score, and AUC-ROC. These measurements provide a more comprehensive picture of the model's performance and aid in determining each model's advantages and disadvantages. As a result, choosing the right evaluation measures is essential to the outcome of every experiment.

Tools: Machine learning models were developed for the experiment, and for this purpose, Python, a high-level programming language, was used. Python offers a wide range of machine-learning packages and tools, making it a popular choice among researchers and practitioners. To build the machine learning models, two popular deep learning libraries, PyTorch and Keras, were used. While Keras is a high-level neural networks API that provides a user-friendly interface for creating deep learning models, PyTorch is an open-source machine learning toolkit that supports the development of neural networks and tensor computations. Along with these libraries, other important libraries such as Pandas, NumPy, and Scikit-learn were also used. Panda is a toolkit for

data manipulation that offers simple data structures for data analysis, a library for numerical computing called NumPy supports numerical operations on big matrices and arrays. A machine learning library called Scikit-learn offers many methods for clustering, regression, and classification. These libraries worked together to offer the tools and capabilities needed to preprocess the data, train and validate the models, and evaluate the models' performance.

Chapter Two

2. Literature Review

In this study's part, many previous works in the research field are examined, and important ideas related to those works are presented. The idea also discusses and reviews several sentiment classification approaches, sentiment feature extraction, level of sentiment analysis, and evaluation methodologies. Social media overviews, also included are a concept of sentiment analysis and a study of how the Amharic languages behave.

2.1. Theoretical Background

2.1.1. Sentiment Analysis

Sentiment analysis is a method for identifying and extracting subjective information from text, such as views, attitudes, emotions, and sentiments. It is a type of natural language processing. It is frequently utilized in many different applications, such as market research, brand reputation management, customer feedback analysis, and social media monitoring. Text preparation, extraction of features, classification of sentiment, and evaluation are some of the common processes in the sentiment analysis process.

The original text information is cleaned up and converted into an analysis-ready format during test preparation. Finding pertinent textual elements, such as words, phrases, or other language trends, that might be predictive of sentiment is known as feature extraction. Giving the text a sentiment score or label, which can be positive, negative, or neutral, is the task of the sentiment classification phase. Several methods, including rule-based approaches, machine learning algorithms, and deep learning models, can be used to do this (Rodriguez & Spirling, 2022).

In the evaluation stage, the quality and efficiency of the emotion analysis system are evaluated, frequently utilizing measures like precision, recall, and F1-score. Sentiment analysis may help businesses and organizations make better decisions and increase performance by offering insightful information about feedback from customers, online community trends, and other types of textual data (Manish, 2020).

2.1.1.1.Levels of Sentiment Analysis

Typically, sentiment analysis is done at three different levels (Mehta & Pandya, 2020):

Document-level Sentiment Analysis: This kind of analyzing sentiment is concerned with figuring out the general attitude of a substantial text or document, like a blog post, review, or tweet. It conveys a fundamental impression of the text's tone, whether it is

neutral, negative, or positive.

Sentence-level Sentiment Analysis: The sentiment of specific phrases in a document or text is examined at this phase of sentiment analysis. It provides a more detailed understanding of the sentiment portrayed in the text because different sentences in identical documents may indicate various emotions.

Aspect-level Sentiment Analysis: This form of analysis of sentiment focuses on determining the attitude toward particular qualities or characteristics of a good, service, or person referenced in the text. It involves analyzing the sentiment of individual phrases or clauses that refer to a particular aspect, such as price, quality, or customer service. This type of analysis is particularly useful in understanding the strengths and weaknesses of a product or service and can help businesses to improve their offerings.

2.1.1.2.Sentiment Analysis Approaches

Sentiment analysis is a type of natural language processing (NLP) that involves the use of computational algorithms to determine the emotional tone of a piece of text, such as positive, negative, or neutral. There are several approaches to sentiment analysis, including:

The lexicon-based approach: To determine the general feeling of a piece of text, sentiment analysis uses predetermined glossaries of words and their corresponding emotions (positive, negative, or neutral) (Biele et al., 2022). In this method, a set of terms and their corresponding sentiment ratings are found in a dictionary or sentiment lexicon. A word's sentiment score might be good, negative, or neutral.

The lexicon can be created by hand or automatically via statistical methods or algorithms based on machine learning on a sizable corpus of text. To determine the text's sentiment, the terms in the text are contrasted with those in the sentiment lexicon. The overall sentiment score for the text is then calculated by adding the scores for sentiment for the matched terms.

There are several approaches to calculating the sentiment score, like adding up the emotional scores of the matched words, averaging the sentiment scores, or utilizing more complex techniques like TF-IDF weighting or regression analysis. The lexicon-based technique has the benefit of not requiring data that is labeled for training, making it simple to use and adaptable to diverse domains and languages. This approach has some difficulties, too, as the sentiment of a word might vary based on how it is used. It may also be unable to convey the subtleties and contextually-dependent nature of sentiment. It is crucial to employ a lexicon of emotions that is suitable for the text's particular domain

and context.

The machine learning-based approach: Machine learning algorithms are trained on a dataset that is labeled with text documents and the associated sentiments to perform sentiment analysis. Based on the patterns it has discovered from the training data, the algorithm learns to categorize fresh texts into positive, negative, or neutral feelings (Xu et al., 2022).

Neural Networks (NNs): are particular kinds of machine learning algorithms that take their cues from how the human brain is organized and operates. NNs consist of interconnected layers of neurons that process input data and produce output predictions (Sah, 2020). The input layer, hidden layer(s), and output layer are the three different sorts of layers that make up a conventional neural network. The input layer takes the data as input, which is then processed by one or more hidden layers to create the prediction of the outcome from the output layer.

The computing unit known as a neuron in a neural network conducts a weighted sum of its inputs, assigns an activation function to the total, and sends the outcome to the following layer. By utilizing an optimization algorithm like gradient descent during training, the biases and weights of the neurons are learned. The activation function of a neuron determines its output, which can be binary (0 or 1), continuous (between 0 and 1), or any other function that maps the input to an output.

The sigmoid function and the rectified linear unit (ReLU) function are the two activation functions that are employed the most frequently. To reduce the error between the anticipated result and the actual output, the neural network modifies the biases and weights of the neurons during training. Using an optimization method, this procedure, known as back propagation, calculates the gradient of the error for the biases and weights and updates them.

Several machine learning tasks, such as classification, regression, and speech and picture recognition, can be performed using neural networks. For jobs involving high-dimensional data and non-linear correlations among features and the target variable, they are especially effective. The key benefits of neural networks are their capacity to learn intricate patterns in data, versatility in processing various input data sources, and good generalization to new data. To avoid over fitting, they may need a lot of data and can be computationally challenging to train.

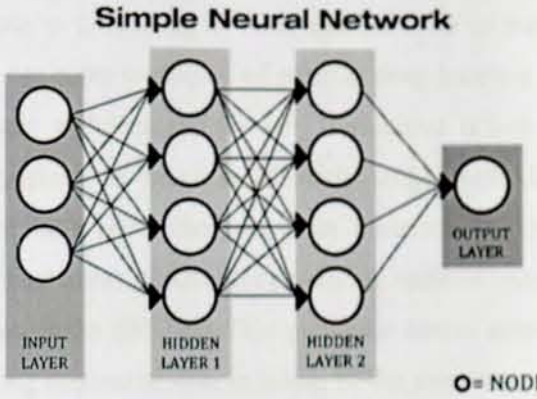


Figure 1: A simple three-layered feed-forward neural networks

Model evaluation: A distinct set of text documents is used to test the efficacy of the trained model in predicting sentiment. A few examples of evaluation metrics are the ROC curve, F1-score, recall, accuracy, and precision.

The classification of freshly written material into either positive or negative emotions can be done using a machine learning approach after it has been trained and evaluated. As the technique learns to understand the patterns and interactions between the data and the sentiment labels, one benefit of a machine learning-based method is that it can record the convoluted and dependent upon-context nature of sentiment. However, this method needs a lot of labeled training data, and it could struggle with topics or dialects that are quite dissimilar to the training data.

Deep learning: is a branch of machine learning that makes use of deep neural networks for data-driven learning. Deep neural networks, which have several hidden layers, are neural networks that can recognize more intricate patterns and connections in data. Deep learning has proven to be remarkably effective in a variety of applications, including speech recognition, visual processing, natural language processing, and gaming. For instance, advances in picture classification, object recognition, and image segmentation exercises have been made possible by deep learning. It has also been used to develop models of language that can generate text that sounds natural and speech recognition software that precisely records spoken words.

Large volumes of labeled data are often used for training deep learning models, and methods based on gradients, like stochastic gradient descent, are used for optimization. To reduce a loss function, which calculates the gap between the anticipated output and the actual output, the neural network's weights and biases are adjusted during the training phase.

Convolutional neural networks (CNNs) for processing images, RNNs (recurrent neural

networks) for consecutive processing of data, and networks of transformers for natural language processing are some examples of popular deep learning architecture. In their respective fields, these architectures are very successful (Chen et al., 2021). Deep learning has also sparked the creation of cutting-edge methods like reinforcement learning (RL) for teaching agents how to make decisions in complex situations and generative adversarial networks (GANs) for producing realistic images.

Recurrent neural networks (RNNs): This particular neural network architecture was created with processing sequential data in mind. RNNs preserve a hidden state that can store knowledge about earlier inputs in the sequence, in contrast to feed-forward neural networks which analyze input data in a single pass (Bagchi, 2022). Natural language processing, audio identification and time series prediction are just a few of the sequential data-related applications where RNNs have demonstrated outstanding performance. They are particularly effective for jobs involving variable-length sequences because the hidden state may adjust to the input's length.

A recurrent layer, that analyzes the initial information consecutively and keeps an unnoticed state that changes at each time step, is the fundamental component of an RNN. A non-linear activation function, such as the hyperbolic tangent or the corrected linear unit (ReLU), is commonly used to pass the hidden state through. The ultimate hidden state or the state that is invisible at every step in the process can be used to calculate the RNN's output.

The long short-term memory (LSTM) network is a popular variation of the standard RNN architecture that was created to deal with the vanishing gradient problem that can arise during the training of deep RNNs. The LSTM network employs a memory cell with long-term information storage capacity and three gates to regulate data flow through and out of the cell.

The gated recurrent unit (GRU), a different form that is comparable to the LSTM but employs fewer specifications and is theoretically more effective, is another option. Creating a loss function that evaluates the discrepancy between the output that is predicted and the actual output is a common step in training an RNN.

Gradient descent is then used to optimize the network's parameters. The network parameters' gradients of the loss function are calculated using the back propagation algorithm, and these gradients are then utilized to alter the settings in a way that minimizes the loss.

RNNs are a potent and adaptable kind of neural network which can identify intricate

links and patterns in sequential data. They are highly suited for a variety of applications in time series analysis, speech recognition, and natural language processing due to their capacity to represent the temporal changes of a sequence.

Long Short-Term Memory (LSTM): a specific type of recurrent neural network (RNN) created to deal with the issue of vanishing gradients that can happen during the training of deep RNNs. Three gates that regulate the flow of data through and out of the memory cell are used in LSTM networks together with a cell of memory that can retain data for extended periods. The entry gate, the forget gate, and the outcome gate are the three gates of an LSTM network. These gates control the information flow into and out of the storage cell, enabling the LSTM to precisely retain and access information as necessary (Yang et al., 2021).

How much fresh data should be inserted into the memory cell is decided by the input gate. It processes the prior hidden state and the current input via a sigmoid activation function to yield a number between 0 and 1. The amount of fresh data that has to be stored in the memory cell is indicated by this value. Which data should be removed from the memory cell is decided by the forget gate. It processes the prior hidden state and the current input through a sigmoid activation function to yield a number between 0 and 1. The amount of data that should be removed from the memory cell is represented by this value. Which information to be output from the memory cell is decided by the output gate. It accepts as input the present input, the prior concealed state, the present values of the memory cell, and the current input.

To generate a number between -1 and 1, these values are put via sigmoid and hyperbolic tangent activation functions. The output of the LSTM at the current time step is represented by this number. According to the results of the input gate, forget gate, current input, and prior concealed state, the storage cell is updated. The prior contents of the memory cell and the incoming input are combined linearly and graded via the gate for input and forget gate, respectively, to determine the most recent contents of the memory cell. Many different applications, including speech recognition, image captioning, and natural language processing, benefit greatly from the use of LSTM networks. They are highly suited to activities involving long-term dependencies due to their capacity to deliberately keep and retrieve data over extended periods.

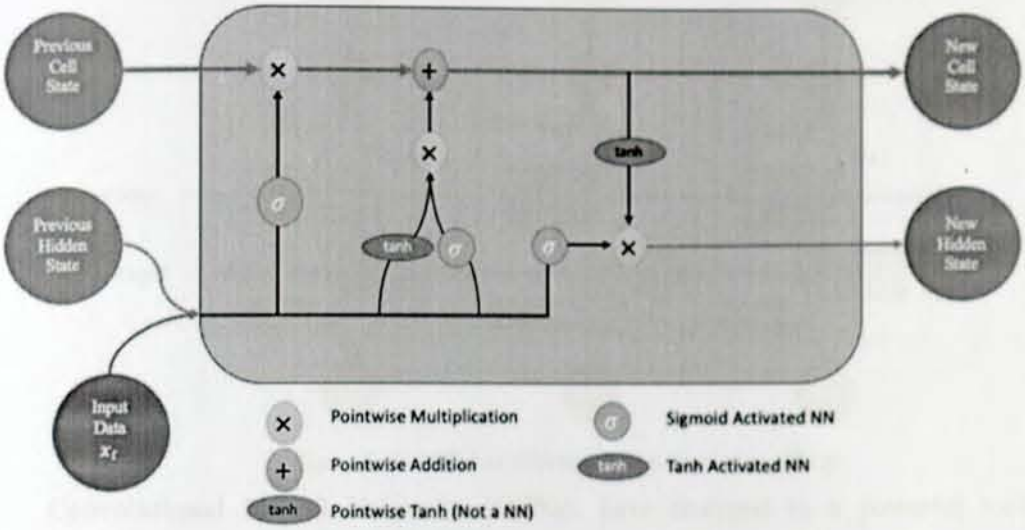


Figure 2: Architecture of LSTM (Yang et al., 2021)

Bidirectional Long Short-Term Memory (Bi-LSTM): is a kind of LSTM network that employs backward and forward processing of the input sequence. Bi-LSTMs work well for tasks like sentiment analysis or named entity recognition that need a grasp of the context of a word or phrase within a sentence or paragraph. In a Bi-LSTM network, a forward LSTM layer processes the input sequence first, processing it from the first input to the last input in a forward direction.

A backward LSTM layer receives the output of the forward LSTM layer and processes the sequence from the final input to the first input in the opposite way (Choudhary et al., 2022). A final output sequence that includes data from each of the forward and backward contexts is created by concatenating the output from the forward and backward LSTM layers. Due to its ability to record interdependence in both directions, the Bi-LSTM can now have a more thorough knowledge of the circumstances in which a word or phrase is used.

A second LSTM layer capable of processing the pattern in the opposite direction is added to the regular LSTM's design to create a Bi-LSTM. Each LSTM layer has its memory cell and gates, and the final output sequence is created by concatenating the output from both layers. Bi-LSTMs are effective for many different NLP applications, including sentiment analysis, speech recognition, and machine translation. Given that they can record interconnections in every direction and give a more thorough knowledge of the input sequence, they are especially helpful in jobs where context is crucial.

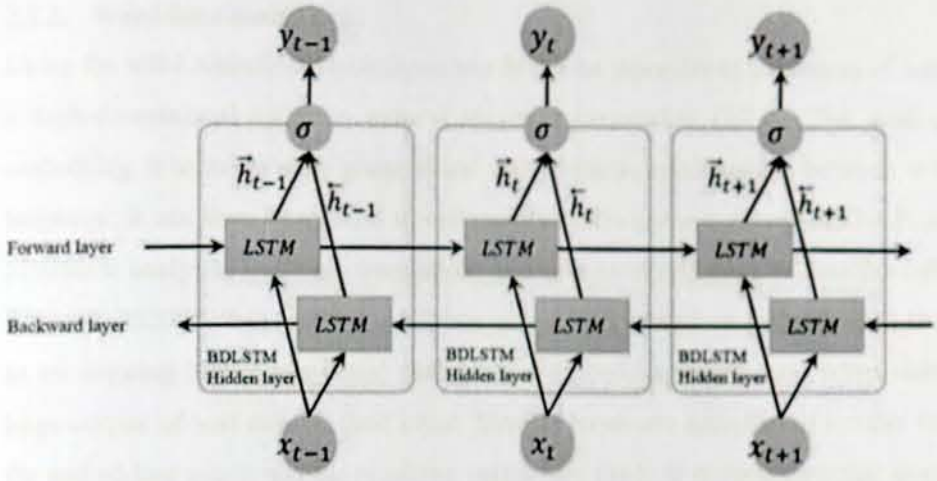


Figure 3: Bi-LSTM architecture (Choudhary et al., 2022)

Convolutional Neural Networks (CNNs): have emerged as a powerful tool for sentiment analysis in text post comments. Originally developed for image recognition tasks, CNNs have shown great potential in analyzing textual data due to their ability to capture local patterns and hierarchical features. In the context of sentiment analysis, CNNs can effectively identify and extract relevant features from the text, such as words, phrases, or even sentiment-specific patterns, by applying convolutional filters across different parts of the input. This allows the model to learn discriminative representations that capture the sentiment information within the comments. By combining these learned features with subsequent layers, such as pooling and fully connected layers.

CNNs can effectively classify the sentiment of text post comments, enabling applications in areas like social media analysis, customer feedback processing, and market sentiment tracking. The adaptability of CNNs, coupled with their ability to automatically learn meaningful representations, make them a valuable tool in sentiment analysis tasks, improving our understanding of people's opinions and emotions expressed in the text (Wangpoonsarp et al., 2020).

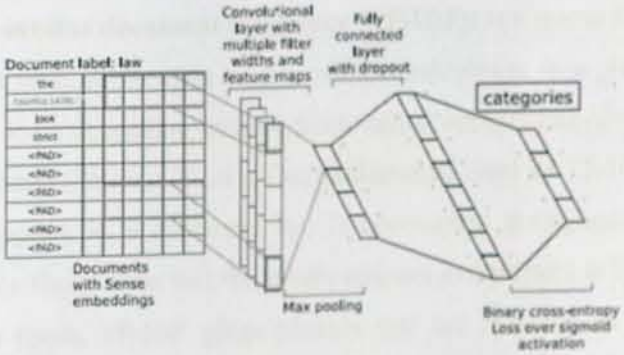


Figure 4: Convolutional Neural Network (CNN) model for text classification (Wangpoonsarp et al., 2020)

2.1.2. Word Representations

Using the word embedding technique, words can be represented as vectors of numbers in a high-dimensional space in natural language processing (NLP). The goal of word embedding is to capture the grammatical and syntactic relationships between words in a language; it can then be applied to enhance the effectiveness of various NLP tasks like sentiment analysis, language translation, and text categorization. (Camacho-collados & Pilehvar, 2020). When learning a relation across every word on the collection to a vector in an ongoing high-dimensional space, word embedding algorithms often start with a huge corpus of text data as their input. Similar terms are assigned to similar vectors in the embedding space, and the resulting vectors are made to encode semantic associations between words.

The vectors for "king" and "queen" should, for instance, be closer together than those for "king" and "apple" in a well-trained word embedding. There are several popular algorithms for generating word embeddings, such as Word2Vec, GloVe, and fastText. Once a word embedding has been trained, it can be used as a feature representation for downstream NLP tasks, either by using the word vectors directly or by feeding them as input to a machine-learning model (Science, 2021).

2.1.2.1.Approaches to word embedding

In processing natural language (NLP), word embeddings can be approached in various ways, such as:

2.1.2.1.1. Count-based methods

These techniques work by counting the times words appear together in a huge body of text. The term frequency-inverse document frequency (TF-IDF) weighting scheme, which provides more weight to words that are more informative and less common in the corpus, is the most often used count-based strategy. Latent Semantic Analysis (LSA) and Hyperspace Analogue to Language (HAL) are two further count-based techniques.

Term frequency-inverse document frequency (TF-IDF): is a metric that quantifies the weight given to a term in a corpus or document. Each phrase in a piece of writing or corpus is given a score depending on how frequently it appears and how relevant it is to the corpus as a whole (Science, 2021). The fundamental tenet of TF-IDF is that a term that frequently appears in a document but is uncommon throughout the corpus has greater significance than a term that frequently appears everywhere in the document and the corpus. As a result, TF-IDF gives phrases that are common in a document but uncommon across the board a high weight (Jalilifard et al., 2021). The term frequency

(TF), a measure of how frequently the term shows up in the document, and the inverse frequency of the document (IDF), which represents how important the phrase is over the entire corpus, are combined to provide the TF-IDF score for a term in a document. The formula for TF-IDF is TF-IDF = TF * IDF, where TF = log (total number of documents/number of documents with a term in it) and IDF = (number of times term appears in document) / (total number of terms in the document). The TF-IDF is frequently employed for information retrieving, data mining, and natural language processing to classify documents based on their content, rank the value of documents to a query, and extract key phrases from a corpus.

Latent Semantic Analysis (LSA): is a technique used in the processing of natural languages to examine the relationships between a group of files and the words they contain. Based on the connections between words and texts, it is a mathematical approach for encoding meaning. LSA works by constructing a matrix of word frequencies across all the documents in a corpus. This matrix is then decomposed into a set of smaller matrices that represent the underlying patterns or "latent" relationships between the words and documents. These patterns are referred to as "latent semantic factors". The high-dimensional word frequency matrix is converted into a lower-dimensional space using a technique called LSA.

This makes it possible to spot patterns and connections among words and files that would not have been obvious in the original high-dimensional space. LSA can be used for a range of applications, such as document classification and information retrieval, and automatic summarization. In information retrieval, LSA can be used to improve search results by identifying documents that are semantically related to a query, even if they do not contain the same keywords. In document classification, based on the underlying semantic structure of the texts, LSA can be used to group the documents. In automatic summary, LSA can be used to identify the most important sentences in a document based on their semantic content. LSA is a powerful technique for analyzing and extracting meaning from large collections of text data.

The Hyperspace Analogue to Language (HAL): is a natural language processing technique developed by Stuart C. Shapiro in the 1980s. HAL is a knowledge representation technique that represents the meanings of words and phrases based on the contexts in which they appear. The method is predicated on the notion that a word's meaning is connected to the words that frequently appear in similar settings to it. To build a semantic network in HAL that illustrates the relationships between words, a

sizable corpus of text is processed. Each word is represented as a vector of co-occurrence rates with other terms in the corpus to create the network. A collection of clusters representing various semantic domains is produced when the vectors are grouped according to their commonalities. HAL can be used for a number of the processing of natural languages, such as information retrieval, text categorization, and text production, once a semantic network has been built.

In information retrieval, HAL can be used to find documents that are semantically related to a query, even if they do not contain the same keywords. In text classification, HAL can be used to classify documents based on their underlying semantic content. Using the connections amongst words in the semantic network, HAL can be used to generate new text. HAL is an effective method for representing phrases and words based on the settings in which they are used. It is effective at capturing the subtleties of language use and has been employed in many natural language processing applications.

2.1.2.1.2. Prediction-based methods

These techniques work by learning a model to anticipate a word's context from its surrounding terms. The most well-known prediction-based technique is Word2Vec, which forecasts the setting of a word-based N and the words around it using a shallow neural network. The Global Vectors for Word Representation (GloVe), another well-known prediction-based technique, employs matrix factorization to develop word embedding based on the co-occurrence statistics of words (Camacho-collados & Pilehvar, 2020).

Word2Vec: is a method of natural language processing used to learn word distributions based on the circumstances in which they appear. It is a neural network-based strategy that Tomas Mikolov and his coworkers at Google introduced in 2013. Utilizing the words that commonly appear in contexts that are comparable to it, one can use a massive corpus of literature to discover the vector representation for each word, Word2Vec uses the basic principle of word co-occurrence. This is done by training a neural network to predict the likelihood that a word will emerge in a specific circumstance. Word embeddings are the vector representations created by the neural network once it was trained on a significant corpus of text.

Word2Vec primarily uses the Continuous Bag-of-Words (CBOW) model and the Skip-Gram model. The Skip-Gram model forecasts the other context words according to the current word, whereas the CBOW model predicts the word at hand depending on the adjacent context words. It is learned by both models that words have vector

representations that preserve their grammatical and semantic connections to other phrases in the corpus. In language processing applications like text categorization, retrieval of data, and machine translation, the generated word embeddings are frequently employed as features. Word2Vec has been utilized in a wide range of applications and is successful at catching the underlying semantic links between words. Word2Vec is a potent method for learning word representations that are dispersed and capture the semantic and syntactic connections between terms in a corpus of text.

Global Vectors for Word Representation (GloVe): This method of word embedding learning in natural language processing captures the two types of syntactic and semantic links between words. Experts at Stanford University introduced GloVe in 2014. The foundation of GloVe, like Word2Vec, is the concept of acquiring vector illustrations for words according to their interaction statistics in a significant corpus of literature. Contrary to Word2Vec, which employs a neural network strategy, GloVe makes use of a matrix factorization technique. GloVe's fundamental premise is to generate a vector illustration for every word in an archive based on the odds that those words will appear together. To create a co-occurrence matrix, the co-occurrence probability is calculated using a window of words around the target term. The word embedding is then obtained by factorizing the co-occurrence matrix using a low-rank approximation.

GloVe's goal is to record word context on a global scale rather than a local one. This indicates that the method learns word embeddings by considering the full corpus of text rather than simply considering the local setting of each word. Because of this, GloVe can capture more complex word semantic associations than Word2Vec. Natural language processing activities including text categorization, retrieval of information, and machine translation frequently employ the outcome of word embedding as a feature. GloVe has been utilized in a wide range of applications because it is effective at catching the fundamental semantic and syntactic links between words. The gloVe is an effective method for learning word vector representations that capture their overall semantic and syntactic associations with other phrases in a corpus of text.

2.1.2.1.3. Hybrid methods

These methods combine count-based and prediction-based approaches to word embedding. For example, fast Text is a hybrid method that extends Word2Vec by using character-level n-grams to capture the morphology of words in addition to their context.

Fast Text: is a lightweight, open-source library that may be used to conduct text categorization tasks and learn word embeddings. It was developed by Facebook AI

Research in 2016 as an extension of the Word2Vec algorithm. Fast Text and Word2Vec differ primarily in that Fast Text also trains sub-word representation as well as word representations. Fast Text treats each word as being composed of character n-grams. Thus, the character n-gram sum represents the vector for a word. Consider the term vector "ዘገባ" is produced by multiplying the vectors of the n-grams. ዘገ, ዘገባ,ገባ. Is the outcome of adding the vectors of the n-grams. Additionally, Fast Text anticipates vectors for words that are not in its training set based on its character n-grams. Glove and Word2vec, in contrast, treat unseen words as out-of-vocabulary terms.

The fast Text library includes several pre-trained models for various languages, as well as tools for training custom models on user-defined corpora. In addition to learning word embedding, fast Text can be utilized for tasks related to text classification including topic categorization and sentiment analysis. The library uses simple neural network architecture with a softmax classifier, which can be trained using either supervised or unsupervised learning techniques. Fast Text is a useful tool for NLP researchers and practitioners who want to perform word embedding and text classification tasks quickly and with high accuracy.

2.1.3. Optimizers

An optimizer in machine learning is an algorithm that modifies a model's weights and biases during training to reduce the difference between expected and actual results.

Here are some popular optimizers used in deep learning:

Gradient descent: is a typical optimization technique used in deep learning and machine learning. During training, it is utilized to update a model's parameters to reduce a loss function (Zhang et al., 2021). By estimating the gradient of the loss function for each parameter and shifting the parameters in the reverse direction of the gradient, the primary principle behind gradient descent is to iteratively modify the weights and biases of a model. This causes the loss to decrease. The following formulation represents the gradient descent update rule: theta is equal to theta - alpha * dL/d. Theta stands for the model's weights or biases, alpha for the learning rate that regulates the update step size, L for the loss function and dL/d (theta) for the gradient of the loss for the parameters.

There are three types of gradient descent algorithms:

1. **Batch gradient descent:** This method involves training the model once on the full dataset. The weights are adjusted once every epoch, and the gradient of the loss for the parameters is calculated for the entire dataset.

2. **Stochastic gradient descent (SGD):** This method involves training the model on just one training event at a time. Following each example, the weights are revised according to the gradient of the loss for the associated parameters.
3. **Mini-batch gradient descent:** This method involves training the model on small groups of training instances. Following each batch, the weights are revised per the gradient of the loss for the batch's parameters.

Due to its efficacy and simplicity, gradient descent is an often-used optimization process. However, it has certain drawbacks, including the potential to become trapped in local minima and the requirement to carefully select the rate of learning to avoid convergence problems.

Stochastic Gradient Descent (SGD): an iterative optimization procedure used to increase the target outcome of a machine learning model or minimize the cost function. It is frequently employed in the training of deep learning models, including neural networks (Mehta & Pandya, 2020). The primary idea behind SGD is to modify the model parameters by taking gradually in the direction of the cost function's negative gradient using a randomly selected fraction of the learning data at each iteration. By not utilizing the complete training set during each iteration, the approach is more effective than traditional gradient descent.

The SGD algorithm is composed of the following steps: Initialize the model's parameters with random values. Select a random mini-batch, also referred to as a portion of the training data. Find the gradients of the cost function for the model parameters using the mini-batch. Move a tiny amount in the preferred direction of the cost function's negative gradient to adjust the model's parameters. Until convergence is obtained or for a certain amount of repetitions, steps should be repeated. SGD is helpful when working with huge datasets or models with numerous parameters since it can converge more quickly than other optimization strategies. It may, however, be hypersensitive to the mini-batch size and learning rate. To address some of SGD's drawbacks, several enhancements, including momentum and adjustable training rates, have been proposed.

Adam (Adaptive Moment Estimation): updates the parameter values of neural network models throughout training using an optimization technique. It is a well-known modification of the stochastic gradient descent (SGD) algorithm that accelerates convergence by adding momentum and adaptive learning rates (Liao et al., 2022). To adjust the training rate for each weight, Adam retains an exponentially declining average of earlier gradients and earlier squared gradients. Adam may modify the step size for

each weight based on the gradient's strength thanks to the adaptive learning rate, allowing for faster convergence and improved performance compared to traditional SGD. Adam has become a popular optimization algorithm due to its ability to handle sparse gradients and adaptive learning rates.

Adagrad (Adaptive Gradient Algorithm): is a machine learning technique that updates a model's parameters while it is being trained. Comparable to stochastic gradient descent, Adagrad modifies the acquisition rate for each parameter following the previous gradients, which can enhance convergence (Zhang et al., 2021). Adagrad's main principle is to vary the rate of learning for each parameter in a manner that is inversely proportional to the total number of squared gradients for that feature. As a result, parameters with lower gradients will have higher learning rates than those with bigger gradients, which can aid in accelerating convergence and preventing oscillations. Adagrad performs well with little information, where some input attributes could be sporadic. Adagrad can, however, build up gradients over time and reduce the successful retention rate, making it challenging to make significant parameter modifications after the training phase. Other adaptive optimization methods, including RMSprop and Adam, have been created to overcome this problem.

RMSprop (Root Mean Square Propagation): is a well-liked optimization method for gradient-based optimization of a neural network's parameters in machine learning. It is a modification of the stochastic gradient descent (SGD) technique, which modifies the network weights based on the gradient of the weights' loss function (Tian et al., 2023). The fundamental principle of RMSprop is to modify the acquisition pace for each weight parameter according to the strength of the gradients observed thus far during training. To achieve this adaptation, divide the learning rate by the average value of the squared gradients, which decays exponentially.

The learning rate is changed to better match each weight's behavior using the squared gradient, which is a measurement of how much the gradient differs for a specific weight. For non-convex optimization issues where the objective function could have numerous local minima, RMSprop is highly effective. RMSprop can move more quickly and effectively across the space of parameters and converge to a satisfactory solution by changing the rate at which it learns for each weight. RMSprop is frequently the default optimization technique for many neural network designs and has been widely employed in deep learning applications.

2.2. Related Work

2.2.1. Sentiment Analysis for foreign language

The paper titled "Evaluating pre-trained word embedding and neural network architectures for sentiment analysis in Spanish financial tweets" was presented to the audience at the Mexican International Conference on Artificial Intelligence (Arouri & Sayyafzadeh, 2022). The authors, J. A. Garcia-Diaz, O. Apolinario-Arzube, and R. Valencia-Garcia, aimed to explore the effectiveness of pre-trained word embedding and neural network architectures for sentiment analysis of Spanish financial tweets. The paper evaluated three different pre-trained word embeddings, namely Word2Vec, GloVe, and Fast Text, along with three different neural network architectures, namely Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional LSTM (Bi-LSTM).

The authors used a dataset of 4,000 tweets related to the Mexican stock market, which were annotated with sentiment labels (positive, negative, or neutral) by human annotators. They then trained and tested various models using different combinations of pre-trained word embeddings and neural network architectures, and evaluated their performance using metrics such as accuracy, precision, recall, and F1-score. The findings revealed that the Fast Text embedding performed the best among the three pre-trained embeddings, while the Bi-LSTM architecture outperformed the other two architectures. The authors also discovered that the models' effectiveness varied according to the sentiment class, with the neutral class being the most challenging to classify accurately. The paper provides useful insights into the effectiveness of pre-trained word embeddings and neural network architectures for sentiment analysis in Spanish financial tweets and highlights the importance of selecting appropriate models for different tasks and datasets.

The article titled "Deep Bidirectional LSTM Network Learning- Based Sentiment Analysis for Arabic Text" was published in the Journal of Intelligent Systems in January 2021. The authors of the article are H. Elfaik and E. H. Nfaoui (Liao et al., 2022). The article presents a sentiment analysis approach for Arabic text using a deep bidirectional LSTM network. The technique of finding and classifying the sentiment expressed in a text, such as positive, negative, or neutral, is known as sentiment analysis.

The proposed approach utilizes deep learning techniques to accurately classify Arabic text into these sentiment categories. The authors first preprocess the Arabic text by removing stop words, stemming, and tokenizing the text. They then use a deep bidirectional LSTM network to learn the features of the text and classify it into one of the

three sentiment categories. On two Arabic datasets, the suggested method is tested and found to have high sentiment classification accuracy.

The article concludes that the proposed approach can effectively perform sentiment analysis on Arabic text, this has a variety of uses, including social media monitoring and analyzing client feedback, and opinion mining. This article describes a deep learning-based approach for sentiment analysis of Roman Urdu text. The authors used data from 10,000 Roman Urdu tweets for training their model, which consisted of a convolutional neural network (CNN) and a recurrent neural network (RNN) with long short-term memory (LSTM) units (Mustapha et al., 2021). The stop words were first taken out of the text data and applied to the stem by the writers. They then converted the text into a numerical representation using a word embedding technique called Word2Vec.

The CNN was utilized to pull features out of the text, while the RNN with LSTM units was used to capture the temporal relationships between words. The accuracy, precision, recall, and F1-score were some of the common metrics used by the authors to assess the performance of their model.

The outcomes demonstrated that their strategy, which was based on deep learning, beat several other machine learning techniques frequently employed for sentiment analysis, achieving an accuracy of 81.2%. This article demonstrates the efficacy of deep learning techniques for sentiment analysis of Roman Urdu text. These techniques can be used in real-world settings like social media monitoring and customer feedback analysis. This article presents a shared Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) approach for opinion mining in Arabic language text.

The authors used a dataset of online reviews in the Arabic language for training and testing their model (Garcia-Diaz et al., 2023). First, the authors used stemming and normalization algorithms to remove stop words and punctuation from the text data. They then represented the text using Word2Vec embedding and fed it to a combination of CNN and LSTM layers to extract features and capture temporal relationships. The accuracy, precision, recall, and F1-score were some of the common metrics used by the authors to assess the performance of their model.

The findings demonstrated that their joint CNN-LSTM approach outperformed several other machine learning and deep learning algorithms commonly used for opinion mining in Arabic language text. In a comparative analysis of several feature extraction methods and model architectures, the authors demonstrated that their suggested combination CNN-LSTM model outperformed them all. Finally, the authors discussed the practical

applications of their approach in areas such as market research, client input analysis, and social media analysis. This article demonstrates the effectiveness of combining CNN and LSTM models for opinion mining in Arabic language text, which can have important applications in various domains where the Arabic language is used.

The paper "Sentiment Analysis on Bangla Text using Long Short-Term Memory (LSTM) recurrent neural network" was published in the Trends in Computing and Cognitive Engineering: Presentations of the International Conference in 2021(Ahmed & Yousuf, 2021). The paper was edited by M.S. Kaiser, A. Bandyopadhyay, M. Mahmud, and K. Ray and was published in Advances in Intelligent Systems and Computing, vol. 1309, by Springer, Singapore.

The paper describes the use of a Long Short-Term Memory (LSTM) recurrent neural network for sentiment analysis on Bangla text. The authors have used a dataset of Bangla movie reviews to train and test their model. The LSTM model has been realized using the Keras deep learning framework with TensorFlow as the backend. The authors have evaluated the performance of the LSTM model using various metrics, such as accuracy, precision, recall, and F1-score.

The experimental results show that the proposed LSTM model outperforms other state-of-the-art machine learning algorithms in terms of accuracy and F1 score. This paper contributes to the growing body of literature on sentiment analysis in non-English languages, specifically Bangla. The suggested LSTM model has the potential to be realistic to other non-English languages as well, which could have important implications for natural language processing tasks in multilingual contexts.

2.2.2. Sentiment Analysis for Amharic

A paper titled "Exploring Amharic Sentiment Analysis from Social Media Texts: Building Annotation Tools and Classification Models," authored by S. Yimam, H. Alemayehu, and U. Hamburg. The paper was published in the proceedings of the 2020 IEEE International Conference on Big Data (Big Data), and it appeared on pages 1048-1060 (Yimam et al., 2021). In this paper, the authors explore sentiment analysis in Amharic social media texts. Specifically, they develop annotation tools for creating a manually labeled dataset of Amharic social media texts and use this dataset to train and evaluate different classification models, including SVM, Random Forest, and Naive Bayes.

The authors also test several feature selection techniques and evaluate how well the models perform under various experimental circumstances. Overall, the paper presents a

comprehensive study on sentiment analysis in Amharic social media texts and gives information about the difficulties and opportunities facing this field of research. Based on the title and information provided,

The Master's thesis by Y. Getachew focuses on analyzing sentiment in the Amharic language using deep learning methods. Ethiopians speak the Semitic language of Amharic, and The method of locating and categorizing the subjective viewpoints conveyed in the text is known as sentiment analysis (Getachew, 2019). In addition to a description of the Amharic dialect and its distinctive features, the thesis most likely includes a survey of related research on emotion analysis and deep learning. This study is based on information acquired from Facebook without regard to the domain, as well as comments and replays that are categorized as favorable, negative, or neutral. 800 experimental annotated data sets are used to evaluate the model utilizing APIs (Application Program Interfaces) to retrieve annotated data.

The experiment's findings demonstrate that the approach performs with 92% precision and 82% recall for the positive class, 94% precision and 96% recall for the negative class, and 90% recall and 90% precision for the neutral class when determining opinion words. Overall, this Master's thesis may aid in the creation of Amharic-language sentiment analysis tools that might be used in a variety of contexts, including social media analysis, politics, and marketing.

The unpublished Master's thesis by S. Gebremeskels focuses on developing a sentiment-mining model for opinionated Amharic texts(To, 2010). The thesis contains a study of relevant works on sentiment evaluation and machine learning methods, as well as an overview of the Amharic language and its unique characteristics. The author may have also collected and preprocessed a dataset of Amharic texts for sentiment analysis and implemented a using machine-learning algorithm to categorize the messages' emotions. The author's model may have included feature extraction methods like word embedding or bag-of-words as well as classification algorithms like support vector machines or neural networks.

Metrics like precision, recall, and F1-score may have been used to assess the model's performance. This master's thesis could aid in the creation of Amharic sentiment analysis tools and show that low-resource languages can successfully employ machine-learning approaches for sentiment analysis.

The Master's thesis by Sultan Ayita focuses on using deep learning-based techniques for sentiment analysis of short Amharic texts (Sultan Ayita AAU, 2022). The thesis includes

a survey of pertinent articles on emotion analysis as well as deep learning, as well as an overview of the Amharic language and its unique characteristics.

The author may have also collected and preprocessed a dataset of Amharic short texts for sentiment analysis and implemented a deep learning model to classify the sentiment of these texts. In this research, the researchers proposed to use supervised, semi-supervised, and unsupervised learning that automatically extracts sentiments from Amharic text. In addition, the researchers investigate the impact of training data size on sentiment analysis for Amharic text in all three learning approaches.

To evaluate sentiment prediction on Amharic text by developing a deep learning and clustering model, the experiment was carried out with a total of 20,597 Amharic textual comments. The results reveal that supervised learning with the Bi-LSTM algorithm outperforms CNN and LSTM algorithms with 69 % accuracy; semi-supervised learning with the CNN algorithm gives 62% higher accuracy than LSTM and Bi-LSTM algorithms, and unsupervised learning with the k-means algorithm gives 50% higher accuracy than an agglomerative algorithm.

And also, the result shows the increase in training data size positively affects model accuracy for all three learning approaches in Amharic texts. The advancement of sentiment analysis techniques for the Amharic language, particularly for brief texts like social media posts, may be aided by this master's thesis. It might also show that deep learning-based methods for analyzing sentiment in low-resource languages are practicable.

The thesis work conducted by Zemenu Mekonnen focuses on sentiment analysis in the context of Amharic language using a deep learning approach (Technology, 2022), specifically at the aspect level. Sentiment analysis, also known as opinion mining, is a branch of natural language processing (NLP) that aims to extract subjective information from text, enabling companies to better understand the social sentiment surrounding their brand, product, or service.

The research conducted by Zemenu Mekonnen utilized a dataset collected from the Amhara Media Corporation's official Facebook page. The researcher applied various data preprocessing techniques such as tokenization, stop word removal, handling emojis, and punctuation, and eliminating non-Amharic texts. Additionally, the researcher converted English-language comments to Amharic and employed comment exporter software to collect a dataset consisting of 10,000 comments. To perform sentiment analysis at the aspect level, the researcher explored different deep learning models, including

Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), Hybrid CNN with Gated Recurrent Units (GRU), and hybrid CNN with LSTM. These models were trained and evaluated to determine their performance in classifying sentiment at the aspect level. After conducting experiments and evaluating the models, the researcher found that the Hybrid CNN with GRU approach yielded the best accuracy among the tested models. This hybrid model combines the strengths of CNN and GRU, which contributed to its superior performance in sentiment analysis for Amharic text.

The research conducted by Zemenu Mekonnen fills a gap in the existing literature on Amharic sentiment analysis, particularly at the aspect level, as well as utilizing a deep learning approach. This work provides valuable insights for companies, such as Amhara Media Corporation, by enabling them to quickly understand customer sentiments towards their services or products. The findings of this thesis can contribute to improving customer satisfaction and enhancing business strategies based on customer feedback analysis.

In the related works of Deep Learning for Amharic Sentiment Analysis: In the Context of Political Discourse in Ethiopia," several gaps were identified. Firstly, there was a lack of comprehensive datasets specifically tailored for sentiment analysis in the Amharic language, particularly in the context of political discourse. Existing datasets were either limited in size or focused on general sentiment analysis, failing to capture the nuances and complexities of political sentiments expressed in Amharic.

Secondly, the research on deep learning techniques for Amharic sentiment analysis was relatively scarce, with few studies exploring the application of advanced models like recurrent neural networks (RNNs) or transformer-based architectures. This limited understanding of the effectiveness and suitability of different deep learning models for Amharic sentiment analysis. Lastly, the absence of standardized evaluation metrics and benchmarks further hindered the progress in this field, making it challenging to compare and assess the performance of different models and techniques. Addressing these gaps would significantly contribute to the development of accurate and reliable sentiment analysis systems for political discourse in Ethiopia.

Chapter Three

3. Amharic Language

In this chapter, we have explored the origins, characteristics, and challenges of the Amharic language in sentiment analysis. From its roots in Ge'ez to its complex grammatical structure and unique writing system, Amharic stands as a testament to Ethiopia's linguistic and cultural diversity. In the following chapters, we will delve deeper into specific aspects of the language, including vocabulary, grammar, and its role in Ethiopian society.

3.1. Background

Amharic is an ancient Semitic language that belongs to the Afro-Asiatic language family (Linguistics, 2010). It is primarily spoken in Ethiopia, where it serves as the official language and holds significant cultural and historical importance. Amharic is also spoken by a considerable number of Ethiopians residing in neighboring countries and Ethiopian immigrant communities worldwide. The origins of Amharic can be traced back to the Aksumite Empire, which flourished from the 1st to the 7th century CE in the northern regions of present-day Ethiopia and Eritrea. The language developed and evolved, influenced by various factors such as migration, cultural interactions, and political changes within the region.

Amharic's prominence increased during the medieval period when it became the language of the Ethiopian Empire, which emerged in the 13th century and endured for several centuries (Bender, 1968). The empire expanded its territory, culture, and influence, leading to the spread of Amharic across different regions of Ethiopia. Amharic served as a unifying force, facilitating communication among diverse ethnic groups within the empire.

One of the defining features of Amharic is its writing system. It employs a unique script known as Ge'ez or Fidel, which has its roots in ancient South Arabian scripts. Ge'ez script is written from left to right and consists of a set of characters representing consonant-vowel combinations. It has been adapted to write several other languages of Ethiopia as well. Amharic has a rich literary tradition that dates back many centuries. It has been used as a medium for religious texts, historical chronicles, poetry, and other forms of creative expression. Notably, the Ethiopian Orthodox Tewahedo Church, one of the oldest Christian denominations in the world, has played a significant role in preserving and promoting the use of Amharic in religious and cultural contexts.

In recent times, Amharic has experienced further development and modernization. It has adapted to accommodate new words and concepts from fields such as technology, science, and politics. The language has also been influenced by contact with other languages, both regionally and globally.

Amharic is renowned for its unique grammatical structure and features. It is classified as an agglutinative language, where words are formed by adding prefixes, suffixes, and other morphemes to a root or stem. The language exhibits a complex system of verb conjugation and noun declension, with a variety of grammatical forms to indicate tense, aspect, mood, and gender. Today, Amharic holds a vital position in Ethiopia's cultural and linguistic landscape. It serves as a symbol of national identity, connecting Ethiopians from different regions and ethnic backgrounds. The language continues to evolve and adapt to meet the changing needs of its speakers, while also preserving its rich historical and cultural heritage.

3.2. Linguistic Characteristics

Amharic, as a language, exhibits several distinctive linguistic characteristics that contribute to its uniqueness within the Afro-Asiatic language family. Here are some notable linguistic features of Amharic (Ayalew, 2013).

3.2.1. Phonetics and Phonology

Amharic has a rich inventory of consonant and vowel sounds. It includes unique sounds such as emphatic consonants, which are pronounced with a stronger articulation. Vowel length is phonemic in Amharic, meaning that the duration of a vowel can change the meaning of a word.

3.2.2. Agglutinative Nature

The words in the agglutinative language Amharic are created by the addition of affixes, such as prefixes and suffixes, to a root or stem. These affixes carry grammatical information related to tense, aspect, mood, person, gender, and number. This agglutinative nature contributes to the complexity of Amharic grammar.

3.2.3. Complex Verb System

Amharic has a highly developed verb system. Verbs can be conjugated to indicate tense, aspect, mood, and voice. There are multiple conjugation patterns based on the different types of verbs, and each pattern has numerous forms depending on the subject and object

of the sentence. The verb system also includes a rich set of participles and verbal nouns.

3.2.4. Gender and Number Agreement

Amharic employs gender and number agreement in various parts of speech. Nouns, pronouns, adjectives, and determiners agree with the gender and number of the noun they modify. The gender system in Amharic classifies nouns into masculine, feminine, and sometimes neuter genders.

3.2.5. Verb-Subject-Object Word Order

Amharic generally follows a Verb-Subject-Object (VSO) word order. Normally, the verb comes first in a sentence, then the subject, and last the object. However, word order can be flexible, and emphasis and focus can influence the positioning of elements within a sentence.

3.3. Writing System

The Amharic language is one of the major languages spoken in Ethiopia and is primarily written using a writing system known as the Amharic script. The script is unique to the Amharic language and has its roots in the ancient Ge'ez script, which was used to write Ge'ez, an older Ethiopian Semitic language. The Amharic script is an abugida, which means that each character represents a consonant with an inherent vowel sound. However, the inherent vowel can be modified or suppressed using additional diacritic marks. The script consists of a set of 33 basic characters representing consonants, and these characters are modified to represent the different vowel sounds (Meyer, 2017). The basic characters in the Amharic script are called "Fidel" in Amharic. It has additional letters that are not present in the Ge'ez script, like (ቸ), (ኸ), and (ጨ). Each fidel represents a consonant sound, and when combined with the appropriate vowel sound, it forms a syllable.

The Fidel characters are written in a unique calligraphic style with distinct shapes and curves. Amharic is written from left to right, and the script has a numerical system. The script does not typically use spaces between words, so context and familiarity with the language are important for understanding the meaning of the text. However, modern Amharic writing sometimes includes spaces between words to aid readability, especially in printed materials. In addition to the basic fidel characters, the Amharic script includes various diacritic marks to indicate vowel sounds and other modifications. These diacritics are placed above, below, or beside the fidel characters. For example, there are

diacritics to indicate the absence of a vowel, to indicate long

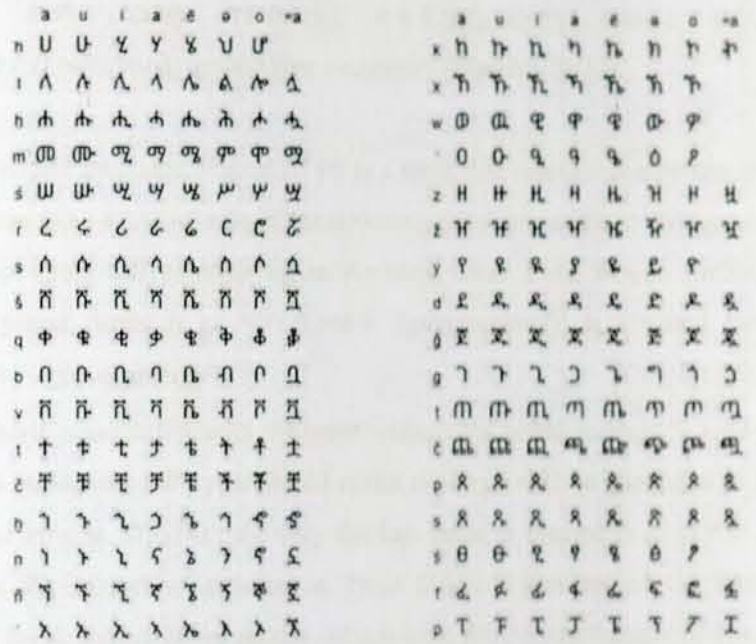


Figure 5: Amharic Language/ Ethiopian Alphabet.

Vowels, or to modify the pronunciation of a consonant. One distinctive feature of the Amharic script is the presence of a set of characters known as "ha hu Amharic." These characters are used to write foreign or borrowed words that do not have equivalent sounds in the Amharic language. They are not part of the traditional Amharic script but were added to accommodate the representation of non-Amharic vocabulary. The Amharic script has been adapted and used to write other languages spoken in Ethiopia, such as Tigrinya and Tigre. While the basic characters remain the same, slight modifications and additional characters are used to represent the specific sounds of these languages.

In recent years, the Amharic script has been digitized, and various keyboard layouts and software have been developed to facilitate typing in Amharic on computers and mobile devices. This has made it easier for Amharic speakers to communicate in their native language in the digital age. Overall, the Amharic script is a beautifully unique writing system that has played a significant role in preserving the Amharic language and Ethiopian culture. It continues to be an integral part of daily life, used in literature, education, media, and official documents throughout Ethiopia (Grover, 2009).

3.4. Word Classes

Amharic words can be divided into five categories based on where they appear in a sentence ስም (noun), ግስ (verb), ቅፅል (adjective), ተውላጠ ግስ (Adverb), and መስተዋድድ (preposition) are the five categories (Fisseha, 2020).

Noun

In the Amharic language, a noun (ስም) is a word that represents a person, place, thing, or idea. Nouns play a crucial role in constructing sentences and conveying meaning. A stem and one or more affixes make up an Amharic noun. Even though Amharic has certain unusual plural forms (e.g., 'መንግሥታት (governments)' is a plural form of a noun 'መንግስት (government)').

The Amharic noun suffix with the most widespread plural form is "-ኦች/-ዎች .If a noun ends in a consonant ኦች, you should make it plural. will be appended to the end of the noun, and vowels 'O' affect the way the last letter is formed (e.g., ሰዎች (ሰው_ኦች). It serves as the subject of a sentence. Prior linguists just copied the English language structure for Amharic, but pronouns, which were previously thought of as an independent category by linguists, have now been grouped with nouns after taking into account the distinctive nature of the language. The following are a few Amharic pronouns ይህ ,ያ ,እሱ, እስዋ, እኔ, እንተ,እንቺ ...;quantifiable specifications, such as እንድ, እንዳንድ, ጥቂት, በጣም ...; possessor-specific terms such የእኔ, የእንተ, የእሱ...,.

Verb

An action, occurrence, or state of being can be expressed by a verb. Amharic verbs, like those in other Semitic languages, are exceedingly complicated. Both the subject and the object of a phrase can be expressed with a single Amharic verb. For example, the term "ይነግረኛል" means in English "he will tell me," and it implicitly states both the subject (he) and the object (me). Verbs in Amharic are made up of a stem with ONE, TWO, or MORE affixes. Prefix, infix, suffix, and circumfix are all types of affixation. A verb's stem is made up of a root and a template.

The fundamental lexical part of the verb is represented by the roots. It is described as a series of "radicals" or consonants. Tense, aspect, mood, and one of a few derivational categories—passive-reflexive, transitive, causative, iterative, reciprocal, and causative reciprocal—are represented by slots for vowels that are put among the consonants of a root to produce a stem. Verbal stems in Amharic are made up of a root, vowels, and a template. For instance, the stem seber ('broke') is formed by sbr + ee + CVCVC. Each

lexeme has four possible tense-aspect moods (TAM).

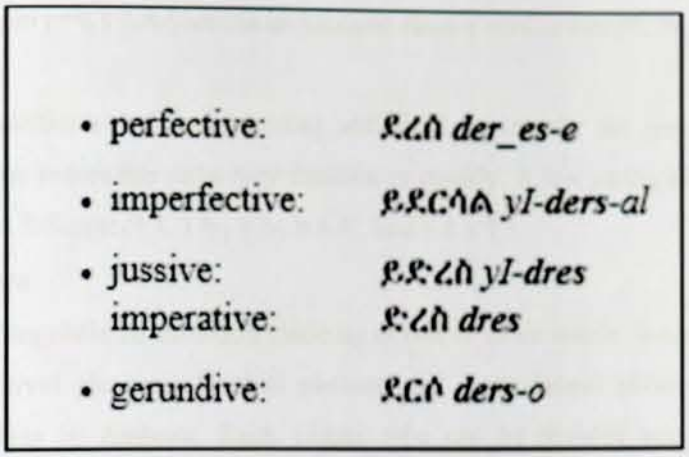


Figure 6: Amharic verb examples in four TAM types (Fisseha, 2020)

A verb in Amharic must concur with its subject. In some TAM categories, such as perfective and gerundive, suffixes are used exclusively to communicate subject-verb agreement. In other TAM categories, such as imperfective and jussive/imperative, prefixes and suffixes are combined. Described the following categories of verbs:

Perfective verbs are produced by incorporating suffixes such as (ኩ, ክ, ኽ, ኧ, ኦች, ኦቹ, ኡ) It provides the ideal verb stem with person, gender, and number indications. As an illustration, from a verb stem ሄድ, the perfective verbs ሄድኩ, ሄድክ, ሄድኽ, ሄደ, ሄደች, ሄዳቹ, and ሄዱ can be created.

Gerundive verbs are created by finishing gerundive verbs with suffixes to denote person, gender, and number. Suffixes like: (አሁ, አህ, አሽ, አ, አች, አን, አችሁ and አተዋል) be included into a verb stem. For illustration, the stem ሰር: gerundive verbs such as (ሰራሁ,ሰራህ,ሰራሽ,ሰራ,ሰራች,ሰራን,ሰራችሁ and ሰርተዋል)can be formed.

Imperfective verbs are created by adding morphemes to them, such as (ል-አ,ት, ት- አ, ይ- አ, ን - አ, ት-ኡ and ይ- ኡ) that indicates gender, person, and number. Here are some instances of verbs that are generated from steam that are imperfect. ሄድ: (ልሄድ, ትሄድ, ትሄጂ, ይሄድ, ንሄድ, ትሄዱ and ይሄዱ).

Jussive and imperative verbs are sometimes referred to as mood verbs and jussive verbs are used to convey command for the first and third person, respectively, whilst imperative verbs are used to express the second person in both the singular and plural.

Adjectives

When used to modify nouns, adjectives indicate a thing's quality and indicate how different it is from other things. It precedes a noun to describe the word's size, sort, and

action. For instance, in the phrase (ጥቁር መኪና) the word (ጥቁር) is used to qualify the color of the noun (መኪና).Adjectives in Amharic share a similar morphology to nouns.

Adverb

An adverb qualifies a verb by providing additional context for the sentence. Adverbs frequently come before the verbs they describe or modify. A few examples of adverbs in Amharic are as follows: (ትናንት, ቶሎ, ክፉኛ , and እደገኛ).

3.5. Phrases

A phrase is a linguistic construction made up of one or more words. Noun phrases, verb phrases, adjectival phrases, adverbial phrases, and prepositional phrases are the five types of phrases in Amharic. Each phrase type can be divided into "simple" and "complex" categories depending on how many different word classes are included.

Noun Phrases

A noun phrase (NP) is a syntactic construction in which a noun or pronoun serves as the headword. The phrase's head is always located at the conclusion. These expressions can be straightforward or complex. A single noun or pronoun makes up the simplest noun phrase, as in: (እሱ, እሷ, እነሱ). A complex noun phrase can consist of a noun (called the head) and other word classes such as complements, specifiers, adverbial and adjectival modifiers. These modifiers change the head from different aspects. A complex noun phrase in the Amharic language contains an embedded sentence within the phrase. For instance, (ማህዛ የገዛችው ጥቁሩ ቦርሳ) a sophisticated NP whose head is ቦርሳ . This head and the complement are united. ጥቁሩ to create the short noun phrase (ጥቁሩ ቦርሳ). In turn, the dependent clause has been combined with this noun phrase (ማህዛ የገዛችው) to make the complex NP described above. If there is a "የ" before the verb, it means that the clause is subordinate and cannot stand alone.

Verb Phrases

A verb phrase (VP) is made up of a verb acting as the phrase's head and other words like complements, modifiers, and specifiers. As with noun phrases, verb phrasal expressions can be simple or complicated. An embedded sentence that functions as a complement or a modifier can be found inside a complex verb phrase. Consider the following example, (በ መኪና ወደ ትምህርት ቤት ሄደች). The adverb በ መኪና and prepositional phrase (ወደ ትምህርት ቤት) have modified the verb (ሄደች).

Adjectival Phrases

An Amharic adjectival phrase is constructed similarly to a noun phrase and a verb

phrase. It is made up of a head (adjective), as well as complements, modifiers, and specifiers. For example, (በጣም ትልቅ-ኑ, "-ኑ" is a specifier, (በጣም) is a modifier that modifies the adjective's head (ትልቁ).

Prepositional Phrase

Prepositions serve as the "head" of prepositional phrases, which also include nouns, noun phrases, verbs, verb phrases, etc. Prepositional phrases, unlike other forms of phrases, always start with the head of the sentence. As an illustration, the prepositional phrase (ከተማሪዎች ጋር ወደ ትምህርት ቤት) for instance, ከ ጋር and ወደ are prepositions which are combined with the nouns ተማሪዎች and ትምህርት ቤት, respectively to form their prepositional phrase.

Adverbial Phrases

A phrase with an adverb as its headword is known as an adverbial phrase. It can be created from a single adverb or several. Adverbs describe the locations, occasions, or particulars of the activity mentioned by the verb. For example, (በፍጥነት መጣች), በፍጥነት is the only adverb that expresses how strong the verb is.

3.6. Clauses

A clause is a collection of words. At least one verb phrase must be present in a clause. Independent clauses, or main clauses, and dependent clauses, or subordinate clauses, are the two different sorts of clauses.

Independent Clause (Main clause)

A clause that can stand alone and conveys a clear, complete idea is an independent clause. It is a short sentence with a verb and a subject. For example, the clause: (ይሃንስ ኳስ ገዛ), has a subject, ይሃንስ, a verb, ገዛ, and an object, ኳስ. The next part goes into more detail about the Amharic simple sentence kinds and structure.

A dependent clause (subordinate clause)

A clause that depends on another clause to form a full sentence is referred to as a dependent (subordinate) clause. Although they have a subject and a verb, subordinate clauses do not fully communicate a thought. Subordinate conjunctions in English are used to identify subordinate clauses. Following, although, as, as if, because, before, even if, even though, if, to, since, though, unless, till, whatever, whether, when, whenever, and while are some examples of subordinate conjunctions in English. These kinds of clauses function in sentences as an adverb. Similar to an adverb, they give information to the main phrase that elaborates on the when, where, why, how, how much, or under what

circumstances the action in the sentence occurs. In contrast to English, Amharic subordinate clauses are distinguished by verbal suffixes. For instance, the verb (ሰጠ) May be given new forms by the addition of affixes (i.e., ስለ-ሰጠ, እየ -ሰጠ, ከሰጠ, ቢ-ሰጠ-እ ም, etc.). A sort of subordinate clause known as a related clause serves as an adjective and describes a noun. When speaking English, relative sentences use pronouns like who, whom, whose, that, and which. On the other hand, the morpheme (የ) that is joined to a verb can be used to recognize Amharic relative clauses.

Examples of Amharic relative clauses:

- ትላንት የ -መጣው ሰው
- አበበ የ -አገኘውን ሰው
- ሰላም የ -ገዛችው ጫማ

3.7. Sentences

A sentence is made up of one or more clauses that communicate something important. Amharic follows a subject-object-verb (SOV) grammatical pattern, unlike English, which has a subject-verb-object word order. For instance, (ማህዛ ወደ ትምህርት ቤት ሄደች). The subject and verb are the two main components of Amharic phrases. The sentence's subject, which is a noun phrase, is always put before the verb. It could be classified as simple, compound, or complex. A verb is always added after an Amharic sentence. Based on their structure, Amharic phrases can be divided into four groups. Sentences: simple, compound, complex, and complex-compound.

Simple Sentence

A sentence is considered simple if only one independent clause is present. An independent clause is a set of words that represents a complete notion and comprises both a subject and a verb. It is composed of a straightforward noun phrase, then a straightforward verb phrase with just one verb. For example, (ማህዛ መጽሃፍ ገዛች) is an independent clause. It contains a subject (ማርታ), an object (መጽሃፍ), and a verb (ገዛች), and it expresses a complete thought. Simple sentences come in four different flavors: exclamatory, interrogative, declarative, and imperative.

Declarative sentences are employed to convey knowledge. Declarative sentences in Amharic always conclude with the punctuation mark "::" which is a period (.) in English.

Example: (ዘመነ በንክ ካፒታሉን ወደ አስር ቢሊዮን ብር አሳደ።)

Interrogative sentences are sentences that ask a question. In Amharic, these types of sentences always end with a question mark (i.e. "?"). Example: (ዘመነ በንክ ካፒታሉን

ወደ ስንት እሳደገ ?)

Complex Sentence

A complex statement is created by combining an independent clause with one or more subordinate clauses. Multiple verb phrases can be found in complex sentences. For example, the sentence: (ወደ ውጭ የ ሚላክ ው የ ሰሊጥ ምርት በከፍተኛ ሁኔታ እየ ቀነሰ መምጣቱ ተገለጸ ።) is made up of dependent clauses that cannot stand alone. There is a subordinate clause in the noun phrase of the main sentence (i.e., "ወደ ውጭ የ ሚላ ክ ው የ ሰ ሊጥ ምር ት (የ ሰ ሊጥ ምር ት) and a verb phrase (i.e., ወደ ውጭ የ ሚላ ክ ው). There is a subordinate clause in the verb phrase of the main sentence (i.e., በ ከ ፍ ተ ኛ ሁ ኔ ታ እ የ ቀ ነ ሰ መ ም ጣ ቱ). It is formed from an adverbial phrase (i.e., በ ከ ፍ ተ ኛ ሁ ኔ ታ) and a simple verb phrase (እ የ ቀ ነ ሰ መ ም ጣ ቱ).

Compound Sentence

A sentence that has at least two independent clauses with connected ideas is said to be a compound sentence. Typically, a coordinator (also known as a coordinate conjunction) joins the clauses. E.g., (እ ና , ወይ ም, ግ ን), and so coordinate clauses are those between which a coordinate conjunction appears. A sentence's coordinating clause shares the same significance as the main clause. Despite sharing the subject, the object, or the verb of the main phrase, some coordinate clauses are entire sentences that can stand on their own. **Examples**:

- አ ል ማዝ ከ ጅ ማ እ የ መጣች ነ ው ነ ገ ር ግ ን እ በ በ እ ቤ ት ው ስ ጥ የ ለ ም።
- አ በ በ ወደ ጅ ማ ሊ ሄ ድ ይ ች ላ ል ወይ ም እ ል ማዝ ከ ሃ ረ ር ል ት መ ጣ ት ች ላ ል ች ።
- ማሪ ቱ ከ ት ም ህ ር ት ቤ ት መ ጣች እ ና ተ ኛ ች ።

Compound-Complex Sentences

Compound sentences and complicated sentences are combined to create compound-complex sentences. At least two independent clauses and one subordinate clause are present. In a compound-complex sentence, the independent clauses that are coordinated by a coordinating conjunction are known as coordinate clauses. The independent clauses in this compound-complex statement are bolded as an example. (የ ጠዋት ጸ ሃ ይ ስ ት ወጣ የ እ ግ ር ጉ ዞ ማድ ረ ግ እ ወዳ ለ ው ነ ገ ር ግ ን ዛ ሬ ዝ ና ብ ስ ለ ነ በ ረ እ ስከ ም ሳ ሰ ሃ ት ድ ረ ስ ከ ቤ ት አ ል ወጣ ውም). This sentence has two independent clauses and two dependent clauses. The dependent clause (የ ጠዋት ጸ ሃ ይ ስ ት ወጣ) and (ዛ ሬ ዝ ና ብ ስ ለ ነ በ ረ) cannot stand on their own as a complete sentence; they are dependent. However, the independent clauses (የ እ ግ ር ጉ ዞ ማድ ረ ግ እ ወዳ ለ ው) and (እ ስከ ም ሳ ሰ ሃ ት ድ ረ ስ ከ ቤ ት

አ ል ወ ጣ ውም) can be complete sentences on their own. The independent clauses are joined by coordinate conjunction (ነ ገ ር ግ ን).

3.8. Challenges of Amharic Sentimental Analysis

The majority of sentiment analysis research is done in languages like English and Arabic, which have a wealth of resources. The results of this research, however, cannot be directly transferred to the Amharic language. The Amharic language differs from the other languages being studied in that it has unique traits. This aspect which may pose a challenge to the direct transfer of the described methodologies to other languages is covered in greater detail below.

Morphological Complexity: Amharic has a complex morphology which makes it challenging to tokenize and lemmatize the language. This complexity can lead to errors in sentiment analysis. For instance, a Root ት ል ቅ Basic stems ታላ ቅ may have several variants ታላ ቁ , የ ታላ ቁ , ከ ታላ ቁ , ታላ ላ ቆ ቹ , ታላ ቅ . Multiple synonyms and polysemous words exist Words with the same meaning but a different wording is called synonyms. For instance, the word "አ ስ ተ ማሪ " which means "teacher" can be called differently, such as "መ ም ህ ር ". Words that have multiple meanings can be identified as polysemous by emphasizing specific letters as you read. For instance, the word "ዋ ና" which means "swimming "can have another meaning of "Main" when "ና " is stressed.

Character Redundancy: in the Amharic writing system, some letters or symbols have the same sound when read, furthermore to several methods of saying the same thing and consonants that sound the same. For instance, the term ሠራዊት "Army" is translatable into Amharic as ሰ ራ ዊ ት, and people interchangeably use letters like ሠ, ሰ , ሀ , ሐ or ኀ and ጸ or ፀ to write the same word.

Chapter Four

4. Data Collection and Analysis

An overview of the data gathering and sample techniques used to annotate the dataset is provided in this section. We will go over the difficulties of annotating language data with limited resources as well as the shortcomings of the current annotation tools. We suggest a novel annotation tool made especially for low-resource linguistic data collecting to overcome these problems. This section seeks to aid in the creation of successful methods for low-resource language data annotation by offering insights into our data gathering and annotation procedures.

4.1. Data Acquisition and Dataset Characteristics

Since the data were not earlier gathered and released for a specified purpose, this study used primary data collection. The shifts in our political discussion and involvement on social media and the trends of replies with post-it belong are discovered via careful analysis of these data. To follow sentiment analysis, the study employs a binary categorization. Therefore, sentiment analysis includes two distinct classifications: the binary classification, where thoughts fall into one of two categories (for instance, positive or negative); the multiclass classification, where opinions fall into one of several categories (for instance, strongly agree, agree, fair, disagree, and strongly disagree); and the subjectivity classification, where critiques or sentences are examined. Various methods have been employed in this field of research in terms of the instruments and procedures used to locate and collect data.

The study's initial objective was to identify the data's source and gather the data using various already-developed and ready-made methods, such as the Facebook API, Python scripts, and comment-exporting websites. Employing the ready-made Facebook scraper tool, we can quickly extract posts and comments from any public Facebook Page or Group and export them to Excel spreadsheets. The first thing we had to do to view the public page material was sign up for a Facebook account. Finally, using Python code, we downloaded Facebook page comments and posts to Excel.

The first part of the information was acquired from both private and public news websites that used social media to reach their followers and audiences who spoke Ethiopian. Religion, sports, entertainment, and other unrelated section of media categories are excluded from our study. We chose Fana broadcasting corporate (FBC) as

the governmental or public TV and radio that uses social media as an alternative tool since nearly all federal and regional government media try to release the same kind of news, especially political news, and the news language that is broadcast is Amharic and other we choose Ethiopian Prime Minister Dr. Abie Ahmed Ali from a privately held Facebook page because, they have a greater number of followers than the other members and employ Amharic as a means of transmission (10 Top Ethiopia Facebook Pages 2023 — AllaboutETHIO, n.d.).

Table 1: Data collection source

Media	people follow this page	people like this page	URL	verified badge	Categories
OfficialFANA Broadcasting Corporate (FBC) Facebook page	3,445,573	2,736,793	https://www.facebook.com/fanabroadcasting/	verified	Media/news company
Official Abiy Ahmed Ali Facebook page	4.2M		https://www.facebook.com/PMAbiyAhmedAli/	verified	Politician

4.1.1. Sample Data Size

In the context of our research, data was gathered during the current political transition period in Ethiopia, which lasted from June 1, 2018, to March 20, 2023. The study chose this period because, first, there was a significant political movement in the nation and, second, many people used social media to openly express their opinions without being frustrated by the executive branch or the dominant party. The information about domain-based systems was carefully gathered, manually, and by selecting political coverage that was extensively covered by all major national news outlets. The study collected data from roughly 500 posts from the selected large-scale public and private social media news, each of which had 500 or more comments. However, throughout the data collection period, the study rejected non-political postings and posts with fewer views from those data sources.

4.1.2. Annotation Strategy

One of the time-consuming and challenging steps in creating machine learning components is annotation. In countries where technology is underutilized, people often conduct surveys or annotations using human labor, for example, by responding to printed questionnaires. This necessitates the conversion of the questionnaire into an electronic form for the machine learning strategy, which is time-consuming and may potentially result in encoding errors.

Crowdsourcing systems are required to annotate a huge number of datasets with the least amount of pay. However, most crowdsourcing sites have a difficult registration process, especially if you want to include worker payment transactions. For instance, Amazon Mechanical Turk (MTurk) does not allow job requesters from Ethiopia to register. We are unable to use MTurk's platform for Amharic sentiment analysis because there aren't enough Amharic speakers there and there aren't enough online payment methods available in Ethiopia. We, therefore, used a spreadsheet application where we distributed our data in a tabular style as our first feasible method for labeling our dataset (see Figure 7).

In this research experiment, the data is meticulously annotated by a dedicated individual, ensuring a high level of accuracy and consistency throughout the annotation process. The annotator carefully examines each data point and assigns relevant labels or tags based on predefined criteria. To ensure quality control, a second person thoroughly checks the annotations, verifying their correctness and addressing any potential errors or ambiguities. While inter-annotation agreement measures are not tested in this particular case due to the involvement of a single annotator, the diligent approach and meticulous checking undertaken by both individuals contribute to the successful annotation of the data, bolstering its reliability for subsequent analysis and interpretation. We discovered that utilizing a spreadsheet to annotate the data is time-consuming, error-prone, and challenging to use.

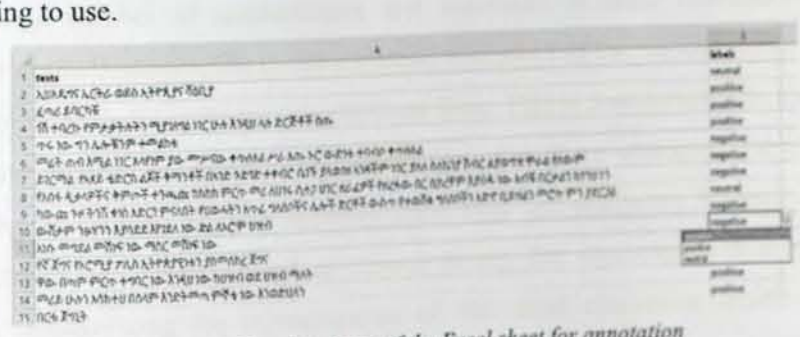


Figure 7: User interfaces of the Excel sheet for annotation

4.2. Pre-processing and preparation of data

Data pre-processing and preparation of Amharic sentences follow the same principles as in other languages. However, some language-specific considerations need to be taken into account. Here are some steps that can be taken for pre-processing and preparing Amharic sentences for analysis:

Tokenization: Splitting the text into words or tokens. In Amharic, words are separated by spaces, so tokenization can be performed by simply splitting the sentence at each space. In this work, Tokenizers are created and used by utilizing pre-trained tokenizers using Kera's, a Python deep computing toolkit. Keras is a popular deep-learning library in Python that provides a high-level interface for building and training deep neural networks. Keras also includes several utilities for text processing and natural language processing (NLP), including tokenization. A text sequence is tokenized when it is divided up into smaller pieces, or tokens.

These tokens in NLP are frequently words or sub-words, but they can also be characters or other units, depending on the application. Keras provides a Tokenizer class that can be used to tokenize text data from a Utilizing the fit_on_texts method, the tokenizer object is fitted to the text data. As a result, an alphabetical list of all the different phrases in the written information is created, and each word is given a distinct integer index. The text data can then be transformed into sequences of integer tokens using the texts_to_sequences method. One of the entered texts is represented by each inner list of the output, which is a collection of lists. Many NLP tasks, including text categorization, language modeling, and machine translation, depend on tokenization. As part of a broader deep-learning workflow, Keras offers an easy and adaptable method to tokenize text input.

Normalizing character level: in Amharic words can be challenging due to the complex nature of the language. Amharic has a unique script that consists of 33 basic characters and a large number of combinations and variations of those characters. Amharic character normalization refers to the reduction procedure of the various forms of a letter or symbol in Amharic script to a standardized form. This is done to simplify the data and reduce the number of different representations of a given character, facilitating the processing of data by machine learning algorithms and analyzing Amharic text data. Normalization can be performed at various levels, but character-level normalization involves standardizing the representation of individual characters, which can have

different forms based on where they are in a word or the characters they are adjacent to. By normalizing Amharic characters at the character level, machine learning models can more accurately recognize and interpret Amharic text, increasing the precision of many NLP tasks including sentiment analysis and text classification, and machine translation.

Normalizing words with Labialized Amharic characters: Labialized Amharic characters are those that have a labialization feature, such as the addition of a "ዋ" sound at the end of a consonant. Normalizing words with Labialized Amharic characters can be a challenging task for machine learning (ML) normalization because these characters can change the meaning of a word. Therefore, it is important to properly handle them during normalization to ensure accuracy in text processing.

One approach is to convert the labialized character to a non-labialized form, which can be achieved through various methods such as mapping them to their respective non-labialized counterparts or using a phonetic transcription system such as ቡል+ዋል or ቡል+ኦል to ቡል±ል. This normalization process can improve the performance of ML models in a variety of applications, including recognition of speech and processing of natural languages, where accurate transcription and understanding of Amharic text are crucial.

Data Cleaning: Removing any unwanted characters, punctuation, or special symbols from the text.

In a comprehensive research project, a total of 300 posts were carefully analyzed, resulting in the collection of an impressive dataset of 150,000 valuable data points. This extensive collection process involved meticulous extraction and organization of relevant information from the posts. Furthermore, the dataset was subjected to a thorough cleaning process, ensuring that the comments were refined and refined to their optimal quality. To enhance the research findings, an additional step was taken to enhance the dataset's depth and accuracy. A remarkable 45,000 comments were diligently annotated, a task that required an immense amount of effort and attention to detail. The annotation process involved categorizing and labeling the comments based on specific criteria.

This significant volume of annotated comments serves as a crucial resource for the research, facilitating the exploration of various trends, patterns, and insights. The careful collection and annotation of this extensive dataset contribute to the robustness and reliability of the research findings, ensuring a comprehensive analysis and allowing for well-informed conclusions for this Research use a total of 24948 positive and 20728 Negative.

Chapter Five

5. Proposed Approach and Architecture

The researched deep learning framework for determining the polarity of Amharic text utilizing the Simple RNN, CNN, LSTM, GRU, and Bi-LSTM deep learning models are laid out in the proposed architecture. Data collection, data pre-processing and preparation, feature extraction, and sentiment categorization make up the majority of its components. The next section talks about each component's specifics.

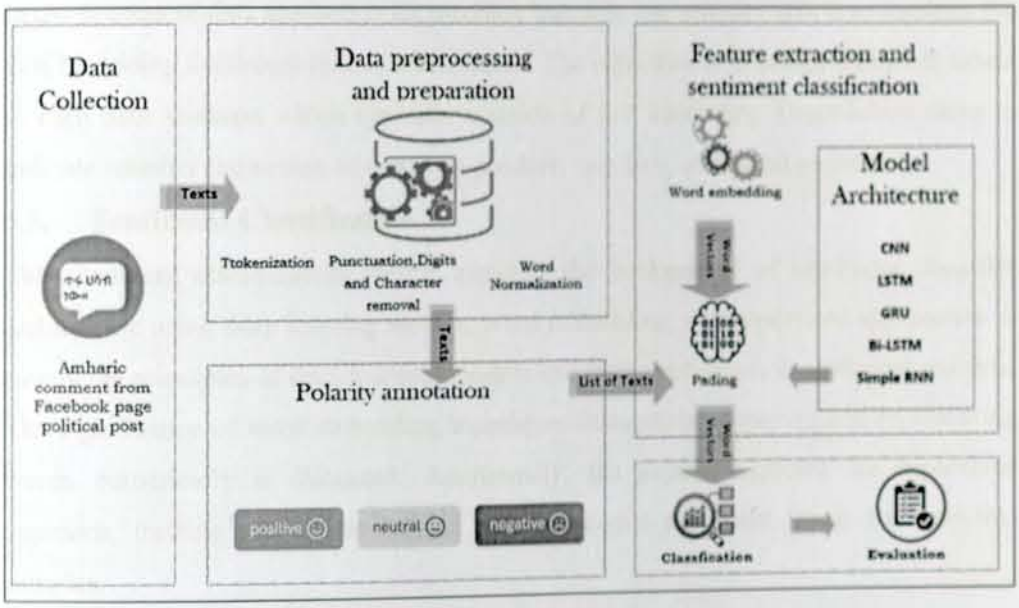


Figure 8: The Architecture of the Proposed Approach

5.1. Data preprocessing and preparation

In the data pre-processing phase of Amharic text for sentiment analysis, several steps are involved to ensure the appropriate format. These steps include punctuation removal, Latin letter removal, stop word removal, digits removal, tokenization, and normalization. These linguistic operations are implemented to eliminate word ambiguity and improve the accuracy and effectiveness of the researcher's analysis.

Tokenization is the process of dividing the text into individual tokens or words, typically separated by spaces. This allows for further processing without necessarily understanding the meaning or connections between the words. In this work, Keras, a Python deep-learning library, is utilized to create a tokenizer.

Punctuation marks such as {፤, ፣, #, -. ;:'...} are removed from the text as they do not contribute to the researcher's approach and analysis.

Latin characters and digits that appear in the documents but do not hold any meaning or relevance to the research are also eliminated.

Word normalization is performed tc reduce the various forms of a letter within a word to a single form. For example, variations like ሀ , ሐ , ሃ , and ኀ are normalized to ሀ , while ሰ and ሠ are normalized to ሰ , and አ , ዓ , ኣ , and ዐ are normalized to አ . This process helps in achieving consistency and reducing redundancy in the text.

5.2. Data Annotation

In the process of data annotation for machine learning, our primary task is to annotate the data by adding sentiment-related information. The objective is to assign sentiment labels to each data instance, which typically consists of text sentences. These labels serve to indicate whether the sentences convey a positive, negative, or neutral sentiment.

5.3. Sentiment Classification

The sentiment classification section explores the background of sentiment classifier architecture using deep learning models, word embedding, and supervised approaches. It covers the principles of deep learning models and their application in sentiment analysis. The significance of word embedding techniques in capturing meaning and representing words numerically is discussed. Additionally, the section explores the supervised approach, training models on labeled data to predict sentiment labels for new text samples.

5.3.1. Word embedding

A method used in natural language processing (NLP) called word embedding enables words to be expressed as vectors of actual integers. It is a means to mathematically record the relationships between phrases in a high-dimensional space where the meaning of words can be represented (Ortis et al., 2020). Word embedding seeks to associate semantically relevant words with vectors that are within range of one another in the embedding space. It does this by mapping words in a vocabulary to high-dimensional vectors. With the use of processes like vector addition and vector subtraction, we may modify the vectors and identify correlations between words. Word embeddings can be created using a variety of techniques, such as prediction-based techniques like Word2Vec, GloVe, and FastText, as well as count-based techniques like Term Frequency-Inverse Document Frequency (TF-IDF) and Latent Semantic Analysis (LSA).

In the instance of the continuous bag-of-words model or the skip-gram model, Word2Vec, a well-known prediction-based method for building word embeddings, uses a neural network to predict the likelihood of a word given its context or the context given a word.

5.3.2. Padding

Padding is a data processing technique used in machine learning, especially, to add more data to the input to make it the right size or shape. Padding is commonly used when the data needs to be transformed or prepared before it can be used in a particular model or algorithm. Padding is frequently applied to sequences of data in machine learning, such as sentences of different lengths in natural language or audio samples. To make shorter sequences easier to process with some algorithms, padding is the process of adding zeros or other filler data to the end of shorter sequences to make all sequences the same length.

5.3.3. Supervised Approach

When using a supervised technique, a framework gets trained on labeled data, meaning every input data point has an associated output label. The goal of supervised learning is to develop a function for mapping that can predict the labels of the output for novel, unknown inputs. The algorithm has to be fed a set of labeled samples throughout the training phase to discover the fundamental trends and correlations between the input data and the output labels. The framework can be utilized for predicting results for fresh data sets after being trained.

5.3.4. Deep learning models

To acquire knowledge and extract features from complex data, deep learning models are a subtype of artificial neural networks (ANN) made up of numerous layers of interconnected nodes. These models are capable of carrying out a variety of tasks, such as speech detection, natural language processing, image recognition, and more. Deep learning models' capacity to learn from huge datasets without manually designing features is one of their key advantages. Instead, the model is capable of recognizing relevant characteristics from the data, which makes them especially helpful in situations when the fundamental trends or structures are unclear.

5.3.4.1. Recurrent Neural Network (RNN)

A particular class of neural network is created to process sequential data, including time-series data, audio, natural language text, and other data with a temporal component. The foundation of RNNs is the idea of using an identical set of weights over all time steps. As a result, they can analyze inputs of varying duration and keep track of prior inputs,

which helps them predict future outputs. An RNN's fundamental design has a looping link that enables data to be transmitted from one step to the next. The ability of RNNs to cope with sequential data with a dynamic length and orientation is their main advantage over standard neural networks. This qualifies them for a variety of applications, including sentiment analysis, image captioning, machine translation, and speech recognition. RNNs come in a variety of forms, including the fundamental RNN, LSTM networks, and gated recurrent units (GRUs). The more complex RNN variants such as LSTM and bi-LSTM are better at modeling dependency over time while preventing the vanishing gradient problem.

5.3.4.2. Basic RNN Model

A particular kind of neural network intended for sequential data processing, where the current result is based on the prior input. Each input x_t to an RNN is fed into a hidden state h_t together with the hidden state h_ t-1 that came before it. The output y_t is then obtained by passing the hidden state h_t through a non-linear activation function, where h_t = f_W (x_t, h_ t-1), and y_t = g_W(h_t), respectively, where f_W and g_W are non-linear functions quantified by the weight matrix W. The fundamental concept of RNNs is that they can keep a "memory" of previous inputs by using the hidden state h_t. This makes the model suitable for jobs like linguistic modeling, recognizing speech, and machine translation since it enables the model to generate recommendations according to the context of prior inputs. Typically, during training, a loss function like cross-entropy or mean squared error is used to train the model to reduce the variance between its anticipated result y_t and the true output y_t*. Backpropagation through time, which updates the parameters W using an optimization algorithm like stochastic gradient descent (SGD), computes gradients for the parameters W at each time step. The problem of vanishing gradients is addressed by a variety of RNN modifications, including Long Short-Term Memory and Gated Recurrent Unit networks, which enhance the model's performance on long-term dependencies.

5.3.4.3. Gated Recurrent Unit (GRU)

GRUs are made to handle sequential data, just like LSTMs, and are especially useful for applications that call for capturing long-term dependencies (Mu, 2022). GRUs and LSTMs differ primarily in that GRUs have a more straightforward architecture and fewer parameters. Similar to LSTMs, GRUs use gating techniques to regulate the exchange of information across the network, but they only have two gates as opposed to three. The reset gate and the update gate are the names of the two gates in a GRU. The update gate

in a GRU defines what percentage of the latest hidden state should be based on what is currently input and how much should be based on the previous hidden state. The reset gate in a GRU determines how much of the previous hidden state should be forgotten. GRUs are particularly suited for tasks that involve modeling long-term dependencies because they can arbitrarily modify and forget data over time utilizing these gating processes.

5.3.4.4. Long Short-Term Memory (LSTM) Model

An RNN architecture design was created primarily to overcome the vanishing gradient issue with conventional RNNs. The difficulty in the model's ability to learn dependencies that last forever is due to the vanishing gradient problem, which happens when gradients get too small to be effective during backpropagation across time (Bergström & Hjelm, 2019). The state that is hidden h_t and the cell state c_t in an LSTM model are calculated using gated units that regulate the information flow. The entry gate, forget gate, and output gate are the names of these components.

5.3.4.5. Bidirectional LSTM (Long Short-Term Memory) model

a specific variety of recurrent neural networks that can process sequential data both forward and backward. It's comprised of two LSTMs, one of which moves the input sequence forward and the other of which moves it backward (Arora & Bansal, 2022). The input sequence is given to the initial LSTM layer at the forward pass, which creates a series of hidden states that store the previously learned information. The second LSTM layer receives the input sequence during the backward pass in reverse order, creating a series of hidden states that record the data from the future. The final output sequence is created by concatenating the results from both LSTMs. When anticipating the subsequent word in a phrase or the next element in a time series, the key benefit of utilizing a bidirectional LSTM is that it can take into account both past and future context.

5.3.4.6. Convolutional Neural Network (CNN) Model

a specific class of deep neural network model that's mainly utilized for image and object identification tasks. By convolving many filters over the input and employing pooling procedures to reduce the size of the resulting feature maps, it is intended to analyze information with a grid-like structure, such as photographs or audio spectrograms. Convolutional layers, pooling layers, and entirely connected layers make up the foundation of a CNN. Convolutional layers produce a set of feature maps by applying a collection of filters to the input. Pooling layers reduces the spatial size of the maps by downsampling the feature maps. The feature maps are processed by fully connected

layers to produce a final prediction or classification. Several convolutional layers are followed by pooling layers, which are followed by a few fully linked layers that make up a standard CNN design. A softmax layer, which produces possibilities for every category in a classification job, is typically the last layer of a CNN. By adjusting the weights of the filters and fully connected layers while learning to reduce a loss function, including cross-entropy loss, CNNs can be trained via backpropagation (Upreti, 2022).

Chapter Six

6. Experiment and Performance Evaluation

6.1. Introduction

In this chapter, we present the results obtained from the experiment conducted based on the concepts discussed in the previous chapters. This experiment aims to identify the best polarity classification solution among five deep learning algorithms, namely CNN, simple RNN, LSTM, Bi-LSTM, and GRU. The researcher used supervised learning techniques to evaluate their effectiveness in identifying sentiments. The test data size used in the experiment was 0.2, and the train-test split was implemented using the Scikit-learn Python machine learning library via the train_test_split () function.

To guarantee the reliability and correctness of the findings, the researcher used various types of architecture, parameters, data size, optimizers, and pre-trained models to protect the models from underfitting and overfitting. The purpose of my thesis experiment was to explore and transform raw data using the Python libraries Pandas and Numpy. These libraries were essential for cleaning and processing the data to prepare it for analysis. To aid in the visualization of the data, I utilized the Matplotlib, Seaborn, and Plotly libraries. These libraries allowed me to create insightful graphs, charts, and interactive plots that helped to illustrate key trends and patterns within the data. To build and train accurate predictive models, I employed a range of AI and machine learning techniques using the Scikit-learn, TensorFlow, Keras, PyTorch, and XGBoost libraries.

These libraries provided a powerful set of tools for model selection, feature engineering, and evaluation. In general, my use of these libraries enabled me to conduct a thorough and comprehensive analysis of the data, yielding valuable insights and predictive models that could be used for future research and real-world applications. The findings of this experiment will be useful for sentiment analysis applications including social media monitoring and client feedback analysis, and product review analysis. The results obtained from this experiment will help researchers and practitioners in selecting the best algorithm to analyze sentiment and increase sentiment analysis systems' accuracy.

6.2. Dataset

We scraped data from social media headlines and comments and labeled them as positive, negative, and Neutral, which consists of a total of 45,000 rows. We preprocessed the data by, removing all stop words, filtering out emojis, numbers, and

symbols, normalizing characters, tokenizing each word, converting them to sequences, and padding the sequences to a maximum length of 10.

6.3. Evaluation Metrics

In our deep learning sentiment analysis model experiment, we used a variety of evaluation indicators to assess the model's performance, including Accuracy, Precision, Recall, F1 score, and ROC AUC. There are several reasons why we might choose to use multiple evaluation metrics.

Accuracy: is a widely used evaluation metric for text sentiment prediction models. It calculates the percentage of all predictions that the model made that was accurate. Accuracy, then, is a measure of how frequently a model can anticipate a text's emotional content. Even though accuracy is a useful metric, it might be deceptive in some situations where there is a class imbalance. Class imbalance occurs when one sentiment is much more frequent in the data than the others. In such cases, the model may be biased towards the more common sentiment, resulting in high accuracy for that particular sentiment but poor performance for the other sentiments. This could result in an erroneously optimistic assessment of the model's efficiency. Other evaluation metrics, like precision, recall, and F1-score, can be utilized to get around this problem. These measurements are more resistant to class imbalance since they account for both true positives and false positives. Because of this, it's crucial to choose evaluation criteria carefully based on the particulars of the task at hand.

Precision: is a machine learning assessment metric used to gauge how well a model performs at foreseeing positive occurrences. In the realm of emotion analysis, positive instances might refer to texts that do so. Precision is defined as the proportion of the model's positive predictions that are accurate. In other words, precision quantifies the model's accuracy in forecasting positive instances, or the percentage of positive instances correctly predicted out of all positive instances anticipated. Precision is especially helpful when reducing false positives is crucial.

A false positive result in a medical diagnosis, for instance, could prompt needless medication or surgery that is harmful to the patient. To reduce the possibility of false positives in such circumstances, a model with great precision is essential. As a measure of the effectiveness of the model in forecasting positive instances, precision is a valuable measure of assessment for sentiment analysis models. It is particularly relevant in circumstances when limiting false positives is important.

Recall: This is a crucial evaluation parameter used to judge how well deep-learning sentiment models perform. It calculates the percentage of real positive samples in the dataset that the model correctly predicted as true positives. Recall, then, assesses the model's accuracy in properly identifying every positive sample in the dataset. The recall is very helpful when minimizing false negatives, which are instances where the model predicts an inappropriately adverse outcome for a positive sample. For instance, false negatives in illness detection can be extremely dangerous because they can result in a delayed or missed diagnosis. Similarly to this, false negatives in spam filtering might cause critical emails to be ignored or filtered into the spam folder.

Recall optimization allows the model to better recognize all positive samples, this can assist to lessen false negatives and enhance the model's overall performance. The percentage of true positive forecasts and generally favorable predictions provided by the model is measured by precision, which is an important factor to consider while maximizing for recall. Therefore, depending on the particular use case and model objectives, it is crucial to achieve a compromise between recall and precision.

F1 score: The F1 score is a popular evaluation statistic in deep machine learning sentiment analysis. It is a performance metric for the model that accounts for both recall and precision, giving a balanced picture of how effectively the model performs on both positive and negative classes. Out of all the cases the model has classified as positive, precision measures how frequently it does it accurately. Contrarily, recall is a measurement of how frequently the model properly recognizes all positive cases. The harmonic mean of recall and precision is the F1 score.

The F1 score will be low if either precision or recall is poor since the harmonic mean gives more weight to the lesser of the two figures. Since it offers a balanced assessment of the model's efficacy on both positive and negative classes, it is a helpful metric when the classes are unbalanced. When there is a trade-off between recall and precision, the F1 score is a useful metric. For instance, precision may be more important in sentiment analysis if we wish to avoid mislabeling negative sentiment as positive. As an alternative, if we want to ensure that we don't overlook any instances of unfavorable sentiment, we could wish to prioritize recollection. We can determine the ideal ratio between these two parameters using the F1 score.

ROC AUC: ROC AUC (Receiver Operating Characteristic - Area Under the Curve) is a frequently used evaluation statistic in deep learning sentiment analysis. By comparing the true positive rate (TPR) and false positive rate (FPR) for various threshold values, it

assesses the model's capacity to distinguish between positive and negative classes. The TPR and FPR values of the model are shown against one another for various threshold values to determine the ROC AUC score. The percentage of true positive cases that the model correctly classifies as true positive is represented by the TPR, whereas the proportion of true negative instances that the model wrongly classifies as true positive is represented by the FPR. The resulting figure is a curve that shows how well the algorithm can differentiate between positive and negative cases when the threshold for classification is altered.

The ROC AUC score is the area that is underneath this curve, which has a range of 0 to 1. A score of 0.5 denotes guessing at random, whereas a score of 1.0 denotes the flawless separation of positive and negative groups. The model's performance is summed up in a single number by the ROC AUC score across all potential threshold values. It is especially helpful when the classes are unbalanced because it gives a total assessment of how well the model can categorize both beneficial and detrimental examples. The ROC AUC score is particularly helpful since it enables the user to select the best threshold for classification based on their unique requirements whenever the cost of false positives and false negatives isn't equal.

6.4. Model Architecture and parameters

The choice of initial parameters in machine learning models is influenced by existing research and literature. Different initial parameters are explored during experimentation to achieve optimal results. Factors such as model complexity, problem-specific considerations, generalization, over fitting, optimization landscape, and model initialization methods all play a role in determining these parameters. Researchers aim to strike a balance between model performance and generalization to new data. In this specific case, various types of architectures, parameters, pre-trained tokenizers, and pre-trained word embedding have been used for different models.

The LSTM model includes LSTM layers with 32 neurons each, followed by a Dense layer with 64 neurons and an activation function. A dropout rate of 0.5 is employed to prevent overfitting. The model also has two final Dense layers with 32 neurons and a sigmoid activation function. A pre-trained word embedding of dimension 100 is added for the Amharic language. The CNN model architecture consists of a conv1d layer with 32 neurons, followed by a global max pooling 1d layer and another Dense layer with 32 neurons. A dropout rate of 0.5 is used to prevent overfitting. Similarly, a pre-trained

word embedding of dimension 100 is incorporated for the Amharic language. The RNN model architecture consists of a simple RNN layer with 32 neurons, 0.2 l2 kernel regularization, 0.2 l2 recurrent regularization, a batch normalization layer, another dense layer with 32 neurons, and a dropout layer with 0.5. It is followed by a fully connected layer with a sigmoid activation function.

The GRU model architecture comprises a GRU layer with 32 neurons and 0.2 l2 kernel regularization. It is connected to a fully connected layer with a sigmoid activation function. A dropout rate of 0.2 is applied before the last layer to prevent overfitting. The bi-directional LSTM model architecture includes a bi-directional LSTM layer with 32 neurons, 0.01 l2 kernel regularization, and 0.01 recurrent regularization. It is connected to a dense layer of 32 neurons and a dropout of 0.2. Finally, a fully connected layer with a sigmoid activation function is employed. A dropout rate of 0.5 is used before the last layer to prevent overfitting.

6.5. Experiment Results

6.5.1. LSTM Model Results

The implementation of additional data, pre-trained word embeddings, and pre-trained tokenizers allow us to enhance the extraction of meaningful insights and predictions from textual data. However, during the training of our model, we carefully tackle the challenges of overfitting and underfitting to achieve the most appropriate model fit. By adopting the appropriate methodology, we ensure that our models effectively capture the subtleties of language, resulting in reliable and valuable outcomes. Our LSTM model demonstrated a precision of 80% on the training set and a test accuracy of 79.41% on the test set and the ROC AUC for the model was 0.87. The table below illustrates the precision, recall, and F1 score for both positive and negative classes:

Table 2: LSTM precision, Recall, F1 score Result

Metric	Negative	Positive
Precision	0.79	0.80
Recall	0.74	0.84
F1 Score	0.76	0.82

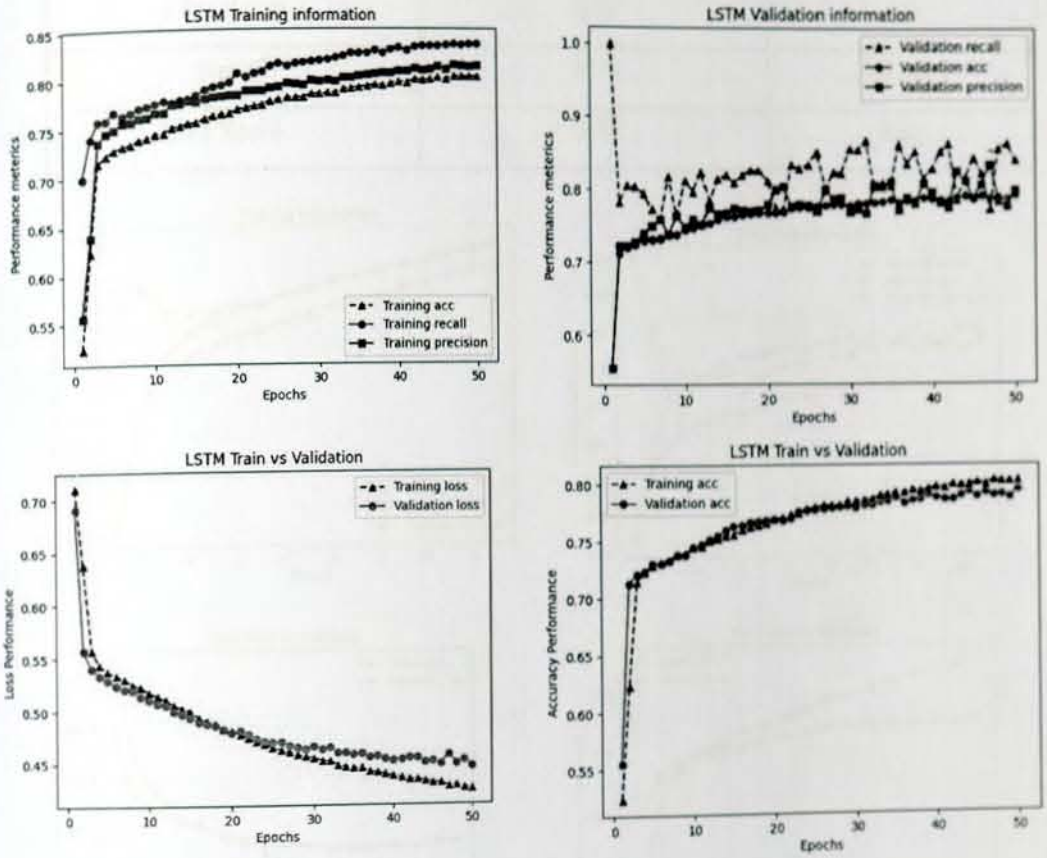


Figure 9: LSTM, Training and Validation information

6.5.2. CNN Model Results

Following the incorporation of additional data and pre-trained word embeddings, a noteworthy enhancement in the performance of our model has been observed, while successfully avoiding any instances of overfitting. Notably, the convergence speed of the model surpassed that of the LSTM architecture. However, it is important to acknowledge that the accuracy of our model fell short of that achieved by LSTM. It is imperative to bear in mind that the selection of the most appropriate model hinges upon the precise specifications of the problem at hand. Our CNN model demonstrated an accuracy of 82.9% on the training set and 77% on the test set. Furthermore, the model exhibited a Receiver Operating Characteristic Area under the Curve (ROC AUC) value of 0.85. For further evaluation, the precision, recall, and F1 score about both positive and negative classes are presented in the following table:

Table 3: CNN precision, Recall, F1 score Results

Metric	Negative	Positive
Precision	0.75	0.79
Recall	0.74	0.80
F1 Score	0.74	0.80

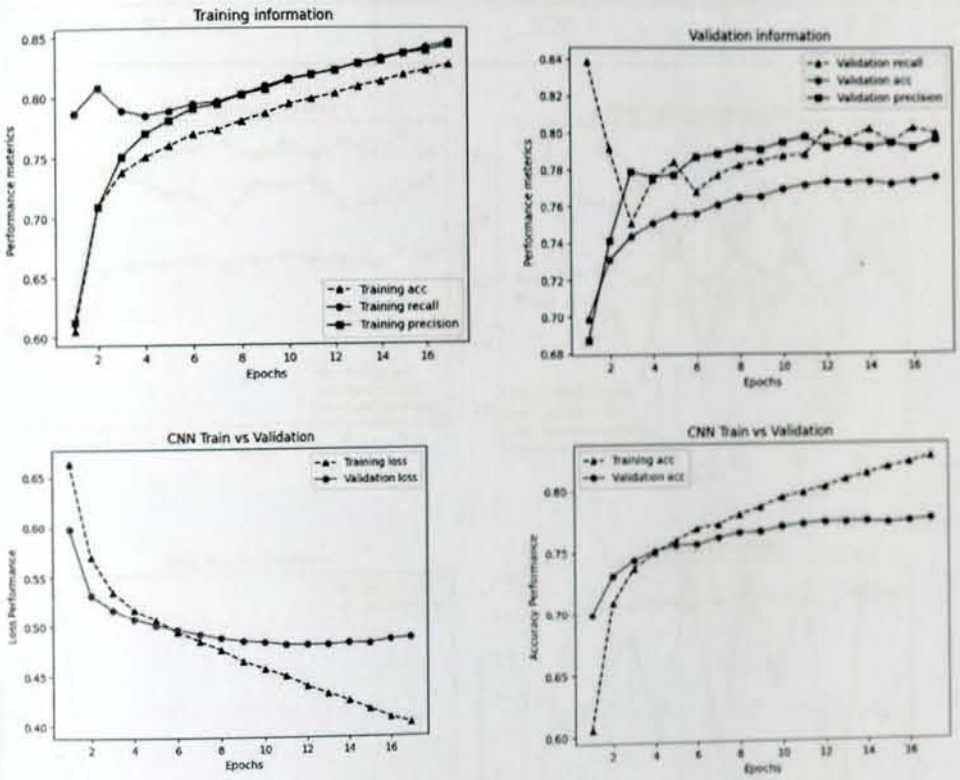


Figure 10: CNN, Training, and Validation information

6.5.3. RNN Model Results

The introduction of additional data and pre-trained word embeddings yielded positive outcomes by effectively mitigating overfitting and enhancing the performance of the model. However, despite these notable improvements, the model under consideration displayed the lowest performance among all the models explored, exhibiting a considerable degree of variability. This emphasizes the significance of employing diverse approaches and techniques when developing machine learning models, as well as the necessity for further experimentation to effectively address overfitting and optimize the model's performance. Regarding our RNN model, it achieved an accuracy of 73.1% on the training set and 73.3% on the test set. Furthermore, the model yielded a Receiver Operating Characteristic Area Under the Curve (ROC AUC) value of 0.81. The precision, recall, and F1 score for both the positive and negative classes are provided in

the following table:

Table 4: Precision, recall, and F1 results for the RNN model

Metric	Negative	Positive
Precision	0.72	0.75
Recall	0.67	0.79
F1 Score	0.70	0.77

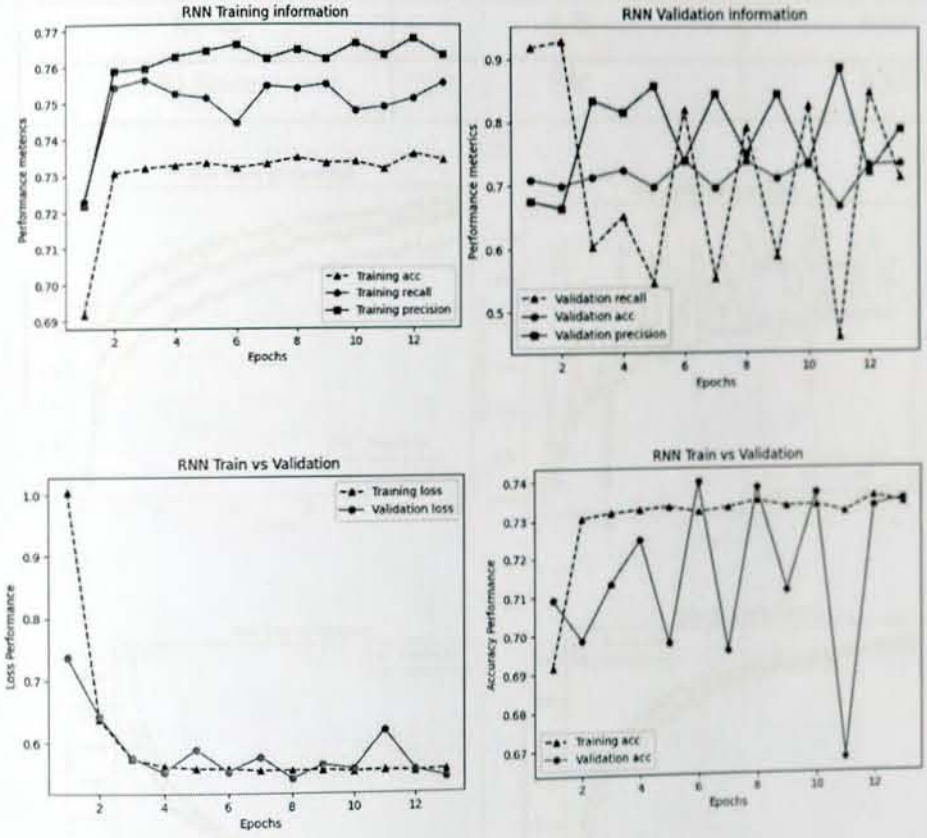


Figure 11: RNN, Training, and Validation information

6.5.4. GRU Model Results

Among the various models investigated, our GRU (Gated Recurrent Unit) model has demonstrated superior performance, exhibiting notable gains in accuracy and reduced loss. Particularly noteworthy is the GRU model's ability to make improved predictions on the test set, showcasing its proficiency in capturing the underlying patterns within the input data. This outcome serves as a testament to the efficacy of the GRU architecture for tasks involving sequential data modeling. The findings obtained from this study have the potential to significantly enhance the performance of diverse applications that rely on sequence modeling techniques. our GRU model achieved an accuracy of 82.972% on the

training set, while on the test set, it achieved an accuracy of 82.49%. Additionally, the model exhibited a Receiver Operating Characteristic Area Under the Curve (ROC AUC) value of 0.90, signifying its strong discriminatory power. The precision, recall, and F1 score for both the positive and negative classes are presented in the following table:

Table 5: GRU precision, Recall, F1 score Results

Metric	Negative	Positive
Precision	0.81	0.84
Recall	0.79	0.85
F1 Score	0.80	0.86

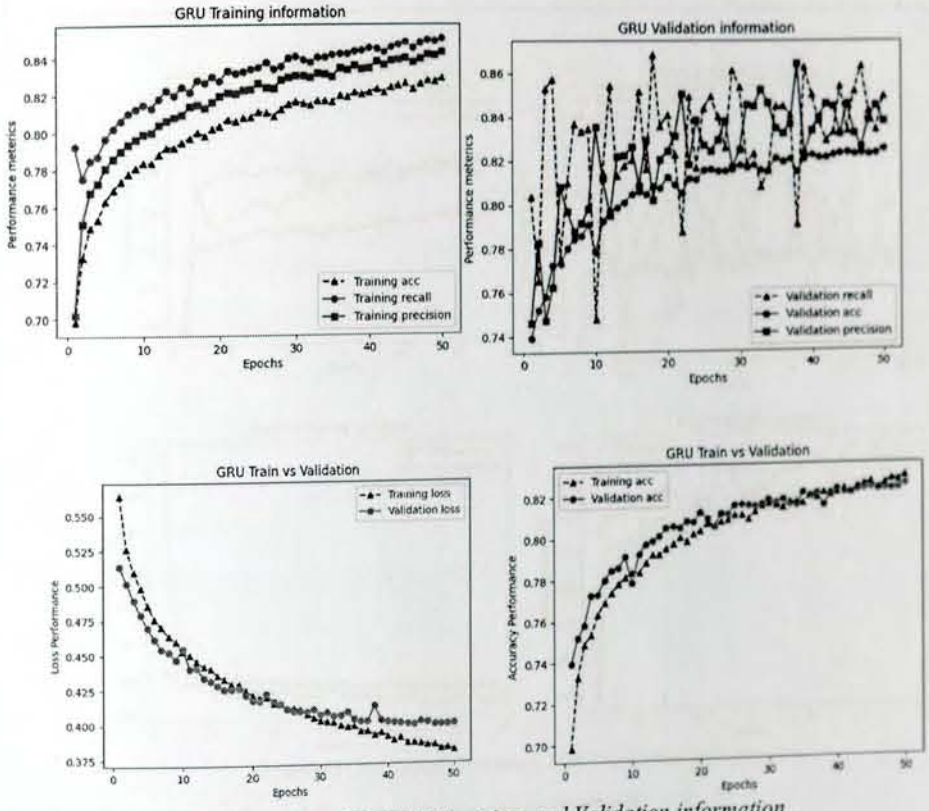


Figure 12: GRU, Training, and Validation information

6.5.5. Bi-LSTM Model Results

The bi-directional LSTM (Long Short-Term Memory) model exhibited a positive characteristic by demonstrating no signs of overfitting, which is a notable advantage. However, in comparison to the other models evaluated, its performance was relatively subpar. The model struggled to effectively capture the underlying relationships and patterns present in the data, resulting in its diminished performance. our bi-LSTM model

achieved an accuracy of 74.6% on the training set and 74.2% on the test set. Additionally, the model obtained a Receiver Operating Characteristic Area Under the Curve (ROC AUC) value of 0.83, indicating a moderate level of discriminatory power. The precision, recall, and F1 score for both the positive and negative classes are provided in the following table:

Table 6: Bi-LSTM model precision, recall, and F1 score results

Metric	Negative	Positive
Precision	0.72	0.71
Recall	0.72	0.74
F1 Score	0.72	0.72

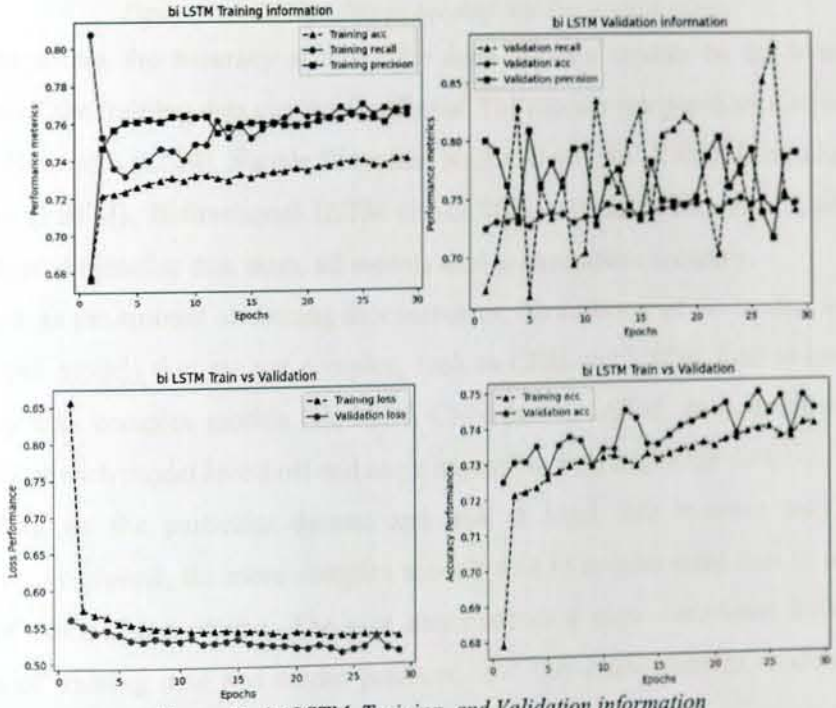


Figure 13: Bi LSTM, Training, and Validation information

6.5.6. Data size and model accuracy dependency result

Increasing the size of the dataset has been observed to greatly enhance the accuracy of algorithms. This suggests a direct relationship between the number of data instances and the resulting accuracy, indicating that as the dataset grows, the accuracy of the algorithms also increases.

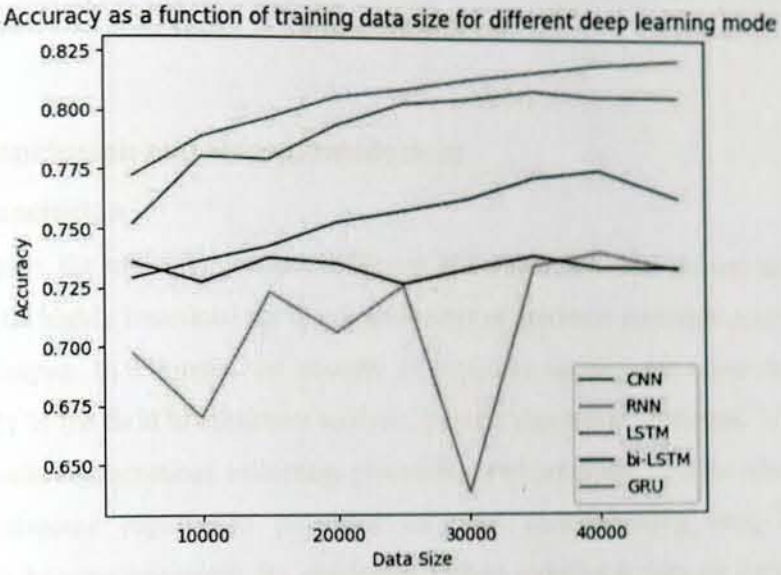


Figure 14: Data size effect on five deep learning model accuracy.

The plot shows the accuracy achieved by deep learning models on the Y-axis, as a function of the training data size on the X-axis. The models compared are Convolutional Neural Networks (CNN), Simple Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), and Gated Recurrent Unit (GRU) algorithms. At smaller data sizes, all models tend to have lower accuracy.

However, as the amount of training data increases, the accuracy of the models improves. In general, models that are not complex, such as GRU and LSTM, tend to have better accuracy than complex models like RNN, CNN, and Bi-LSTM. At a certain point, the accuracy of each model levels off and stops improving with additional data.

Depending on the particular dataset and task at hand, this moment may change. However, in general, the more complex models tend to require more data to reach this point of diminishing returns. The plot demonstrates a clear correlation between the volume of training data and model precision. and that more complex models tend to perform better with larger amounts of data.

Chapter Seven

7. Conclusion and Recommendations

7.1. Conclusion

In conclusion, the utilization of data collection and annotation methods and technologies proves to be highly beneficial for the development of Amharic sentiment analysis of the political corpus. In Ethiopia, the scarcity of available datasets for research purposes, particularly in the field of sentiment analysis, poses a significant challenge. With limited AI and media organizations collecting, processing, and providing data for academic use, and cumbersome registration processes on most crowdsourcing sites, alternative approaches become necessary. By employing Python scraping scripts for data collection and leveraging freely available websites to facilitate manual annotation processes, researchers can overcome these obstacles and obtain valuable datasets for academic research.

Furthermore, this study demonstrates that deep learning models yield the most accurate results when analyzing sentiment in Amharic. Factors such as the size of the training data, regularization techniques, and optimization strategies greatly influence the performance of Amharic sentiment analysis models. The implementation of additional data, pre-trained word embeddings, and pre-trained tokenizers enhance the extraction of meaningful insights and predictions from textual data in the supervised approach for Amharic sentiment classification.

Among the classifiers evaluated, including GRU, LSTM, CNN, Bi-LSTM, and RNN, respective results (82.49%, 79.41%, 77%, 74.2%, 73.3%), the GRU classifier achieves the highest accuracy, with results of 82.49%, outperforming LSTM, CNN, Bi-LSTM, and RNN classifiers. This indicates that, despite the limited data available, less complex models like the GRU classifier are more suitable for supervised Amharic sentiment classification compared to LSTM, CNN, Bi-LSTM, and RNN deep classifiers. Moreover, this study highlights the deep learning model's ability to learn intricate patterns but also emphasizes the risk of overfitting, especially when working with small datasets. To mitigate this, employing various regularization and optimization strategies becomes essential to obtain the best results. In summary, this research underscores the significance of employing data collection and annotation methods and technologies for Amharic sentiment analysis. It also emphasizes the advantages of utilizing deep learning

models and the importance of careful consideration when selecting classifiers, regularization techniques, and optimization strategies. By addressing these factors, researchers can maximize the accuracy and effectiveness of Amharic sentiment classification and extract meaningful insights from limited data.

Here are some remarks that could be forwarded to readers of a thesis or future developers of sentiment analysis:

I'd like to share a few observations based on the results of this thesis. First of all, it is critical to emphasize that although this study offers encouraging results for analyzing feelings in the Amharic language, more research is required to assess the models' generalizability and robustness using diverse datasets and situations.

Moreover, the practical implications of this study are vast, especially in disciplines including consumer feedback analysis, market research, and social media monitoring in Amharic-speaking regions. The potential for sentiment analysis in these areas is enormous, and Future research can be built on the findings of this study.

In addition, for future researchers who plan to use deep learning approaches, a significant volume of labeled data is necessary. Deep learning algorithms have higher performance and deeply learn patterns, which can cause overfitting in small dataset sizes. Therefore, using simple layered deep learning models like GRU, simple RNN, or reducing parameter sizes, such as dropout rate, dense layer, learning rate, batch size, and epoch size can help in reducing overfitting and improving performance.

Lastly, further research could explore the application of pre-trained Fast Text word embedding in other languages and domains. This could lead to the creation of sentiment analysis models that are more precise and effective in various languages and domains.

In conclusion, sentiment analysis in the Amharic language has significant potential, and this research offers insightful information on the creation of models using deep learning for sentiment analysis. However, there is still much to explore and improve upon, and future researchers can build upon the findings of this study to further advance the field of sentiment analysis in the Amharic language.

7.2. Recommendations

Here are some suggestions based on research examining the viability of deep learning-driven analysis of sentiment for Amharic opinions in an environment of political debate in Ethiopia:

7.2.1. **Increase the size of the dataset:** Although there is no standard

sentiment corpus available for Amharic sentiment classification experiments, the researcher can gradually improve the accuracy of each model's performance, and enhance the dataset. More data on training can significantly improve how well the suggested strategy performs.

7.2.2. **Use pre-trained word embedding:** The study found that using pre-trained word embedding achieved better performance compared to other models such as wordvec, GloVe. Therefore, the researcher should use the fast Text open-source library developed by Facebook AI Research to develop a better-performing sentiment analysis as it includes pre-trained word embedding for various languages, including Amharic. By using pre-trained models, the researcher can save time and computational resources and focus on other aspects of model development.

7.2.3. **Use different regularization and optimization strategies:** The study used several types of architecture and parameters and found that using 32 neurons, followed by a Dense layer with 64 neurons, a dropout rate of 0.5, and a final two Dense layers with 32 neurons, and the last layer with a sigmoid activation function, and using different Training Procedure achieved the best result. The researcher should consider using these specific model architectures and parameters to achieve better results.

7.2.4. **Use pre-trained Tokenizer:** The study found that using a pre-trained Tokenizer may greatly increase the precision and effectiveness of activities involving natural language processing. By improving the quality of the model. Therefore, the researcher should use a trained Tokenizer that has been properly trained to recognize words on a specific dataset or corpus of text and classify different patterns and structures within the text.

Finally, by implementing the aforementioned suggestions, the researcher can increase the viability of deep learning-powered analysis of sentiment for Amharic opinions in the context of Ethiopian political debate.

8. Reference

10 Top Ethiopia Facebook Pages 2023 — allaboutETHIO. (n.d.). Retrieved April 14, 2023, from https://allaboutethio.com/10-top-ethiopia-facebook-pages.html

Ahmed, A., & Yousuf, M. A. (2021). Sentiment analysis on bangla text using long short-term memory (lstm) recurrent neural network. *Advances in Intelligent Systems and Computing, 1309*(July), 181–192. https://doi.org/10.1007/978-981-33-4673-4_16

Ali, M., Baqir, A., Psaila, G., & Malik, S. (2020). Towards the discovery of influencers to follow in micro-blogs (twitter) by detecting topics in posted messages (tweets). *Applied Sciences (Switzerland), 10*(16). https://doi.org/10.3390/app10165715

Almalki, J. (2022). A machine learning-based approach for sentiment analysis on distance learning from Arabic Tweets. *PeerJ Computer Science, 8*. https://doi.org/10.7717/PEERJ-CS.1047

Alsayat, A. (2022). Improving Sentiment Analysis for Social Media Applications Using an Ensemble Deep Learning Language Model. *Arabian Journal for Science and Engineering, 47*(2), 2499–2511. https://doi.org/10.1007/s13369-021-06227-w

Arora, A. M. H., & Bansal, B. D. M. (2022). LSTM and Bi-LSTM Deep Learning Technique for better Tourism Services in future by analyzing Hotel Reviews. *Journal of Algebraic Statistics*, *13*(3), 3114–3123. https://publishoa.com/index.php/journal/article/view/990%0Ahttps://publishoa.com/index.php/journal/article/download/990/863

Arouri, Y., & Sayyafzadeh, M. (2022). An adaptive moment estimation framework for well placement optimization. *Computational Geosciences*, *26*(4), 957–973. https://doi.org/10.1007/s10596-022-10135-9

Ayalew, B. T. (2013). The submorphemic structure of Amharic: Toward a phonosemantic analysis. *ProQuest Dissertations and Theses*, 167.

Bagchi, T. P. (2022). *S upport Vector Machines. January*.

Balakrishnan, V., Shi, Z., Law, C. L., Lim, R., Teh, L. L., & Fan, Y. (2022). A deep learning approach in predicting products' sentiment ratings: a comparative analysis. *Journal of Supercomputing*, *78*(5), 7206–7226. https://doi.org/10.1007/s11227-021-04169-6

Belay, B. H. (2021). *Deep Learning for Amharic Text-Image Recognition : Algorithm , Dataset and Application*.

Bender, M. L. (1968). *ki"fi';*

Bergström, C., & Hjelm, O. (2019). Impact of Time Steps on Stock Market Prediction with LSTM. *Degree Project in Computer Science, Communication and Industrial Management, First Level*, 10.

Biele, C., Kacprzyk, J., Kopeć, W., Owsiński, J. W., & Romanowski, A. (2022). *Digital Interaction and Machine Intelligence* (Vol. 440). https://link.springer.com/10.1007/978-3-031-11432-8

Britzolakis, A., Kondylakis, H., & Papadakis, N. (2020). A Review on Lexicon-Based and Machine Learning Political Sentiment Analysis Using Tweets. *International Journal of Semantic Computing*, *14*(4), 517–563. https://doi.org/10.1142/S1793351X20300010

Camacho-collados, J., & Pilehvar, M. T. (2020). *Embeddings in Natural Language Processing*. 10–15.

Chen, H., Hu, S., Hua, R., & Zhao, X. (2021). Improved naive Bayes classification algorithm for traffic risk management. *Eurasip Journal on Advances in Signal Processing*, *2021*(1). https://doi.org/10.1186/s13634-021-00742-6

Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., Park, C. W., Choudhary, A., Agrawal, A., Billinge, S. J. L., Holm, E., Ong, S. P., & Wolverton, C. (2022). Recent advances and applications of deep learning methods in materials science. *Npj Computational Materials*, *8*(1). https://doi.org/10.1038/s41524-022-00734-6

Fisseha, S. G. (2020). *Amharic Open Information Extraction.pdf*. *March*.

Fraumann, G., & Colavizza, G. (2022). The role of blogs and news sites in science communication during the COVID-19 pandemic. *Frontiers in Research Metrics and Analytics*, *7*. https://doi.org/10.3389/frma.2022.824538

Garcia-Diaz, J. A., Garcia-Sanchez, F., & Valencia-Garcia, R. (2023). Smart Analysis of Economics Sentiment in Spanish Based on Linguistic Features and Transformers. *IEEE Access*, *11*(January), 14211–14224. https://doi.org/10.1109/ACCESS.2023.3244065

Getachew, Y. (2019). DEEP LEARNING APPROACH FOR AMHARIC SENTIMENT ANALYSIS UNIVERSITY DEEP LEARNING APPROACH FOR AMHARIC SENTIMENT ANALYSIS Yeshiwas Getachew (Msc) Faculty of informatics Department of IT Abebe Alemu (Assistance professor) Faculty of informatics Department. *ReaserchGet*, *March*.

Grover, H. (2009). The World's Major Languages : Amharic. *In The World's Major Languages.*, 594–617. https://doi.org/10.4324/9780203301524.ch35

Guo, J. (2022). Deep learning approach to text analysis for human emotion detection from big data. *Journal of Intelligent Systems*, *31*(1), 113–126. https://doi.org/10.1515/jisys-2022-0001

Iqbal, A., Amin, R., Iqbal, J., Alroobaea, R., & Binmahfoudh, A. (2022). *Sentiment Analysis of Consumer Reviews Using Deep Learning*.

Jalilifard, A., Caridá, V. F., Mansano, A. F., Cristo, R. S., & da Fonseca, F. P. C. (2021). Semantic Sensitive TF-IDF to Determine Word Relevance in Documents. *Lecture Notes in Electrical Engineering*, *736 LNEE*, 327–337. https://doi.org/10.1007/978-981-33-6987-

0_27

Katrekar, A. (2019). An Introduction to Sentiment Analysis About me !! *Advances in Soft Computing and Its Applications*. http://link.springer.com/chapter/10.1007/978-3-642-45111-9_41

Kaur, H., Ahsaan, S. U., Alankar, B., & Chang, V. (2021). A Proposed Sentiment Analysis Deep Learning Algorithm for Analyzing COVID-19 Tweets. *Information Systems Frontiers*, *23*(6), 1417–1429. https://doi.org/10.1007/s10796-021-10135-7

Kindeneh, M. (2020). *Opinion Mining for Amhara Broadcasting Agency News*.

Liao, X., Sahran, S., Abdullah, A., & Shukor, S. A. (2022). AdaCB: An Adaptive Gradient Method with Convergence Range Bound of Learning Rate. *Applied Sciences (Switzerland)*, *12*(18). https://doi.org/10.3390/app12189389

Linguistics, A. (2010). new LINCOM Studies in The Origin of Amharic Languages : Arabic Secret Language : The Moroccan Arabic ġuş : Arabic Secret Language : The Moroccan Arabic Secret Languages : Evidence from. *Linguistics*, *28*.

Liu, B., Tang, S., Sun, X., Chen, Q., Cao, J., Luo, J., & Zhao, S. (2020). Context-aware social media user sentiment analysis. *Tsinghua Science and Technology*, *25*(4), 528–541. https://doi.org/10.26599/TST.2019.9010021

Manish, P. (2020). Quick Guide to Evaluation Metrics For Supervised and Unsupervised Machine Learning. *Data Science Blogathon*, *X*. https://www.analyticsvidhya.com/blog/2020/10/quick-guide-to-evaluation-metrics-for-supervised-and-unsupervised-machine-learning/

Marouf, A. Al, Rokne, J. G., & Alhajj, R. (2022). Detecting and Understanding Sentiment Trends and Emotion Patterns of Twitter Users—A Study on the Demise of a Bollywood Celebrity. *Big Data and Cognitive Computing*, *6*(4). https://doi.org/10.3390/bdcc6040129

Mehta, P., & Pandya, S. (2020). A review on sentiment analysis methodologies, practices and applications. *International Journal of Scientific and Technology Research*, *9*(2), 601–609.

Meyer, R. (2017). The Ethiopic Script: Linguistic Features and Socio-cultural Connotations. *Oslo Studies in Language*, *8*(1), 137–172. https://doi.org/10.5617/osla.4422

Mu, Y. (2022). Gated Recurrent Unit Framework for Ideological and Political Teaching System in Colleges. *Scientific Programming*, *2022*. https://doi.org/10.1155/2022/9615461

Neshir, G., Rauber, A., & Atnafu, S. (2021). Meta-learner for amharic sentiment classification. *Applied Sciences (Switzerland)*, *11*(18), 1–19. https://doi.org/10.3390/app11188489

Ortis, A., Farinella, G. M., & Battiato, S. (2020). Survey on visual sentiment analysis. *IET Image Processing*, *14*(8), 1440–1456. https://doi.org/10.1049/iet-ipr.2019.1270

Reviews, C.-, Singh, C., Imam, T., & Wibowo, S. (2022). *applied sciences A Deep Learning Approach for Sentiment Analysis of*.

Rodriguez, P. L., & Spirling, A. (2022). Word Embeddings: What Works, What Doesn't, and

How to Tell the Difference for Applied Research. *Journal of Politics*, *84*(1), 101–115. https://doi.org/10.1086/715162

Salı, S. (2020). Machine Learning: A Review of Learning Types. *ResearchGate*, *July*. https://doi.org/10.20944/preprints202007.0230.v1

Saron, A., & Zelelew, H. (2021). *Amharic Question Classification System Using Deep Learning*. *April*.

Schmidt, T., Fehle, J., Maximilian, W., Richter, J., Gottschalk, P., & Wolff, C. (2022). Sentiment Analysis on Twitter for the Major German Parties during the 2021 German Federal Election. *KONVENS 2022 - Proceedings of the 18th Conference on Natural Language Processing*, *Konvens*, 74–87.

Science, C. (2021). *DESIGNING AN AUTOMATIC TEXT-INDEPENDENT AMHARIC LANGUAGE SPEAKER IDENTIFICATION*.

Sultan Ayita AAU. (2022). *Deep Learning-based Sentiment Analysis for Under-resource Language: Amharic Short Text* (Issue 1, pp. 1–68).

Technology, I. (2022). *Aspect Level Sentiment Analysis Using Hybrid Deep Learning Approach for Amharic News Comments*.

Tian, Y., Zhang, Y., & Zhang, H. (2023). Recent Advances in Stochastic Gradient Descent in Deep Learning. *Mathematics*, *11*(3), 1–23. https://doi.org/10.3390/math11030682

To, A. T. S. (2010). *SCHOOL OF GRADUATE STUDIES SENTIMENT MINING MODEL FOR OPINIONATED By : Selama Gebremeskel FACULTY OF COMPUTER AND MATHEMATICAL SCIENCES SENTIMENT MINING MODEL FOR OPINIONATED AMHARIC TEXTS*.

Upreti, A. (2022). Convolutional Neural Network (CNN): A comprehensive overview. *International Journal of Multidisciplinary Research and Growth Evaluation*, *August*, 488–493. https://doi.org/10.54660/anfo.2022.3.4.18

Wangpoonsarp, A., Shimura, K., & Fukumoto, F. (2020). Unsupervised predominant sense detection and its application to text classification. *Applied Sciences (Switzerland)*, *10*(17). https://doi.org/10.3390/app10176052

Wankhade, M., Rao, A. C. S., & Kulkarni, C. (2022). A survey on sentiment analysis methods, applications, and challenges. In *Artificial Intelligence Review* (Vol. 55, Issue 7). Springer Netherlands. https://doi.org/10.1007/s10462-022-10144-1

Xu, Q. A., Chang, V., & Jayne, C. (2022). A systematic review of social media-based sentiment analysis: Emerging trends and challenges. *Decision Analytics Journal*, *3*(June), 100073. https://doi.org/10.1016/j.dajour.2022.100073

Yang, X., Guan, J., Ding, L., You, Z., Lee, V. C. S., Mohd Hasan, M. R., & Cheng, X. (2021). Research and applications of artificial neural network in pavement engineering: A state-of-the-art review. *Journal of Traffic and Transportation Engineering (English Edition)*, *8*(6),

1000–1021. https://doi.org/10.1016/j.jtte.2021.03.005

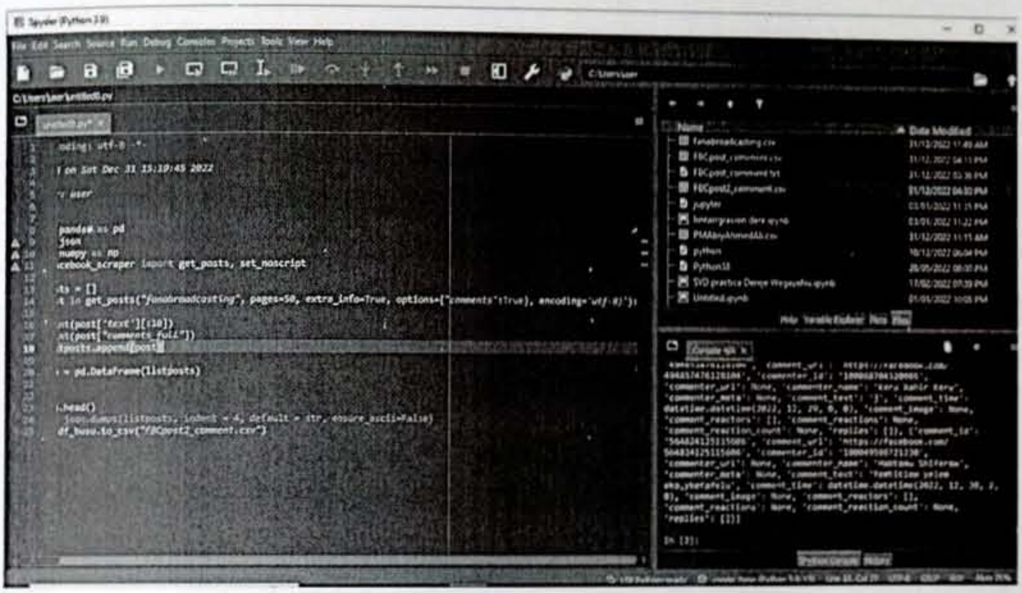
Yenkikar, A., Babu, C. N., & Hemanth, D. J. (2022). Semantic relational machine learning model for sentiment analysis using cascade feature selection and heterogeneous classifier ensemble. *PeerJ Computer Science*, *8*, 1–34. https://doi.org/10.7717/PEERJ-CS.1100

Yimam, S. M., Ayele, A. A., Venkatesh, G., Gashaw, I., & Biemann, C. (2021). Introducing various semantic models for amharic: Experimentation and evaluation with multiple tasks and datasets. *Future Internet*, *13*(11). https://doi.org/10.3390/fi13110275

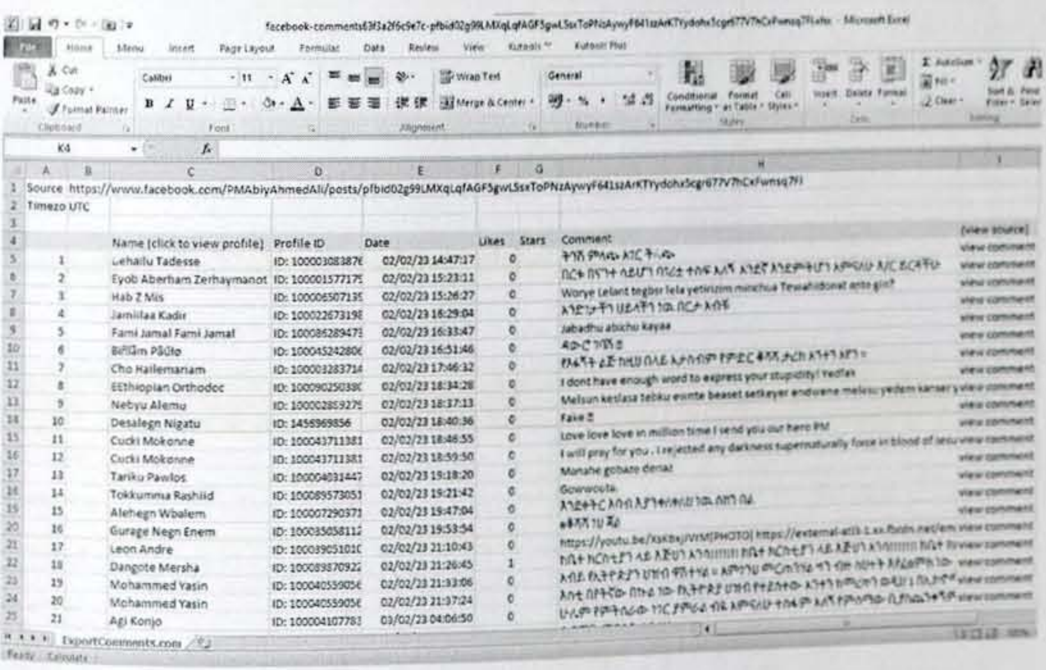
Zhang, C., Yao, M., Chen, W., Zhang, S., Chen, D., & Wu, Y. (2021). Gradient Descent Optimization in Deep Learning Model Training Based on Multistage and Method Combination Strategy. *Security and Communication Networks*, *2021*. https://doi.org/10.1155/2021/9956773

Annexes

Annex A: The user interface of Python data scraping script.



Annex B: Sample of collected data.



Annex C: Amharic character homographs.

Amharic character	Normalizeto character level	Amha ric character	Normalize tocharact er level
ሃ, ኅ, ኃ,ሐ, ሓ, ኻ	ሀ	ዓ, ኣ, ዐ	አ
ሑ, ኁ,ዅ	ሁ	ዑ	ኡ
ኺ, ሒ, ኺ	ሂ	ዒ	ኢ
ኼ, ሔ,ኼ	ሄ	ዔ	ኤ
ሕ, ኅ	ህ	ዕ	እ
ኆ, ሖ, ኾ	ሆ	ዖ	ኦ
ሠ	ሰ	ጸ	ፀ
ሡ	ሱ	ጹ	ፁ
ሢ	ሲ	ጺ	ፂ
ሣ	ሳ	ጻ	ፃ
ሤ	ሴ	ጼ	ፄ
ሥ	ስ	ጽ	ፅ
ሦ	ሶ	ጾ	ፆ

Annex D: Labialized Amharic characters

Using Labialized Amharic characters to normalize words like ተኝቱዋል or ተኝቱአል to ተኝቷል			
ሉ[ዋአ]	ሏ	ኙ[ዋአ]	ኟ
ሙ[ዋአ]	ሟ	ኩ[ዋአ]	ኳ
ቱ[ዋአ]	ቷ	ዙ[ዋአ]	ዟ
ሩ[ዋአ]	ሯ	ጉ[ዋአ]	ጓ
ሱ[ዋአ]	ሷ	ደ[ዋአ]	ዷ
ሹ[ዋአ]	ሿ	ጡ[ዋአ]	ጧ

ቁ[ዋአ]	ቑ	ጨ[ዋአ]	ጫ
ቡ[ዋአ]	ቢ	ጱ[ዋአ]	ጲ
ቹ[ዋአ]	ቺ	ፉ[ዋአ]	ፊ
ሁ[ዋአ]	ኍ	ቍ	ቁ, ቑ can be written as ቍ
ኑ[ዋአ]	ኒ	ኵ	ኩ, ኹ can be alsowritten as ኵ

Annex E: Used Python libraries during an experiment.

```
import numpy as np
import pandas as pd
import re
import emoji
import sentencepiece as spm
import os
import tensorflow as tf
from keras.models import Model
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_score, recall_score, f1_score, roc_auc_score, precision_recall_fscore_support, accuracy_score
from sklearn.metrics import classification_report
from keras.preprocessing.text import Tokenizer
from keras_preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.callbacks import EarlyStopping
from keras.regularizers import l2
from transformers import AutoTokenizer
import numpy as np
from fasttext import load_model
import fasttext
from sklearn.model_selection import KFold
from keras.models import Sequential
from keras.layers import Embedding, Flatten, Dense
from keras.preprocessing.text import Tokenizer
from keras_preprocessing.sequence import pad_sequences
from keras.layers import Embedding, BatchNormalization, Conv1D, MaxPooling1D, LSTM, SimpleRNN, Bidirectional, Dense, GlobalMaxPooling1D, Dropout,GRU, Input
```

Annex F: Sample Epoch of the model.

```
Epoch 1/50
582/582 [==============================] - 4s 5ms/step - loss: 1.0047 - accuracy: 0.6917 - precision: 0.7217 - recall: 0.7228 - val_loss: 0.7377 - val_accuracy:
Epoch 2/50
582/582 [==============================] - 3s 4ms/step - los.: 0.6381 - accuracy: 0.7505 - precision: 0.7580 - recall: 0.7541 - val_loss: 0.6428 - val_accuracy:
Epoch 3/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5747 - accuracy: 0.7719 - precision: 0.7794 - recall: 0.7763 - val_loss: 0.5769 - val_accuracy:
Epoch 4/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5626 - accuracy: 0.7727 - precision: 0.7826 - recall: 0.7723 - val_loss: 0.5527 - val_accuracy:
Epoch 5/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5585 - accuracy: 0.7734 - precision: 0.7842 - recall: 0.7712 - val_loss: 0.5893 - val_accuracy:
Epoch 6/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5581 - accuracy: 0.7720 - precision: 0.7859 - recall: 0.7644 - val_loss: 0.5534 - val_accuracy:
Epoch 7/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5553 - accuracy: 0.7732 - precision: 0.7821 - recall: 0.7746 - val_loss: 0.5771 - val_accuracy:
Epoch 8/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5553 - accuracy: 0.7748 - precision: 0.7847 - recall: 0.7739 - val_loss: 0.5422 - val_accuracy:
Epoch 9/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5562 - accuracy: 0.7734 - precision: 0.7821 - recall: 0.7751 - val_loss: 0.5682 - val_accuracy:
Epoch 10/50
582/582 [==============================] - 2s 4ms/step - loss: 0.5536 - accuracy: 0.7737 - precision: 0.7864 - recall: 0.7678 - val_loss: 0.5576 - val_accuracy:
Epoch 11/50
582/582 [==============================] - 3s 5ms/step - loss: 0.5556 - accuracy: 0.7718 - precision: 0.7832 - recall: 0.7689 - val_loss: 0.6190 - val_accuracy:
Epoch 12/50
582/582 [==============================] - 2s 4ms/step - loss: 0.5533 - accuracy: 0.7769 - precision: 0.7878 - recall: 0.7711 - val_loss: 0.5558 - val_accuracy:
Epoch 13/50
574/582 [============================>.] - ETA: 0s - loss: 0.5566 - accuracy: 0.7740 - precision: 0.7828 - recall: 0.7551Restoring model weights from the end of
582/582 [==============================] - 2s 4ms/step - loss: 0.5562 - accuracy: 0.7743 - precision: 0.7831 - recall: 0.7555 - val_loss: 0.5443 - val_accuracy:
Epoch 13: early stopping
```

Annex G: Python script for pre-processing

```
[ ] def filter_stop_words(ls):
        new_ls_stop = []
        for input in ls:
            cleaned = [w for w in input.split(" ") if not w in STOP_WORDS]
            new_ls_stop.append(" ".join(cleaned).strip())

        return new_ls_stop

    def remove_emojis(ls):
        new_ls = []
        for input in ls:
            new_ls.append(emoji.replace_emoji(input)
    )
        return new_ls

    def remove_punc_and_special_chars(ls):
        new_ls = []
        for text in ls:
            text = str(text)
            text = re.sub('[፡]', ' ',text)
            normalized_text = re.sub('[\!\@\#\$\%\^\«\»\&\*\(\)\…\[\]\{\}\;\"\"\ [illegible] ]', '',text)
            new_ls.append(normalized_text)
        return new_ls

    def remove_ascii_and_numbers(ls):
        new_ls = []
        for text_input in ls:
            text_input = str(text_input)
            rm_num_and_ascii=re.sub('[A-Za-z0-9]','',text_input)
            text = re.sub('[\'\u1369-\u137C\']+','',rm_num_and_ascii)
```