ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

EFFECT OF PREPROCESSING ON LONG SHORT TERM MEMORY BASED SENTIMENT ANALYSIS FOR AMHARIC LANGUAGE

By: Turegn Fikre

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in partial fulfillment of

Master of Science in Computer Engineering

July 4, 2020

Addis Ababa,Ethiopia

EFFECT OF PREPROCESSING ON LONG SHORT TERM MEMORY BASED SENTIMENT ANALYSIS FOR AMHARIC LANGUAGE

By: Turegn Fikre

Thesis Advisor	Examiner
Dr. Surafel Lemma	
Chairman of Department	Examiner
Dr. Yalemzewud Negash	

Submitted in Partial Fulfillment of the Requirements
for Masters of Science in Computer Engineering
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering
July 4, 2020

Acknowledgments

First and foremost, I would like to thank God for giving me his strength and ability to undertake this research study. Without his countless blessing, everything would not have been possible.

I would like to thank my advisor Dr. Surafel Lemma. The door to Dr. Surafel's office was always open whenever I go for guidance and comments. His consistent constructive ideas enabled me to a gain good research experience.

Finally, I must express my very profound gratitude to my parents, brothers, sisters, and friends for providing me with unfailing support and continuous encouragement throughout my years of study, through the process of researching and writing this thesis. This accomplishment would not have been possible without them Thank you.

Turegn Fikre

Statement of Authorship

I hereby declare that I am the sole author of this master thesis and I have not used any sources other than those listed in the bibliography that is identified as reference. I further declare that I have not submitted this thesis at any other institution to obtain a degree.

Signed----------

Date----------

Certificate

This is to certify that this thesis work entitled "Effect of Preprocessing On Long Short Term Memory(LSTM) Based Sentiment Analysis For Amharic Language" which is submitted by Turegn Fikre is carried out under my supervision and guidance for fulfilling the nature and standard required for partial fulfillment of the requirements of masters of Science in Computer Engineering. The work in this thesis has not been submitted elsewhere for a degree.

Supervisor: Dr. Surafel Lemma

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Date ----------

ABSTRACT

This paper presents effect of preprocessing on Long Short Term Memory (LSTM) based sentiment analysis for Amharic language. Sentiment analysis or opinion mining is an approach used to analyze user generated textual contents to a way that is important for decision making. User generated textual contents are found everywhere such as, social media posts, product reviews blogs and form. Developing sentiment analysis is a challenging task due to different writing styles and variation of word meanings. To analyze the sentiment of these textual contents, several approaches use labeled lexicons. In the preprocessing step of the approaches, Emojis are removed and words are stemmed. However, Emojis are usually used to express opinions.

In this research, we propose to use Emojis to automatically label texts for sentiment analysis. In addition, we investigate the impact of using unstemmed words on sentiment analysis. To evaluate the proposed labeling scheme on sentiment anaslysis, we conducted an experiment using 9,138 Amharic textual comments. The results show that integrating Emojis with lexicons for labeling gives 0.55% higher accuracy than using only lexicons. To investigate the effect of using stemming as part of preprocessing strategy, LSTM based Amharic sentiment analysis with and without stemming is conducted using 1077 comments. Result shows that applying stemming drops the accuracy of the sentiment analysis by 6.43% while using long short-term memory based sentiment analysis, and 0.43% while using bi-gram multinomial naive bayes.

Keyword: - Amharic sentiment analysis, Emoji, Natural Language Processing (NLP), Sentiment analysis, Stemming.

Contents

List of Tables

List of Figures

List of Abbreviations

The following are some abbreviations used in this Thesis document.

BI-LSTM	.....	Bidirectional Long Short-Term Memory
CBOW	.....	Continious Bag of Words
CNN	.....	Convolutional Neural Network
CPU	.....	Central Processing Unit
CSV	.....	Comma Separated Value
DNN	.....	Deep Neural Network
LSTM	.....	Long Short-Term Memory
ME	.....	Maximum Entropy
MLP	.....	Multi Layer Perceptron
NLP	.....	Natural Language Processing
NB	.....	Naïve Bayes
MNB	.....	Multinomial Naïve Bayes
NLTK	.....	Natural Language Toolkit
POS	.....	Part of Speech
RNN	.....	Recurrent Neural Network
DBN	.....	Deep Belief Network
RNDM	.....	Recursive Neural Deep Model
RF	.....	Random Forest
RNTN	.....	Recursive Neural Tensor Network
SG	.....	Skip-Gram
SVM	.....	Support Vector Machine
TF-IDF	.....	Term Frequency- Inverse Document Frequency

Chapter 1

Introduction

Sentiment analysis or opinion mining is an area in natural language processing (NLP) that deals with human feelings, judgments, and responses generated from texts [1][2]. Information gathered from a text can be facts or opinions. Facts are true expressions about certain event or situation, whereas opinions describe one's feeling towards certain event or situation [2][36]. Sentiment analysis is one of the approaches that is used to automatically identify opinions from a text. There are some challenges in developing sentiment analysis. These challenges are summarized as follows: i) usage of different writing styles; ii) one word can be positive in one situation and negative in an other. For example, **የኮምፒውተሩ ባትሪ ረጅም ሰአት ይቆያል** express a positive feeling but **ኮምፒውተሩ ለመክፈት ረጅም ሰአት ይቆያል** express a negative feeling about the same computer; iii) in the same sentence people usually express their positive or negative feeling in comparative manner [47].

Before the introduction of the web, organizations conduct surveys by providing questionnaires for different parts of a society to make a wise and effective decision. However, this time people express their opinion on the web. This increases the need for analyzing opinionated online content for various purposes. If these on-line contents are properly and efficiently analyzed, they will have many benefits for organizations, factories, and service giving companies in many ways. For instance,it helps to determine the success of an advertisement or new product launch, to give immediate feedback on products, to determine which version of a product is popular for a customer, to provide services in

a way that satisfies their customers' interest.

Sentiment analysis can be performed at three levels: document, sentence and feature or entity-aspect level [2][47]. Document level sentiment analysis is the process of determining the overall opinion of a document by assuming that each document is about a certain entity [47][2]. Aspect level sentiment analysis aims to find what people like or dislike about entities within a document or sentence. For instance, **የኮምፒውተሩ ባትሪ ረጅም ሰአት ይቆያል** the word **ባትሪ** is a feature of the entity **ኮምፒውተሩ** [47][2]. The third one is sentence level that focus on extraction of people's opinion or emotion in at the sentence or phrase level. These are two major operations in sentence-level sentiment analysis these are subjectivity classification which is a determination of sentence either as opinion or facts and polarity determination of the subjective sentence as positive, negative or neutral [2][47].

Many researches have been done for resource rich languages, such as English and Arabic. The researches followed different approaches such as Machine learning approach [28][34], deep learning [30][31][32][34][38][42], Feature based [37], Rule(Linguistic) based, Hybrid approach [36], and combination of them. However, sentiment analysis for Amharic language did not get a considerable attention among researchers for the realization of commercial and business benefits. There are only limited number of research works on Amharic sentiment analysis [1][2][3][4][5]. The reasons include morphological complexity, spelling variation, in-availability of labeled data, less existence of researched resources, lack of texts available on the web, inaccessibility and inconvenience to typing [1][2], character redundancy and absence of abbreviation rules. The few researches on Amharic sentiment analysis use approaches such as feature extraction based [2], lexicon-based [3] and machine learning-based [1][4]. These works have several limitations: (a) they use stemming as part of preprocessing. Literatures suggested that, stemming may have a positive [28][53] for or negative impact on Indonesian and Arabic twee [10][14]

sentiment analysis. This is due to the preprocessing we apply may have positive or negative impact depending on the type of dataset used and type of language it has [28] and the stemming process[14][53]. Since Amharic is a language that is highly inflectional and derivative which makes it morphologically complex. The preprocessing applied may positively or negatively affect the performance. However, there is no previous work that show whether adding stemming as part of preprocessing affect Amharic sentiment analysis positively or negatively. (b) previous works that develop automatic labeling system do not consider Emojis for comment labeling [2][3][5]. Emojis are sentiment carrying Unicode symbols that are specially used when people face difficulty in expressing their feeling and opinion using words only [39][40][41]. This thesis focuses on developing LSTM based sentiment analysis that study the effect of using labeled Emojis and lexicons on the automatic labeling system, and the effect of integrating stemming as part of preprocessing.

1.1 Statement of the Problem

Nowadays there are a lot of important data on the web that are difficult to use due to their unstructured format. However, analyzing this data can help to predict future products, economic trends and social facts [28]. Social media content is full of impurity —it contains non-textual contents and numbers that have no meaning. To make real world data clean and suitable for analysis many chained approaches can be followed. Morphological analysis especially stemming is one of the techniques used to make these contents suitable for analysis. Previous Amharic sentiment analysis researches integrate stemming as part of preprocessing [1][4]. However, it is not investigated whether stemming will improve Amharic sentiment analysis or not.

The general objective of a sentiment analysis is to classify user generated contents as positive, negative and neutral. These user-generated contents can be expressed in

different ways. Some use alphabetic characters, while others use Unicode symbols (Emojis) or combination of alphabetic characters and these symbols. The previous researches on Amharic sentiment analysis focus on lexicons for comment labeling and give little attention to these sentiment carrying symbols (Emojis)[2][3][5]. In this research, we investigate the use of Emojis to automatically label texts and the impact of using stemmed words on sentiment analysis.

This research paper will in particular aim to answer the following research questions (RQ).

RQ1.[Effect of using Emojis for labeling]: Can Emojis be used to enhance lexicon based automatic comment labeling for Amharic sentiment analysis?

RQ2.[Effect of stemming]: What is the effect of stemming on the Amharic language sentiment analysis?

RQ3.[Annotation consistency]: How the automatically labeled data (using Emojis and lexicons) is consistent with human labeled data?

1.2 Objective

1.2.1 General Objective

The general objective of this research is to investigate incorporating labeled Emojis with lexicons for comment labeling, and effect of integrating stemming as part of preprocessing on Amharic sentiment analysis.

1.2.2 Specific Objective

This research paper has the following specific objectives.

- Annotating Amharic comments using labeled lexicons and Emojis.

- Analyzing the effect of labeling text using only labeled lexicon and combining both labeled lexicons and Emojis.
- Building and testing the LSTM based sentiment classifier model that can handle the context of a text.
- Analyzing the effect of stemming on the LSTM based Amharic sentiment analysis model.
- Comparing the automatically annotated corpus with the human-annotated one to identify the gaps in the automatic annotation system.

1.3 Methodology

In order to accomplish the objectives of the research, the following procedure will be followed

i) **Literature Review:** Related works on sentiment analysis in different languages and effect of stemming on sentiment analysis will be reviewed.

ii) **Data Collection and Preparation:** Sentences will be collected from comments of movie, music, and politics related videos uploaded on YouTube. In addition, preprocessing activities will be done on the collected sentences to make the data suitable for analysis.

iii) **Data Annotation:** The preprocessed data will be annotated using both labeled lexicons and Emojis to investigate the effect of using labeled Emojis on the automatic labeling system.

iv) **Neural Network Development:** A deep neural network specifically LSTM model is developed.

v) **Evaluation:** Experiment will be conducted to test the effect of stemming and use of

labeled Emoji on Amharic sentiment analysis. The performance of the system will be evaluated in terms of precision, recall, accuracy, and F score.

1.4 Scope and Limitation

This research includes the design and development of sentiment analysis for Amharic Language using Amharic reviews that are collected from YouTube. This research on LSTM based sentiment analysis for the Amharic language is not limited to textual data but also consider Emojis, that have changed the way we express our opinion, ideas, and feeling on a certain situation or events on social medias like Facebook, YouTube and Twitter [39][40][41]. However, Amharic explicit expressions such as "**ቅኔያዊ አነጋገር**" are out of the scope of this research work.

1.5 Contribution

This research will have a contribution to the sentiment analysis of Amharic review by classifying the reviews as positive, negative and neutral. This research paper uses Long Short Term memory based sentiment analysis that can handle the context of a text.
This research paper has the following contributions.

- Designing and developing a labeling system for user-generated comments using both labeled lexicons and Emojis.
- Carrying out experiments to determine effect of incorporating Emojis for comment labeling.
- Designing and performing experiments to evaluate effect of using stemmed words on the LSTM based Amharic sentiment analysis model.
- This research paper collects 9,138 Amharic comments for future researches.

- Performing analysis to identify the gaps on the developed automatic annotation system by comparing the automatically and human annotated corpus.

1.6 Thesis Organization

The rest of this document is organized as follows. Chapter Two discuss a general overview of Amharic language, challenges in developing sentiment analysis for the Amharic Language, and how LSTM and word2vec algorithms work.
Chapter Three discuss previous research works on both resource-rich and Amharic language and state of the art approaches for sentiment analysis. Our proposed approach for sentiment analysis is elaborated in Chapter Four. The experimental setups, procedures, evaluation metrics, results, and discussion are discussed in Chapter Five. Finally, Chapter Six discusses the conclusion and future research direction on Amharic sentiment analysis.

Chapter 2

Theoretical Background

2.1 Amharic Language

Ethiopia is a country located in the horn of Africa with more than 80 nations and nationalities that have their own language and identity. Amharic which is the working language of the country is spoken by more than half of the population [1][2][8]. Amharic uses Geez or Ethiopic writing system that is written from left to right similar to English unlike other Semitic languages (Arabic and Hebrew) [1][2][8].

Amharic is second largest spoken Semitic language next to Arabic. The number of speakers of the language is increasing because of the following reasons. First, it is the working language of Ethiopia. Second, Amharic language is a written language with its own alphabets that are actively used in newspapers, books and social media [1][2][3][9].

Punctuation Marks of Amharic Language:- Punctuation marks are conventional signs that help reading and understanding of a written text. The Amharic language has about ten punctuation signs. However, mostly used signs are the end of sentence called AratNetib ("::"), question mark("?"), an exclamation of sentence called "ቃል አጋኖ" ("!") and word separator Hulet Neteb(":") [8].

2.2 Challenges of Amharic Sentimental Analysis

Applications like information retrieval, text classifications or document filtering could benefit more by the existence and availability of basic tools like stemmer, morphological

analyzer, and POS taggers. However, little is known about their effect on classification performance or retrieval accuracy due to a little number of researches on Amharic language [7].

The majority of sentiment analysis researches is conducted for resource-rich languages like English and Arabic languages. The finding of these researches however, cannot directly be used for Amharic language. Amharic language has a unique characteristic that are different from the languages used in the studies. These characteristic that could impose a challenge on the direct use of the defined approaches for other languages are discussed as below.

Character Redundancy: Some Amharic Language characters have the same sound but different forms. There is no clear guideline that tells where to use each characters [8]. For instance, the word "**ጸሀይ**" ('sun') can be represented in Amharic as **ጸሀይ, ጸሐይ, ጸኀይ, ፀሀይ, ፀሐይ** [8], people use the characters **ሀ, ሐ** or **ኀ** and **ጸ** or **ፀ** interchangeably to write the same word.

Table 2.1 below shows the characters that have different form with the same sound.

Character	Other forms of the character
(Hä)	**ሀ, ሃ,ሐ,ሓ** and **ኃ**
(Sä)	**ሰ, ሠ**
(ä)	**አ, ዐ,ኣ** and **ኣ**
(tsä)	**ፀ, ጸ**

Table 2.1: Different characters with the same sound [8]

Morphological Complexity : Languages that are under Semitic family are rich in morphology. As Amharic is one of the Semitic family, a single word has many variant forms. For instance, the word **አደረሰ** can be inflected to **ያደርሳል, አስደረሰ, ያስደርሳል** and other forms.

Absence of Abbreviation Rule: There is no clear rule in abbreviating Amharic words. For instance, the word "**ዶክተር**" can be abbreviated as **ዶር, ዶ.ር**, and **ዶ/ር** [8].

Spelling Variation: The same word can be written in different forms. For example, the word computer can be written as **ኮምፒዩተር** or **ኮምፒውተር** [2][8].

Less Resource: Amharic Language is one of the least researched and under resourced language in terms of processing tools and electronic resources [7]. For instance, there is no dataset for Amharic language sentiment analysis and other NLP researches.

2.3 Word Representations Approaches

The preprocessed data should be changed to a format that is suitable to the neural network classifier. Mainly, there are three word representation approaches; these are TF-IDF, bag of words and word embedding.

2.3.1 Bag of Words Model

Bag of words model takes raw data as input and counts the number of occurrences of each word in a document as output. In Bag of words model, word lists are paired with their amount of occurrence per document [46][1]. Bag of words may use Unigram, Bigram, Trigram or N-gram for creating a vocabulary of a document that is used to classify a document [46].

Unigram, Bigram and N-gram:- Unigram, Bi-gram, Trigram and N-gram are different approaches for creating a vocabulary. Uni-gram is taking each word as a token and N-gram is taking N pairs of words as a token, N can be 1, 2, 3 or any other number[28].

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

TF-IDF is another way of converting raw data to numerical features that can be input to any NLP or machine learning algorithms to discover hidden topics [28][46]. TF-IDF mainly measures the relevance of words than their frequency count because higher

frequency count does not give high information about the document. This makes TF-IDF better transformation than the bag of words model [28][46].

2.3.3 Sentiment Representation using Word Embedding

Word embedding is the most well-known and widely used word representation approach in NLP applications that allows words with similar meaning to have a similar representation. Word embedding is a feature learning technique that aims to map words from a vocabulary of real numbers into low dimensional space [46][50].

This continuous space representations using word embedding can be computed for capturing both syntactic and semantic information about words [46]. Word embeddings as input achieved a good performance in many NLP tasks like sentiment classification of a document by adding semantic information of the documents [46]. Mainly, there are three well known word embedding algorithms: Glove [16][46][20], FastText [17][19] and Word2Vec [15][20][50].

Glove: Global vector for word representation is unsupervised learning algorithm developed by Pennington et al. to get vector representation of words [17][46][20].

FastText: FastText is a library written in c++ for efficient representations of words and sentences classification in both supervised and unsupervised learning. FastText supports both continuous bag of words (CBOW) or Skip-gram models [17][19].

Word2Vec

Word2vec which is developed by Mikolov et al [15] is an efficient word embedding model from the raw text [20]. The word2vec model can be applied in many NLP applications such as sentence classification, sentiment analysis, and other problems. In word2vec vectors of the words in the sentence are fed to the classifier model as input to handle

the meaning of the sentence.

Vectors of word2vector model give an increase in performance on NLP applications due to a large number of parameters. Pre-trained word2vec do not need update during training. This reduces the occurrence of overfitting which increases the performance of the neural network classifier.

There are two word2vec architectures for learning word embedding namely, Skip-Gram and a Continuous Bag of Words (CBOW) [20] [25][21][50]. Skip-gram and CBOW are similar except CBOW predicts a context of single target word whereas Skip-gram(SG) predicts the context of a sentence using neighboring words [21][50]. Both CBOW and SG can be used with different window sizes (10, 50, 100, 200 and 300) [21] depending on the amount of training data. Skip-gram models are efficient with small amount of training data whereas CBOW is efficient with a large amount of training data.

Word2vec is a two-layer neural network such as input, hidden, and output layers. Input layer takes input sentences of a corpus represented as x_{1k}, x_{2k} up to x_{ck} for CBOW and x_k for SG as shown in Figure2·1. The hidden layer processes texts of the input layer by vectorizing words. Each inputs in the input layer will be multiplied by some weight represented as W_{Vx}bscript N. The output layer gives a set of vectors or feature vectors that represent words in that corpus. Word2vec gives distributed numerical representations of word features that are context of individual words. Vectors are represented as y_{1j}, y_{2j} up to y_{cj} fo SG and y_j for CBOW as shown in Figure2·1.

The input to a word2vec model is a list of words in a sentence. A sliding window is used to calculate vectors of each word and distribute the vectors on a vector space. The vector of a word is updated each time when the word appears more than once in the corpus [21]. Figure 2·1 shows architecture CBOW and Skip-Gram models of word2vec.

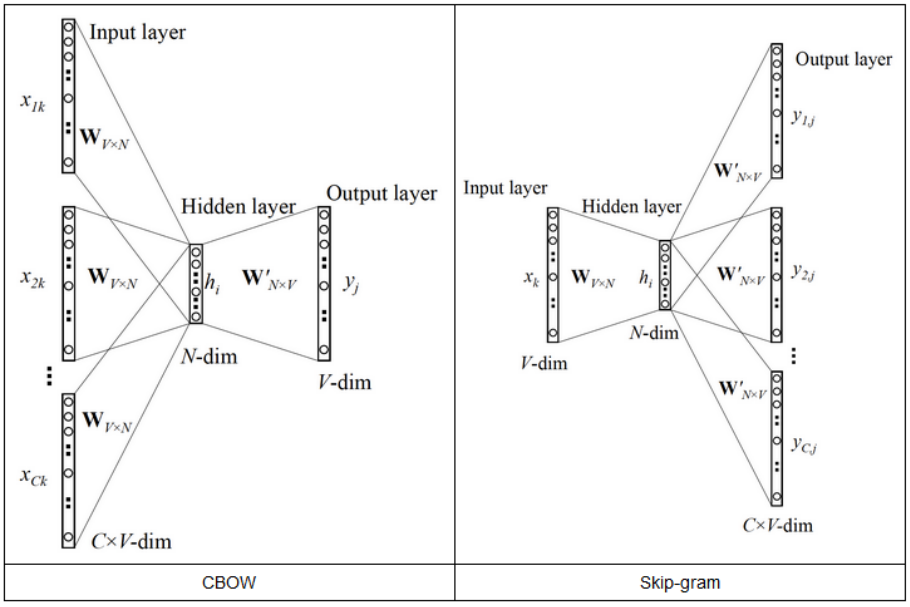

Figure 2·1: Word2vec Archetecture[25][50].

2.4 Deep Neural Network

2.4.1 Recurrent Neural Network(RNN)

Recurrent Neural Networks are a subclass of Artificial Neural networks. Decisions made by RNN are dependent on what is learned in the past and during training in the present. Recurrent Neural Network (RNN) is a DNN that is adapted to sequence data, and as a result, the RNN is also extremely expressive [22][23] Recurrent Neural Network has two inputs, these are the present, and the recent past. When RNN makes a decision, it considers the current input and also what it has learned from the inputs it received previously.

During the training of RNN, the information goes in a loop again and again which results in very large updates to neural network model weights. This large updates makes RNNs fail in maintaining the gradient called vanishing gradient decent. In theory, RNN is capable of predicting long term dependencies [22] but the vanishing gradient decent

makes the RNN not to maintain long term dependencies. Figure 2·2 shows an architecture of RNN.

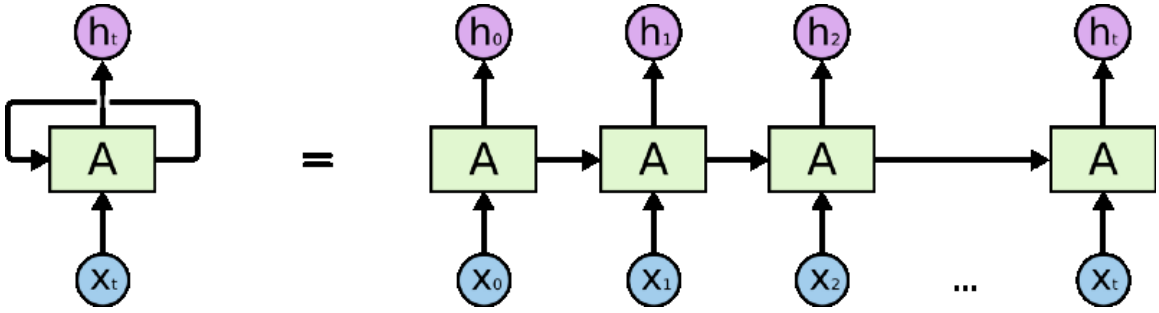

Figure 2·2: An unrolled Recurrent Neural Network[22].

2.4.2 Long Short-Term Memory (LSTM)

A special kind of RNN called the Long Short-Term Memory (LSTM) architecture, as a solution to the vanishing gradient problem is developed by Hochreiter and Schmidhbur [22][23][46].

LSTM has the capability of learning long term dependencies, and this enables RNN's to remember their inputs over a long period. LSTM contain its information in a memory that can be seen as a gated cell in which the cell decides whether to store or delete information based on the importance of LSTM network assigned to the information [22][23][46]. The LSTM network can store, write, read, delete information on its memory. The memory of LSTM is similar to computers memory.

LSTM networks can remove or add information to the cell state using three gates such as input gate (to let or not to let the new input), forget gate (to delete the information if it is not important) and output gate (to see the impact of the information on the current time step) [22][46].

LSTM has been widely used for speech recognition, language modeling, sentiment analysis,and text prediction. LSTM Networks have a chain-like structure as RNN. However,

the repeating modules of LSTM have four interacting neurons instead of a single neural network layer which is the case of RNN. Figure 2·3 below shows the interacting LSTM network layers. The symbols used in Figure 2·3 are defined in Figure 2·4 with their corresponding operation.

The notations in figure 2·3 are presented below.

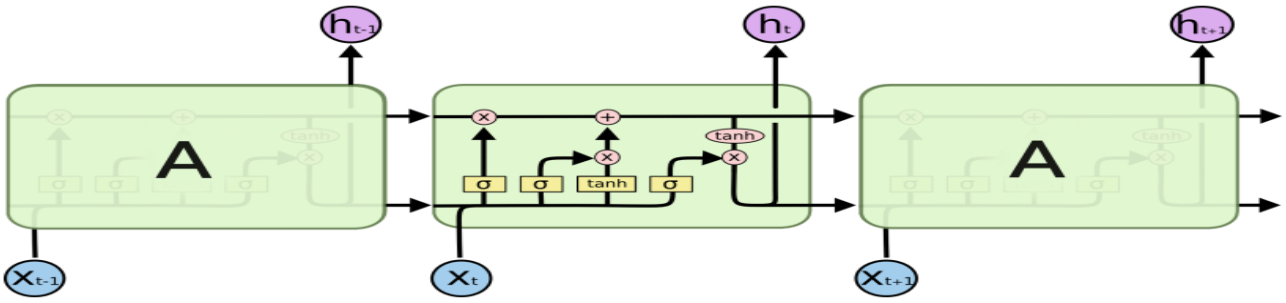

Figure 2·3: LSTM Layers[22][23].

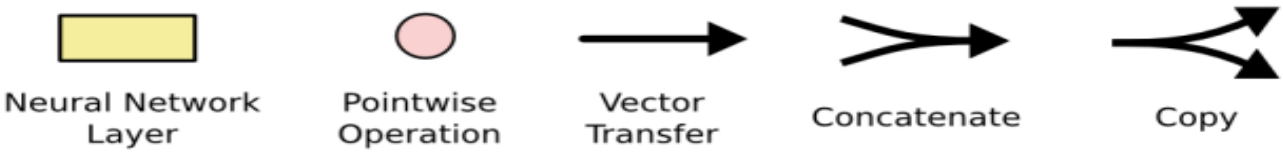

Figure 2·4: Symbols used in LSTM Networks[22][23].

+= Adding information

c(t-1)= Memory from last LSTM unit

X(t)= Current input

c(t)= New updated memory

h(t)= Current output

σ(Sigmoid layer)

tanh= tanh layer

X= Scaling of information(point wise multiplication)

σ(Sigmoid layer): A Sigmoid layer decides whether the new information should be updated or ignored. The sigmoid layer gives a number between zero (let nothing through) and one(let everything through) [22].

tanh (tanh layer): To overcome the vanishing gradient problem, we need a function

whose second derivative can sustain for a long range before going to zero.

Steps to develop LSTM network

1. Decide which information to remove:- First decide which information to remove from the cell using a sigmoid layer called forget gate layer.

Figure 2·5 shows a forget layer of the LSTM layer.

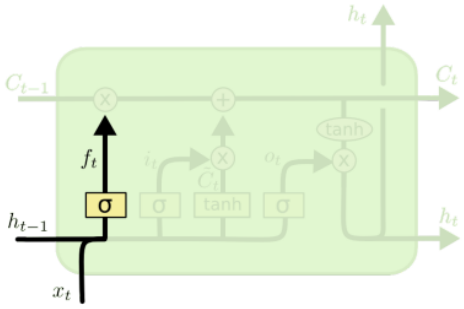

Figure 2·5: Forget layer of LSTM Network[22][23].

$$\mathbf{ft} = \sigma(\mathbf{Wf}.[\mathbf{ht-1}, \mathbf{xt}] + \mathbf{bf}) - - - [22, 23, 46] \tag{2.1}$$

2. Decide what new information to store:- After deciding which information to remove you have to decide what information you are going to store in the cell using input gate layer (decide which values we will update) and tanh layer (creates new candidate values that could be added to the network). Figure 2·6 and Equations 2.2 and 2.3 show how information goes in the input and tanh layer.

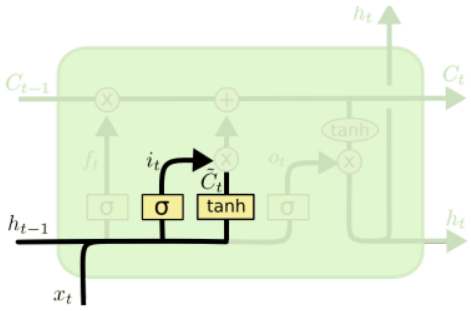

Figure 2·6: Input and tanh of LSTM Network[22][23].

$$\mathbf{it} = \sigma(\mathbf{Wi}.[\mathbf{ht-1}, \mathbf{xt}] + \mathbf{bi}) - - - - - [22, 23] \tag{2.2}$$

$$\acute{\mathbf{C}}\mathbf{t} = \mathbf{tanh}(\mathbf{WC}.[\mathbf{ht-1}, \mathbf{xt}] + \mathbf{bC}) - - - - [22, 23, 46] \tag{2.3}$$

3.Update the old cell state into the new cell state:- Update the old cell state by multiplying the new cell state using what you forget earlier, and then add the new candidate cell state. The new candidate values is scaled by how much we decided to update each state value. Figure 2·7 and Equation 2.4 show how an information is updated in the LSTM memory cell.

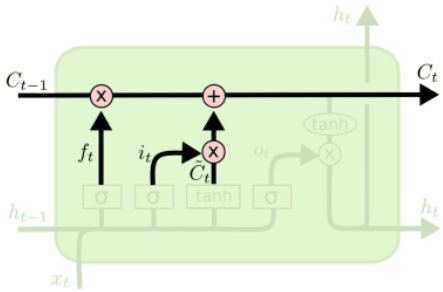

Figure 2·7: Memory cell update layer of LSTM Network[22][23].

$$\mathbf{Ct} = \mathbf{ft} * \mathbf{Ct-1} + \mathbf{it} * \acute{\mathbf{C}}\mathbf{t} - - - - [22, 23, 46] \tag{2.4}$$

4. Decide the Output:- Finally decide the output which is based on our cell state. First run the sigmoid layer to decide what parts of the cell state we are going to output and then multiply it with output of sigmoid layer [22].

Figure 2·8 and Equations 2.5 and 2.6 show how the LSTM network determine the output based on what it learn in each LSTM layers.

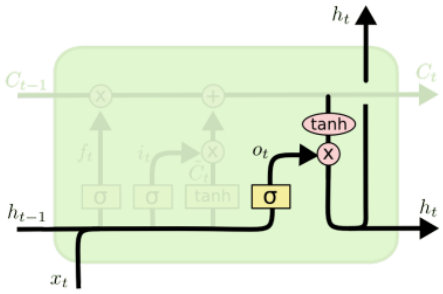

Figure 2·8: Output layer of LSTM Network[22, 23].

$$\mathbf{Ot} = \sigma(\mathbf{Wo}[\mathbf{ht-1}, \mathbf{xt}] + \mathbf{bo}) - - - [22, 23] \tag{2.5}$$

$$\mathbf{ht} = \mathbf{Ot} * \mathbf{tanh}(\mathbf{Ct}) - - - - [22, 23, 46] \tag{2.6}$$

2.5 Naive Bayes Algorithm

Naive Bayes is a probabilistic model that is based on Bayes assumption [6]. Naive Bayes is not a single algorithm but a family of classification algorithms that consists Multinomial, Gaussian and Bernoulli [6]. These Naive Bayes algorithm forms can be implemented in Uni-gram, Bi-gram and Tri-gram word representation approaches [28][1]. Naive Bayes algorithms has shown good performance for various NLP tasks including sentiment Analysis and text classification [1][6][28]. However, in Naive Bayes the probability of each word in a document is independent of word's position and context [6][28].

Multinomial Naive Bayes generate feature vectors to represent frequencies of events

based on multinomial distribution. Multinomial model is typically used for document classification [6][46]. In Bernoulli, events features are independent binary variables that is popular for document classification as multinomial model. In Gaussian Naive Bayes each features assigned continuous values according to a Gaussian distribution [6]. Naive Bayes algorithms are fast relative to other classification algorithms. Naive Bayes needs enough data to understand the probabilistic relationship of each attribute. The algorithm performs well, even with less training data [6][46].

Chapter 3

Literature Review

3.1 Sentiment Analysis Approaches

Sentiment analysis is a process of developing an automated opinion mining system from sources like text, speech and database. There are many tools and techniques to develop an automated sentiment classification system from user-generated data. Generally, there are two main approaches for developing sentiment analysis these are supervised which uses annotated data, and unsupervised approach that does not need annotated data [12][13]. Unsupervised approaches can further be classified as lexicon-based, Rule-based and combination of these approaches [12].

In the following sub-sections, we categorized and discussed sentiment analysis approaches into lexicon based, rule-based, machine learning based, and deep learning-based approaches. Related works to Amharic sentiment analysis and effect of stemming on the sentiment analysis are discussed in section 3.2 and 3.3.

3.1.1 Lexicon Based Sentiment Analysis

Lexicon based sentiment analysis is unsupervised technique. This technique involves collecting and building a set of lexicon words that have positive and negative polarities. The collected lexicons combined with some rules of the languages are used to determine the sentiment of a given sentence.

Muhammed Zubair el al. [27] proposed lexicon-enhanced sentiment analysis framework

using a rule-based scheme to improve sentiment detection and classification accuracy. The developed system worked by Muhammed Zubair integrates a set of modifiers, emoticons, general-purpose, and domain-specific lexicon words [27].

3.1.2 Rule-Based Sentiment Analysis

The rule-based sentiment analysis approach uses lexicon words and some linguistic rules of the language [12][13][18]. This approach has many challenges because it requires deep knowledge of the language to develop rules. Generating rules is time-consuming and challenging. It requires a lot of manual work, and the performance of the system becomes much less because it always generates the rules.

Hutto and Gilbert [18] has developed VADER: a parsimonious rule-based model for sentiment analysis of social media text using 1,915 and 2291 labeled positive and negative words. VADER has better accuracy (F score=0.96) than a single human raters(F score=0.84).

"A Rule-Based Approach for effective sentiment analysis" is developed by Yang and Shih [13] using 3000 and 442, 509 annotated and un annotated product reviews respectively. A 3 phase sentiment analysis is developed that achieves 72.04% macro-F score and 68.96% micro-F score measures [13]. These phases are feature extraction, opinion sentence extraction, and opinion orientation identification.

The last rule based sentiment analysis is a classification of drug reviews using rule-based linguistic approach [12]. The paper develops a model by breaking a sentence into independent clauses and applies different rules on the clauses using dataset from www.druglib.com, and 9630 general, and 10 domain lexicons. The developed model handles complex rules and gives better performance than class one and two SVM. The developed model, class one SVM, and class two SVM give an accuracy 78%, 69%, and 74% respectively [12].

3.1.3 Machine Learning Based Sentiment Analysis

A lot of research is done on sentiment analysis using machine learning algorithms. Machine learning approaches can be supervised, un-supervised and semi-supervised. Supervised learnings require labeled data for training and feature extraction whereas unsupervised learning techniques do not need labeled data. Approaches based on semi-supervised algorithms use both labeled and unlabeled data [34]. These machine learning algorithms may be Naïve Bayes,Support Vector Machine (SVM), Maximum Entropy (ME), Random Forest, and others [28].

Ahmad Kamal proposed to use a supervised machine learning (Naïve Bayes, J48, Multi-layer Perceptron (MLP)) techniques for classifying subjective and objective sentences from customer reviews. The paper applies linguistic and semantic analysis of texts to mine feasible feature opinion pairs from subjective sentences that consists of 400 electronic product review from www.amazon.com [36].

A sentiment analysis that use supervised machine learning algorithm (SVM, RNN, Random Forest, NB, and ME) using 752 negative and 1301 positive labeled movie reviews is presented by Romero Llombart, Ò. [28]. Pang, B. [35] combine machine learning algorithms (J48, and MLP) with rule based approaches to develop a sentiment analysis on Internet movie reviews database(IMDB). In both models SVM has give better accuracy than other machine learning algorithms. While using machine learning algorithm, one or more feature transformation mechanism can be used to make the data easy to be processed by the classifier algorithm. These feature transformation mechanisms include TF-IDF [28][36], N-gram [28][35][36], Part of speech tag [35][36], stemming [28].

3.1.4 Deep Neural Networks Based Sentimental Analysis

Deep learning approaches are part of machine learning which refers to deep neural networks [29]. Deep neural networks consist of many networks such as CNN (Convo-

lutional Neural Network), DBN (Deep Belief Neural Network), RNN (Recurrent Neural Network), Recursive Neural Network, Long Short-term Memory (LSTM), Bidirectional Long Short-Term Memory (BI-LSTM), RNDM (Recursive Neural Deep Model), and RNTN (Recursive Neural Tensor Network). However, CNN and RNNs are the most widely used deep learning models for sentiment representation [42].

Deep learning algorithms are influential because they have shown better performance than SVM and normal networks in many NLP applications including sentiment analysis due to more hidden layers [30]; their capability to provide training in both supervised/unsupervised and semi-supervised ways, and their ability to carry out automatic feature extraction without human intervention which saves time [30].

Deep Neural Networks are widely used in text generation, vector representation, word representation, estimation, sentence-classification, sentence-modeling and applications like speech recognition, computer vision and NLP applications [29][30].

Many research works [30][31][32]][33][34][42] are based on sentiment analysis for sentence, micro blog, visual, textual-visual sentiment analysis, and document estimation. Huang, Qiongxia, et.al., [30] develop a sentiment analysis that combines one-layer CNN directly stacked on two-layer LSTM with word embedding which outperforms the existing CNN-LSTM model [30]. The other work on [31] build a CNN-LSTM model with word embedding which perform better than LSTM, SVM, and CNN models [31].

Dos Santos, C. and Gatti, M. [38], and Severyn, A. and Moschitti, A. [32] develop Character to Sentence CNN (CharSCNN) for short texts, and CNN with word embedding for both phrase and message level tasks respectively. The first model use movie reviews from Stanford Sentiment Tree bank (SSTb) and Twitter posts from Stanford Twitter Sentiment corpus (STS) [38] while the second use a SemEval dataset [32].

Ruangkanokmas, at al. [42] develop a semi-supervised Deep Belief Networks (DBN) with feature selection (chi-squared) for sentiment classification using movie, KIT, ELE, BOO,

and DVD datasets. The developed model (DBNFS) performs better for three datasets except KIT and DVD datasets than TSVM, PIV, DB and HDBN [42].

There are also some state of the art researches on Arabic language sentiment Alayba, Abdulaziz M., et.al., [34] develop a combined CNN and LSTM with word embedding using 2026 main Arabic Health service dataset(AHS) and 1732 sub-AHS. The CNN and LSTM models show 94.24% accuracy for main AHS and 95.68% for sub-AHS datasets[34]. CNN with word embedding sentiment analysis model is developed by Dahou et.al., [33] using reviews of 3.4-billion-word corpus from the collected 10 billion words of web-crawled corpus. The Arabic sentiment analysis provide 86.7%, 89.6%, and 78.3% for CNN-balanced, CNN un-balanced and linear SVM respectively. From the results un balanced CNN model out performs over the Linear SVM approach by 11.3% in terms of accuracy [33].

3.2 Sentiment Analysis for Amharic Language

3.2.1 Lexicon Based Sentiment for Amharic Language

Selama G/Meskel [3] has developed a sentiment mining model for opinionated Amharic texts. In this research, the author used a term counting approach to determine the sentiment of comments by checking each word from the prepared lexicon. The author developed 411 positives, 544 negative general, domain-specific lexicon terms and contextual valence shifter terms.

In the paper, each positive lexicons are tagged with +2 value and each negative lexicons are tagged with -2 value but the final polarity of the review is determined by considering the existence of valance shifter, intensifiers, and negation terms.

Even if the paper prepares opinion terms, the prepared opinion terms are not sufficient because inflected Amharic sentiment words are not handled in this research work. Be-

sides, the paper only considers textual labeled lexicons for determining the polarity of reviews. Emojis which are increasingly used in social media contents to express one's feeling have not been handled.

The other work on the Amharic language is a feature level sentiment analysis by Tulu Tilahun entitled "Linguistic Localization of Opinion Mining from Amharic Blogs" [2]. The Author collected 484 comments from a hotel, university, and hospital as 578 negative and 423 positive words.

Tulu extracted features using some rules and determine the polarity of the review by considering the polarity of adjacent words of the feature word. The effectiveness of the system depends on the existence of valance shifters and the existence of adjacent left and right adjective features.

Yosef Arega [5] uses Posts from facebook page of DireTube, EthioTube, VOA Amharic, mereja.com, and Yehabesha. The model developed by Yosef uses Rules and Lexicons used by Selama G/Meskel [3]. TF-IDF and Latent Dirichlet Allocation approaches are combined for the trend detection system. The model gives different results depending on the topic selected and the existence of valance shifter terms.

All works consider only words for determining the polarity of the review. However, Emojis are increasingly used in social media content to express one's feelings on a certain event. Hence, it is necessary to develop a system that takes Emojis for labeling purpose. The work by Tulu Tilahun used adjectives only as sentiment words to determine the opinions of the review sentences, but sentiment words are not only adjectives but also adverbs, verbs, and nouns. In addition to this, the sentiment words are not sufficient for determining sentiment of a comment.

3.2.2 Machine Learning Based Sentiment Analysis

A Machine Learning approach to multi-scale sentiment analysis of Amharic online Posts is developed by Wondwossen Philemon and Wondwossen Mulugata using a supervised machine learning approach [1]. The authors collected 608 social media product and marketing news from Facebook, Twitter, DireTube and Ethiopian reporter websites. The developed model has an accuracy of 43.6%, 44.3% and 39.5% for uni-gram, bi-gram, and hybrid features respectively. The developed model has performance limitation due to less number of training data.

CNN based Amharic sentiment analysis [4] is developed by Yeshiwas Getachew and Abebe Alemu. Yeshiwas Getachew and Abebe Alemu collected 1602 reviews From Fana Broadcasting Facebook page using Facebook API. The model produces different results depending on the amount of test train-data split ratio and model configuration. For instance, for three layers, 4000 input dimension and 70%:30% split ratio the model gives 95.5%, 90.5% training and validation score respectively. For 4 layers, 1052 input dimension and 70%:30% split ratio, the model produces 84.05 and 79.89 training and validation score respectively.

Both researches [1][4] use stemming as part of preprocessing after tokenization, stop word removal, and normalization. However, literature reviews for other languages showed that stemming has no positive impact on the performance of sentiment analysis model. Hence, it is necessary to investigate the effect of adding stemming as part of preprocessing on Amharic sentiment analysis. CNN and RNN networks are state of the art DNN approaches for various NLP applications. However, CNN is well known for problems that require extraction of features while LSTM is a popular approach that captures a sequential data. LSTM are designed to capture long short term sequence dependency with its memory [49]. The corpus used by both models is annotated by humans which takes much developing time and results in disagreement between annotators.

3.3 Effect of Stemming on Sentiment Analysis

Stemming is one of the techniques used to improve NLP in general and sentiment analysis in particular [21]. Stemming is a process of conflating variants into their stem or root form. Stemming is an important technique in many NLP applications like data mining and information retrieval by reducing variant word forms to a common root to increases retrieval effectiveness but unimportant for others [14].

Some researches use stemming as one of the preprocessing steps while others argue that stemming does not affect text analysis. Many research are done on stemming to show whether it positively or negatively affect NLP applications.

Romero Llombart, Ò [28] showed that applying data transformation approaches such as deleting stop words punctuations, lemmatizing has a positive impact on tweeter dataset. For instance Recurrent Neural network applying stemming or lemmatizing, removing stop words and punctuations give (84% accuracy and F score). However, RNN without any transformation results (83% accuracy and 84% F score). For the case of Random Forest, applying count stemming, removing stop words and punctuations give a better result (81% accuracy and F score) than any other transformations such as TF-IDF stem, removing stop words punctuation that results 78%, 79% accuracy and F score. TF-IDF lemmatize, removing stop words punctuation gives 77%, 78% accuracy and F score, count vectorization and removing stop words punctuation results 78%, 79% accuracy and F score respectively, and TF-IDF with out any transformation gives 74%, 75% accuracy and F score respectively [28].

While applying Bi-gram word representation on the same twitter dataset has negative impact for RNN. However, Random Forest (RF) involving count stemming, removing stop words punctuation, and NB applying Count lemmanize, remove stop words punctuations out performs than any other transformations. For movie reviews dataset and Bi-gram word representation, incorporating stemming, lemmatize, and removing stop

words punctuations negatively affect the performance of MLP (Multilayer perceptron), NB,RF and SVM.

Shereena M. and Mazlina M.[53] has examined effect of noise elimination and stemming in sentiment analysis such as data with tokenization, stop word removal only, addition of normalization only, and addition of stemming using (Othman) and Fatimah Algorithm. From the results all datasets obtain small improvements after applying different types of pre-processing [53]. SVM classifier with data tokenization, stop word removal, normalization and stemming using Fatimah performs better than Othman algorithm. It also achieves the best performance for the Malay documents classification [53].

Hidayatullah, A.F. [10] showed that stemming reduces the performance from 88.84% to 87.72% when Naive Bayes is combined with TF-IDF, and from 92.19% to 91.96% when Naive Bayes and term frequency are applied for Indonesian tweet sentiment analysis.

Wahbeh et.al., [14] have made an empirical study on the effect of Arabic text classification by using stemming as part of pre-processing steps. To show the effect of stemming, the authors conducted an experiment using SVM and two test models. The results show that applying stemming negatively affects the accuracy of the model. The accuracy of the model drops from 87.79% and 88.54% to 84.49% and 86.35% [14]. This research paper tries to investigate the effect of stemming on Amharic sentiment analysis. The preprocessing applied may have positive or negative impact depending on the type of dataset used and type of language it has [28] and stemming algorithm [14][53]. So it is necessary to investigate impact of stemming as part of preprocessing for Amharic sentiment analysis.

Chapter 4

Proposed Methodology

The proposed is a hybrid of two known approaches (Lexicon and deep neural network based) for Amharic sentiment analysis to achieve objectives discussed in the paper. Figure 4·1 shows the entire process of the proposed approach.

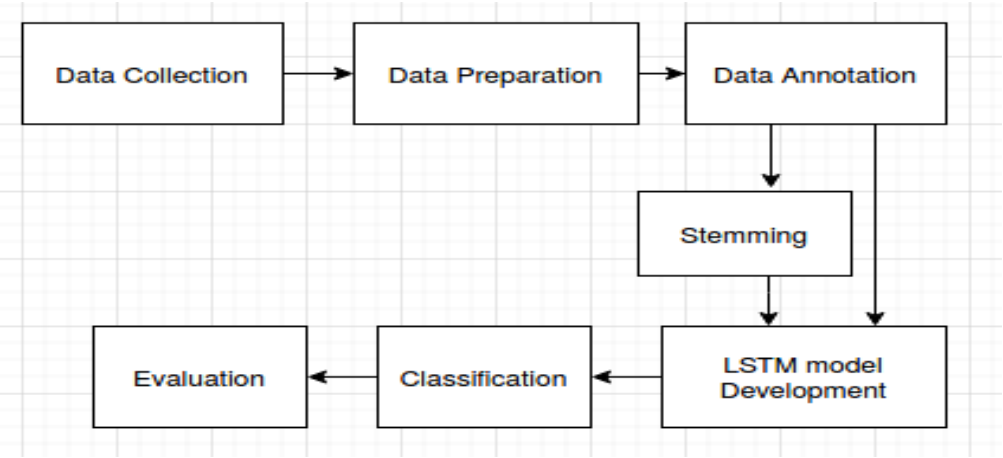

Figure 4·1: Proposed Approach.

Each steps in the proposed approaches is discussed in the sections below.

4.1 Data Collection

To develop a sentiment analysis, data collection is the first and fundamental step that encompasses collecting user-generated contents from blogs or social media like Facebook, YouTube, Twitter, and others. Scraping is the process of collecting user-generated content from social media and blogs. This research paper uses YouTube comment scarper for

collecting raw data in the form of unstructured data. The link below contains collected comments, and labeled lexicons and Emoji
https://drive.google.com/file/d/1UcY6oKwmLwxW9QB88xbz22jRYvhH9uWU/view?usp=sharing.

4.2 Data Preparation (Preprocessing)

Data preparation is a major step in data analysis that encompasses many techniques mainly data transformation, data cleaning, and data reduction. This step helps to make the collected data clean and suitable for analysis. In data preparation, non-textual contents except Emoji and contents that are unimportant for the analysis are first identified and eliminated.
The following tasks are performed to clean the collected data. Figure 4·2 shows the order in which they are applied on the data.

- Eliminating non-textual, non-Amharic contents and symbols like hashtags, URLs, links and others[21].
- Removing all numbers and punctuations like question marks(?), exclamation marks(!) and others because question and admiration do not affect the sentiment of a certain sentence [28][21].
- Spell correction.
- A consistent tokenization rule should be applied to split sentences into words and treat each punctuation as a separate token [9]. A word tokenizer from NLTK is used in this thesis.

4.2.1 Normalization

Normalization is one of the steps used to get clean data from unstructured textual data. Different characters that have the same sound but written in different forms like **ሀ**, **ሐ** or **ኀ** should be changed to the same form. There are two kinds of Normalization. The first one is character level normalization, and the second one is word-level normalization. Both word and character level normalization are used in this thesis. Normalized characters and words are listed in Appendix A.

4.2.2 Stop Word Removal

Removing stop words helps to reduce the dimensionality of a term space. The most common words in text documents such as articles, prepositions and pronouns are treated as stop words. These words do not give the meaning of the documents.

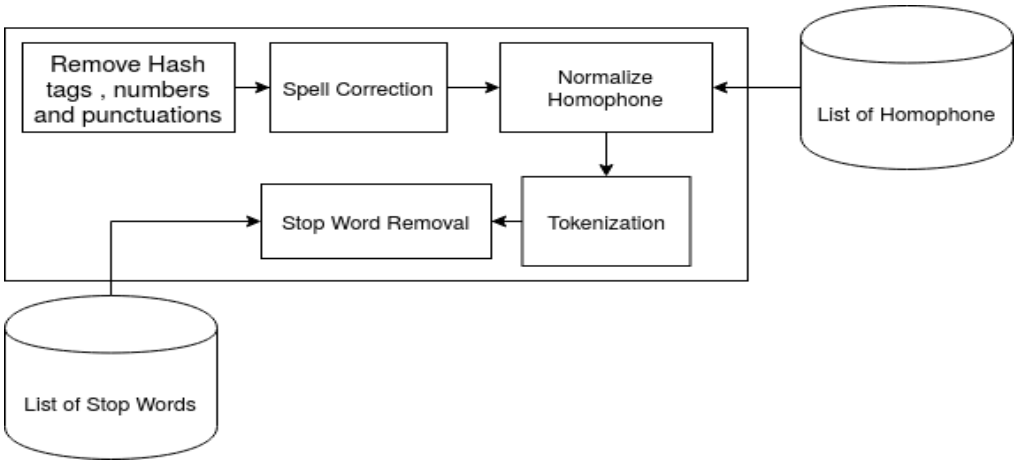

Figure 4·2: Preprocessing Steps.

አረ በዶክተር አብይ በለማ መገርሳ እነሱ ምን ይሉሃል የለፉት...ለፋሁ ትላለህ ጉረኛ ..ወሬኛ .....ኢትዮጵያ ኑሪ!!!!!!!!!!!!.....and **እንደጠበኩት አይደለም**#\uff# are sample unstructured data because they contain punctuations, un normalized and Unicode Characters, and not tokenized. Hence, it is necessary to make the data clean for further processing. After the above preprocessing steps applied on it, the sentences will be ['**እረ**', '**በዶክተር**', '**አብይ**', '**በለማ**',

'መገርሳ', 'እነሱ', 'ምን', 'ይሉሀል', 'የለፉት', 'ለፋሁ', 'ትላለህ', 'ጉረኛ', 'ወሬኛ', 'ኢትዮጵያ', 'ኑሪ'] and ['እንደጠበኩት', 'አይደለም']

4.3 Data Annotation

Data annotation is a process labeling any type of data. Textual data annotation for sentiment analysis is a categorization of user generated textual content as positive, negative and neutral. In this work, we have annotated the processed dataset using labeled lexicons and Emojis to show the effect of using both labeled Emojis and lexicons over using only labeled lexicon only on automatic labeling system. Figure 4·3 shows sample data annotation using both labeled lexicon and Emoji.

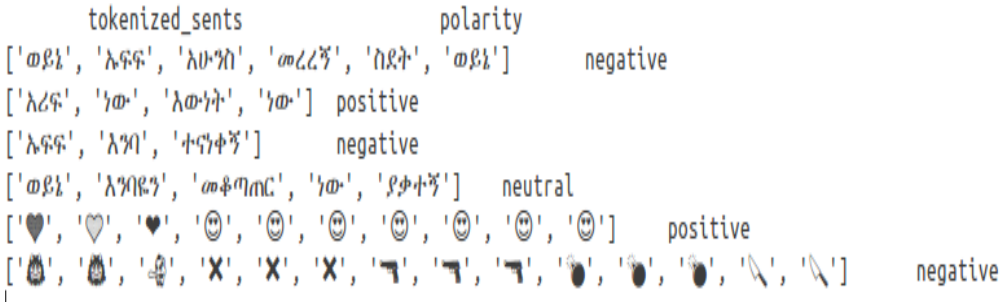

Figure 4·3: Sample Annotated Data.

4.4 Stemming

Stemming is a process of converting different forms of a word to its root form. We have trained a neural network with and without stemming to see its impact on Amharic sentiment analysis. In some natural language processing applications, additional data preparation approaches like stemming, lemmatizing or part of speech tagging may also be done [28]. Stemming is part of preprocessing. However, this paper implement stemming after data annotation because the labeled lexicons are in un stemmed form. If

the stemming is performed before data annotation, the morphological operation implemented makes the automatic labeling difficult. Sentences below show before and after stemming is applied. For instance, **ወይኔ የኔ ሚስኪኖች ባላህ አንጀቴን በሉኝ ወላሂ ኡፍ ችግር አይንህ ይጥፋ , ማጣት ስንቱን ያደርጋል**, and **ማጣት ደጅ ያስጠናል** are sample sentences before applying stemming. After stemming is applied using HornMorpho 2.5, the sentences will be: **ወይን እኔ ሚስኪና ባላ አንጀት አለ ወላሂ ኡፍ ችግር አይን ጠፋ, አጣ ስንት አደረገ**, and **አጣ ደጅ አስጠና** respectively. This research paper investigates the effect of using stemming as part of preprocessing on Amharic sentiment analysis. The stemmed and non-stemmed comments are available at https://drive.google.com/file/d/1UcY6oKwmLwxW9QB88xbz22jRYvhH9uWU/view?usp=sharing.

4.5 Neural Network Development Steps

Figure 4·4 shows the steps followed to develop deep neural network by taking the annotated data as input.

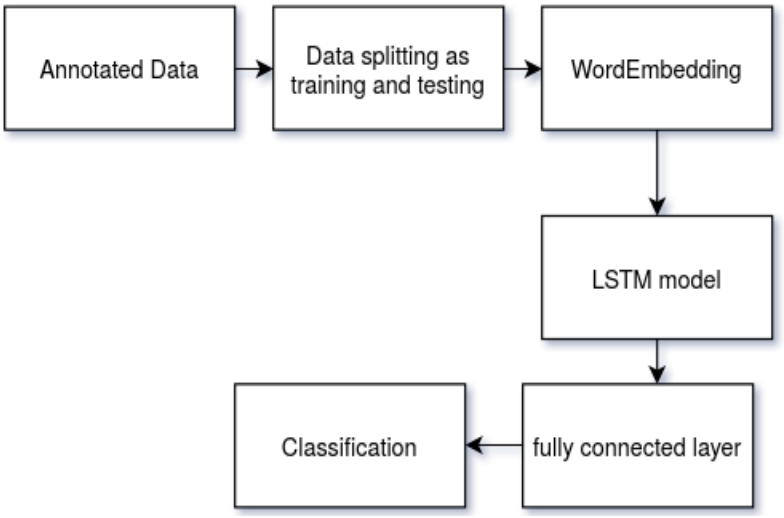

Figure 4·4: LSTM Network Development steps.

The steps followed for developing LSTM model are discussed below.

4.5.1 Data Splitting

Data splitting is one of the crucial steps in developing and testing a neural network classifier model. In this step, the annotated data is split into training and testing. In this work, we have used 90% for training and 10% for testing the neural network.

4.5.2 Word Embedding

After converting the unstructured data to a form suitable to analysis, it is necessary to represent the words in the sentence to a vector form to compute similarity, analogy and other operations using word2vec. The aim of word embedding is to map each words in the word dictionary to a lower dimensional vectors. The embedding layer turn the input to nxm dimensional vector where n is the longest sentence in the corpus and m is the embedding dimension(dimension of the word vector).

4.5.3 LSTM Model

LSTM is a special type of deep neural network that is proved to be extremely effective in capturing long and short-term dependencies in a sequential data [42]. This layer takes vectors of word2vec as input from the dense layer. LSTM layer has a memory unit that helps in adding and removing previous information.

4.5.4 Fully Connected Layer

The outputs of an LSTM layer are merged and combined into a single matrix and passed to the fully connected layer. The fully connected layer is a full connection of previous layers. This layer converts the array into a single output in the range between 0 and 1. The fully connected layer classifies the vectors as positive, negative and neutral. The output layer has 3 neurons which represent the three sentiment classes such as

positive, negative and neutral. This layer uses softmax activation function because if the activation values of each neuron summed it will be one.

4.6 Evaluation

After classification has finished, the performance of the classifier is evaluated using different performance metrics like recall, precision, F score and accuracy. Training and validation accuracies and losses are also used for choosing appropriate epoch number to avoid occurrence of overfitting.

Chapter 5

Experiments

To answer the research questions, we have conducted different experiments. Generally, these experiments are grouped into three categories. The first group of experiments aims to show the effect of labeling a user-generated comments using only labeled lexicons and combining labeled lexicons and Emojis. To evaluate the effect, we used the dataset labeled using the two approaches,using lexicons and combining these lexicons with labeled Emojis, as an input for two sentiment classifiers, LSTM and Naive Bayes. These experiments answer RQ1: **Can Emojis be used to enhance lexicon based automatic comment labeling for Amharic sentiment analysis?**

The second group of experiments aims to identify the impact of stemming on Amharic sentiment analysis on both LSTM and Naïve Bayes sentiment classifier model. These experiments are conducted to answer RQ2: **What is the effect of stemming on the Amharic language sentiment analysis?**. The third group of experiments compare an automatic annotation system with a human annotation system to identify the gaps on our automatic labeling system. These Experiments answer RQ3: **How the automatically labeled data (using Emojis and lexicons) is consistent with human labeled data?**

5.1 Dataset Description

To answer the RQs, we created a dataset containing comments written in Amharic language. The dataset contains 9,138 comments collected from YouTube using YouTube

online comment scraper. The dataset is a collection of comments that are written using text and Emoji, text-only, and Emoji only. Table 5.1 provide information about the dataset before and after applying preprocessing on the dataset. The information includes: the total number of sentences and words, mean and maximum comment length, and number of comments that do and don't contain Emoji.

Before and After pre-processing	Total number of sentences	Total number of words	Mean length	Maximum length	Number of comments that contain Emoji	Number of comments with no Emoji
Before	9138	94,751 words	10.368 words	437 words	1809	7329
After	9138	83,203 words	9.105 words	433 words	1809	7329

Table 5.1: Dataset characteristics

5.2 Building Sentiment Lexicons and Emojis

There are some works that collect Amharic sentiment lexicons [3][2]. Gebremeskel [3] has collected about 950 lexicon terms while Tulu [2] has collected about 578 negative and 423 positive opinion words. It is necessary to include more Amharic lexicons to increase the accuracy of the sentiment analysis. Hence, for this research, in additon to what is collected by others, we collected 406 positive and 595 negative lexicons. For this research, we used a total of 1173 negative, 829 positive, 8 invertors and 13 intensifier lexicons. These lexicons are a set of both domain specific and general-purpose lexicons. Emojis, which are a subset of unicode characters, are stored and represented as any unicode characters [41]. In the beginning, about 176 Emojis were used by Japanese mobile operators [41]. Emojis can be used as a feature or as a label [39][40][41]. To use Emojis for labeling comments, it is necessary to build positive and negative Emoji lexicons.This research paper has collected a list of negative and positive Emojis for comment labeling

purpose from the literature and web [44,45] by comparing their positivity, negativity and neutrality score.

The labeled lexicons and Emojis used for automatic comment labeling are available at https://drive.google.com/file/d/1UcY6oKwmLwxW9QB88xbz22jRYvhH9uWU/view?usp=sharing.

5.3 Evaluation Metrics

In order to measure the performance of a model, there are many evaluation metrics such as accuracy, precision, recall and F score [1]. The most commonly used performance measures in sentiment analysis are precision, recall, accuracy and F score [1]. These measures are based on

TP(True Positive): The number of instances that are positive and are correctly predicted as positive.

FP(False Positive): The number of instances that are negative but are incorrectly predicted as positive.

FN(False Negative): The number of instances that are positive but are incorrectly predicted as negative.

TN(True Negative): The number instances that are negative and truly predicted as negative.

Precision(P) is the ratio of the number of items labeled as a particular desird class, true positives(TP) to a total number of items labeled as that class [25][46].

$$p = \frac{TP}{TP + FP} \tag{5.1}$$

Recall(R) is the number of true positives divided by total number of items that are known to belong to that class [25][46].

$$R = \frac{TP}{TP + FN} \tag{5.2}$$

F score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and negatives into account [25][46].

$$Fscore = \frac{2 * Recall * Precision}{Recall + Precision} \tag{5.3}$$

F score is a better and well-known measure to use if we need a balance between precision and Recall. F score is usually more useful than accuracy, especially if you have an uneven class distribution [13].

Accuracy The other performance measure of a model is accuracy. Accuracy measures how much accurately the model learns to classify the data [25][34][46].

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.4}$$

The overall precision, recall and F score of a classifier is calculated by using a weighted average called a macro-average.

Macro-average values are used to know how the system performs overall across the sets of data.

$$Macro{-}average\_recall = \frac{R1 + R2 + R3}{3} \tag{5.5}$$

$$Macro{-}average\_precision = \frac{P1 + P2 + P3}{3} \tag{5.6}$$

R1, R2 and R3 are recall values of the three sentiment classes negative, neutral and positive respectively. P1, P2 and P3 are the precision values of the three sentiment classes respectively. The macro-average method can be used when we want to know how the

system performs overall across the sets of data. In this paper, we use macro-average precision, recall, and F score as a weighted average.

5.4 Experimental Setup

This paper uses one personal computer for all experiments. Table 5.2 shows the hardware and software specification of the machine used in all experiments.

Manufacturer	Hewlett-Packard
Model	HP ProBook 6470b
Processor	IntelCore™i5-3320MCPU @2.60GHz,2601MHz,2 cores,4 logical processor(s)
Memory	4GB
Operating System	Ubuntu 16.04 LTS

Table 5.2: Table Hardware and Software Specification of the machine

Model Configuration: In addition to hardware and software specifications, we need to configure the LSTM network for developing sentiment analysis model. In the experiment, LSTM network is configured with the following parameters: one LSTM layer with 8 neurons, 8 epoch, 14 batch, 8 vocab_size(Embedding_size), 'adam' optimizer, 7 seed, 90% train_size, and 10% test_size.

Overfitting and underfitting are key problems of deep neural networks. Overfitting occurs when the model learn training dataset so well, which makes noises of the training data to be memorized. Hence, it cannot predict an output for an input that it has never seen before [11]. Overfitting is a phenomenon that occurred on small dataset than large dataset [11][43]. Our dataset is small in comparison to what deep neural networks require, so it is necessary to use techniques such as early stopping and dropout to avoid overfitting.

Early Stopping Configuration:- Early stopping is one of the mechanisms used to reduce

the occurrence of overfitting [11][43][48]. Early stopping is a mechanism to stop training as soon as performance of a model stop improving [11][43]. Early stopping parameters are configured as follows: patience=5, monitor=val_loss, mode=min, and baseline=0.3.

Dropout: Dropout helps to prevent overfitting by randomly dropping some of the units and their connection from the neural network during training [11][43]. In this paper the optimal choice of dropout is between 0.5 and 1 specifically 0.5 and 0.7.

In order to choose an appropriate epoch number, this paper has used training versus validation accuracy plot of the model. To identify occurrence of overfitting from a plot the following conditions will occur.

1. When both training and validation loss simultaneously decreases. However, at some point of time validation loss increases while training loss decreases.

2. When both training and validation accuracy starts increasing and at some point when validation accuracy increases over training accuracy.

Figure 5·1 shows sample training vs validation accuracy and loss plot of the model using our dataset when both lexicons and Emojis are used for labeling.

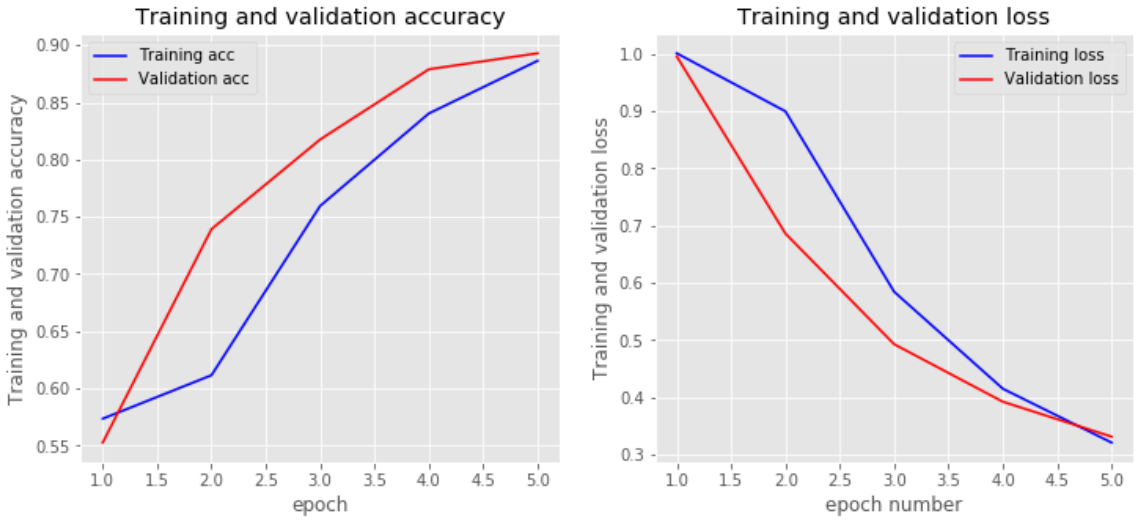

Figure 5·1: Training vs Validation accuracy and loss.

From the loss plot of Figure 5·1, if the model is trained beyond epoch 5, the validation

loss increases while training loss decreases. This is one of the condition to say the model is overfitted. Hence, our epoch choice will be five to see the effect of overfitting.
This paper choose two epochs for the LSTM model based on the preliminary experiments. The two epochs are used (i) to see the effect of using Emojis for labeling (with and without Emoji) choose eight epochs, and; (ii) to see the effect of adding stemming as part of preprocessing on Amharic sentiment analysis using seven epochs.

5.5 Development Tools and packages

Python: Python is a high-level programming language that is processed by a python interpreter to produce results. There are many features that makes python the best choice of language for machine learning and artificial intelligence.

- Python is a high-level language that is easy to learn, read, and maintain
- Short development time in comparison to other programming languages like java, C++ and Ruby.
- There are plenty of libraries in python that makes our task easier, for example, Numpy is a library for python that can solve scientific computation easily.
- It is interactive (has a terminal for debugging and testing) and portable (runs on a variety of hardware platforms).

Pandas: Pandas is an open-source library that provides high performance, easy to use data structure and data analysis tools for python programming language [26]. Pandas is used to read CSV files and perform different operations on the CSV files.
Keras: Keras is a library for developing deep neural networks in python that can run using TensorFlow or Theano as a back-end to preprocess data, to create, evaluate, optimize, fit and test a model [25].

TensorFlow: TensorFlow is an end-to-end open-source platform that has flexible tools and resources for machine learning. Many states of the art NLP applications are developed using TensorFlow as a backend. Popular organizations like Google, Intel, and others also use TensorFlow to develop systems [25].

Natural Language Toolkit(NLTK): NLTK is a platform that is used for building Python programs to work with human language data. NLTK is used for many tasks like tokenization, lemmatization, stemming, parsing and POS tagging. NLTK is developed especially for languages that are rich in resources.

Scikit-Learn: Scikit-learn is a library for python machine learning library. Scikit-learn contains simple and efficient tools for data mining and data analysis algorithms for both supervised and unsupervised problems [25]. Scikit-learn is used to evaluate deep neural classifiers by calculating a confusion matrix which is a table that is often used to describe the performance of a classification model [25].

HornMorpho: HornMorpho is part of the L3 project at Indiana University. It is a python program that is used to analyze Amharic, Oromigna and Tigrinya words into their meaningful parts. HornMorpho gives a representation of the words grammatical structure when the root or stem is given [24]. HornMorpho is free software that can be freely distributed or modified under the General Public License. To develop morphological analysis, HornMorpho uses 227,984 Tigrinya and 397,352 Amharic words [24].

5.6 Results

5.6.1 Effect of using Emojis for labeling

The amount of dataset used in sentiment analysis affects the final result. Prior to answering RQ1: **Can Emojis be used to enhance lexicon based automatic comment labeling for Amharic sentiment analysis?**, we conducted an exploratory experiment

to see the labeled Emojis with lexicons on the dataset, number of labeled comments in each sentiment class, and Amharic sentiment classifier performance. Table 5.3 shows how using labeled Emojis for comment labeling affect the distribution of a datasets to be considered.

Type of comment (class of comment)	Amount of comments in the class
Comments written only with text	7329
Comments written with text and Emoji	905
Comments written with Emojis only	904
Total	9138

Table 5.3: Effect of Emojis on data set distribution

From the total number of collected comments, 19.79% of the comments use Emojis. This shows that a significant amount of comments use Emojis to express their opinion.
Emojis are considered to be reliable indicators of sentiment, they either used to automatically generate more training data or to enhance sentiment classification [41]. The use of labeled Emojis for labeling user-generated data also affects the number of comments classified as positive, negative and neutral. A use of Emoji enhance the expressibility of a text message. For instance, the word "Addis Ababa" cannot be labeled as positive or negative but "Addis Ababa ❤️❤️" is classified as positive because it expresses positive feeling towards the city. In the same way the texts like, **አስቱየ ምን ላርግሽ, ፍቅሬን ታስታውሣኝ አለህ, እንዲች እንዲች, ላዛኛ** are difficult to determine their sentiment class, **አስቱየ ምን ላርግሽ 💕 💕 💕 💕 💕 💗 💗 💗 💗, ፍቅሬን ታስታውሣኝ አለህ ❤ ❤ ❤ ❤ , እንዲች እንዲች ❤**", **ላዛኛ** 😁 can be labeled as positive. Comments that contain only Emojis like "💔 💔 💔 💔 💔 💔 💔 💔 💔 💔 💔 💔 💔 💔" and ♥ 🔥 💯 cannot be labeled unless the labeling system combine list of positive and negative Emoji with labeled lexicon words. Figure 5·2 shows the amount of labeled comments in each sentiment class when we consider labeled lexicons with Emojis and labeled lexicons only.

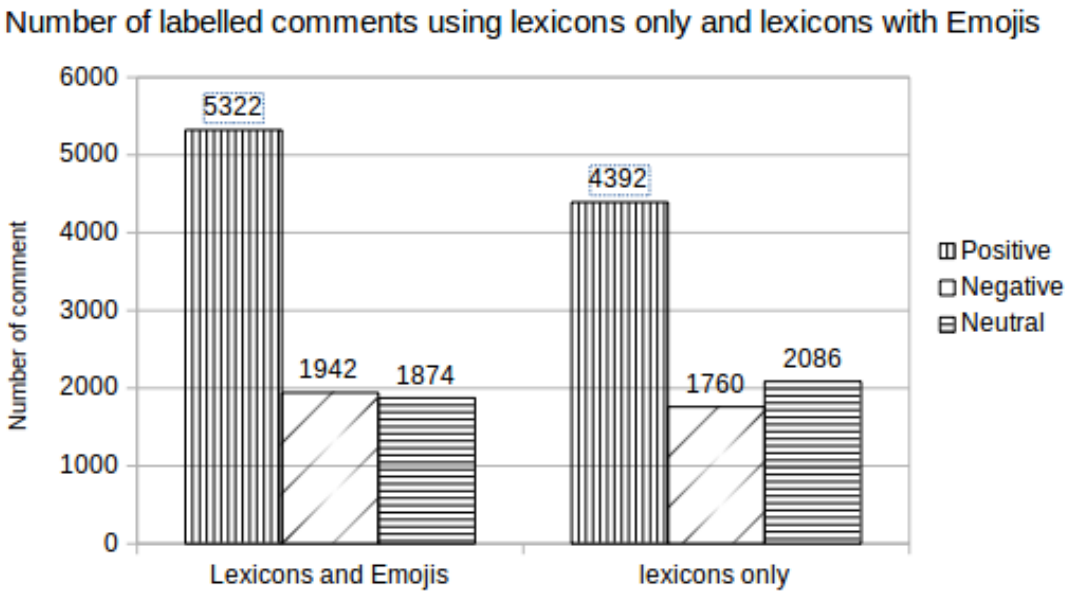

Figure 5·2: Amount of labeled comments in each sentiment class.

Figure 5·2 shows the amount of labeled comments as positive, negative and neutral from the total dataset when we consider both labeled lexicons and Emojis, and only labeled lexicons. The result show that we can label a total 9138 comments when we use both labeled lexicons and Emoji, However, a total of 8234 comments are labeled when we use only labeled lexicons. The amount of labeled comments using lexicon approach is lower by 904 from the combined lexicon and Emoji based approach. The remaining 904 comments will be unlabeled unless we incorporate labeled Emoji on our comment labeling system. This show that Emoji play significant role in automatic labeling system. Recent researches focus on not only imbalanced distribution between classes but also a difficulty in embedded nature of the data due to their relevance [54, 55]. This results a difficulty for the learning algorithm because it will be biased to the majority class. There are three methods to handle imbalanced data algorithm level which focus on modify existing learning algorithms to alleviate the bias towards majority objects and make them to adapt with skewed distributions [54, 55]. The second is data level which focus on modifying the data either by generating new data for minority dataset

with replacement(up sampling) or removing some data from the majority dataset (down sampling). However, this often leads to removal of important samples or introduction of meaningless new objects [54]. The last approach, which is gaining increasing popularity is a hybrid level that takes advantages of the above two approaches [54, 55].

In this research, their are two imbalanced distributions the first is with Emoji dataset which is a majority class and without Emoji which is a minority dataset. The second is between the positive, negative and neutral classes in the datasets itself. To overcome dataset imbalance the paper choose random oversampling of the minority class from 8234 to 9138. Under sampling of majority class is not proper choice because further removing from a small dataset is dangerous [55]. This research aims to investigate the effect of considering labeled Emoji and lexicons for labeling of comments on the performance of deep neural network specifically Long Short-Term Memory using 9138 sample size for both approaches. Table 5.4 shows the results obtained with and without Emoji for comment labeling.

Performance measures	Classes					
	Without Emoji			With Emoji		
	Negative	Neutral	Positive	Negative	Neutral	Positive
Precision	74.8	77.4	95.12	76.04	81.03	92.7
Recall	80.9	69.39	96.89	71.56	77.4	96.21
F Score	77.7	73.34	96.00	73.73	79.43	94.42
Macro-Precision	82.43			83.25		
Macro-Recall	82.39			81.75		
Macro-F score	82.34			82.52		
Accuracy	86.43			86.98		

Table 5.4: Comparison of LSTM performance with and without Emoji as a feature

As we can see the result from Table 5.4, incorporating Emojis for comment labeling with lexicons increases accuracy of LSTM model from 86.43% to 86.98%, and F score from 82.34% to 82.52%. However, from the table, precision and recall of positive, and recall of negative sentiment class without Emoji dataset is higher than with Emoji. This may

be due to the random oversampling with replacement of without Emoji dataset from 8234 to 9138, because it results a more disproportion between the positive, negative and neutral sentiment classes or introduce a noise.

This paper has also made an exploratory experiment to determine the performance of Multinomial Naive Bayes (MNB) with and without Emojis. Results on Table 5.5 shows the performance of MNB with and without Emoji. All the precision, recall and F score in the table are macro-average values.

MNB classes	Performance Measure (Macro-average)	10% test, 90% training data		20% test, 80% training data	
		Without Emoji	With Emoji	Without Emoji	With Emoji
Uni-gram	Precision	92.0	92.0	90.0	87.0
	Recall	89.0	89.0	87.0	82.0
	F score	90.0	90.0	88.0	84.0
	Accuracy	91.14	91.0	89.50	87.02
Bi-gram	Precision	93.0	92.0	90.0	88.0
	Recall	91.0	89.0	89.0	85.0
	F score	92.0	90.0	89.0	86.0
	Accuracy	92.96	92.0	90.59	89.3
Tri-gram	Precision	92.0	92.0	89.0	88.0
	Recall	89.0	88.0	87.0	84.0
	F score	90.0	90.0	87.0	86.0
	Accuracy	91.74	91.67	89.13	88.11

Table 5.5: Comparison of MNB with and without Emoji

Results on Table 5.5 show that adding Emojis reduce accuracy of MNB by 0.14%, 0.96% and 0.07% when using Uni-gram, Bi-gram and Tri-gram respectively with 10% test and 90% training data. This reduction is due to the nature of Naive Bayes which performs better, even when used with small training data [6][46]. However, in Naive Bayes(NB), as the training data decreases and test data size increases both F score and Accuracy reduces significantly. This reduction is because the model cannot capture all attributes well with small amount of training data size during training. For instance, a Bi-gram Multinomial Naive Bayes(MNB) that combine both Emojis and lexicons as labeling have tested with different training and testing data size. The results are described as: (i) using

60% training and 40% test data size, the model gives 81.0, 85.21 F score and accuracy value respectively. (ii) using the same MNB implementation with 40% training and 60% test data size, the model gives 73.0, 79.78 F score and accuracy value respectively. This indicate a reduction of training data size by 20% reduces the accuracy from 85.21 to 79.78 and the F score from 81.0 to 73.0.

Results on the Tables5.3 , and 5.4 answer RQ1: Can Emojis be used to enhance lexicon based automatic comment labeling for Amharic sentiment analysis?. The results of the experiments conducted to answer RQ1 show that Emojis can enhance lexicon based automatic comment labeling for Amharic sentiment analysis. By using Emojis, we were able to automatically label 904 more comments which would have been missed if one uses only lexicon (see Figure 5.2). The use of Emojis in two machine learning (ML) based automatic comment labeling approaches, LSTM and MNB, however, is inconclusive. For LSTM, the use of Emojis with lexicons have improved the accuracy and F score values of the automatic ML based labeling while for MNB values of the measures are lower than only lexicon. This could be due to nature of the algorithms used for labeling and the size of training data [6,46].

5.6.2 Effect of Stemming on the Amharic Sentiment Analysis

In this section, we investigate the effect of stemming on the performance of Amharic sentiment analysis. Table 5.6 shows examples of the root words, inflected forms of the root word and the corresponding stem while using using Hornmorpho 2.5 [24].

Root word	Inflected forms of the Root word	Stem of inflected words
ደረሰ	ይደርሳል ይድረስ ድረስ ደርሶ ሲደርስ ከደረሰ ተደረሰ	ደረሰ ደረሰ ደረሰ ደረሰ ደረሰ ደረሰ ተደረሰ
አደረሰ	አስደረሰ ያደርሳል ያስደርሳል	አስደረሰ አደረሰ አስደረሰ
ኃላፊ	ከየኃላፊዎቻቸው የኃላፊዎቻቸው ኃላፊዎቻቸው	ከየኃላፊዎቻቸው የኃላፊዎቻቸው ኃላፊዎቻቸው
ወሬ	ለወሬ በወሬ ከወሬ ወሬኛ ወሬኞች	ወር ወር ወር ወሬኛ ወሬኛ
አስፈለገ	የማያስፈልጋትስ አስፈላጊ የሚያስፈልጋትስ የሚያስፈልገው አስፈላጊው የማያስፈልገ	አስፈለገ አስፈላጊ አስፈለገ አስፈለገ አስፈላጊው አስፈለገ

Table 5.6: Inflected words and their stem forms of some root words

To show the effect of stemming, we conducted sentiment analysis using stemmed and unstemmed dataset without changing the polarity of unstemmed dataset. To compare results of Multinomial Naive Bayes (MNB) with stemming of this paper with previous works, we followed the same approach as [1]. Tables 5.7 and 5.8 show the performance of LSTM and MNB with and without stemming.

Performance measure	10% test and 90% training data	
	LSTM with stemming	LSTM without stemming
Precision	78.235	82.26
Recall	72.51	78.89
F score	70.76	78.89
Accuracy	75.93	82.36

Table 5.7: Performance of LSTM with and without stemming

Performance Measure	20% test data and 80% training data					
	Without Stemming			With Stemming		
	Uni-gram	Bi-gram	Tri-gram	Uni-gram	Bi-gram	Tri-gram
Precision	88.0	91.0	91.0	87.0	90.0	90.0
Recall	85.0	90.0	90.0	85.0	90.0	89.0
F Score	86.0	90.0	90.0	86.0	90.0	89.0
Accuracy	87.03	90.7	90.7	86.5	90.27	89.8

Table 5.8: Performance MNB with and without Stemming

Table 5.7 and 5.8 show that the accuracy of sentiment analysis with stemming is lower than without stemming by an average of 6.43% for LSTM and 0.53%, 0.43% and 0.9% for Uni-gram, Bi-gram and Tri-gram Multinomial Naïve Bayes models respectively. Stemming gives the same F score for uni-gram and Bi-gram implementation of MNB as without stemming. However, the F score of LSTM and Tri-gram implementation of MNB reduces from 78.89% to 70.76% and from 90.0% to 89.0% respectively.

Stemming reduces inflected words to their stem or root form that results in a reduced number of features or vocabulary size. Sometimes the stem of inflected words need not be identical to the morphological root of the word. For instance, the word "ሰኬት" can be inflected into "ሰኬተኛ", "ሰኬታሞች", "በሰኬት", "የሰኬት", "የተሳካ", "ሰኬታሞቹ" etc. In this example, all those words have the same root that is "ስክ". This shows less dependence between inflected words and the root word "ስክ". Merging inflected forms of words to a common stem form results in weak dependence between features.

LSTM network can capture long dependencies of features in a sentence by introducing a memory unit and gate mechanism. These memory units and gate mechanisms decide how to utilize and update the information kept in memory cell. However, the weak dependence of features in a sentence due to stemming makes the LSTM network not to capture the dependency between these features. This in-turn makes the LSTM not to

predict well for a new features. In conclusion, the experiments conducted to answer RQ2 show that stemming in the preprocessing step has a negative impact on sentiment analysis.

5.6.3 Annotation Consistency

To answer RQ3, we investigate alignment of comment labels that are annotated manually by humans and automatically using Emoji and lexicon. For the investigation, 3000 comments from the total dataset are annotated by two human annotators. We, then, computed the similarity of automatically annotated comments with those annotated by humans using overlap similarity. Overlap similarity is defined as the size of the intersection of two sets, divided by the size of the smaller of the two sets. Table 5.9 shows a sample comparison of automatic annotation and human annotation.

No	Comment	Automatic annotated polarity	Human annotated polarity	Similarity (1 for similar and 0 for different)
1	እባክዎ ውደድ ይጫኑ ከይቃርታ ጋ	Neutral	Neutral	1
2	እጅግ በጣም ደስ የሚል ስራ ነው	Positive	Positive	1
3	ኢትዮጵያዬ ቀና ምትይበት ቀን የደረሰ ይመስላል ተስፋ አለኝ	Positive	Positive	1
4	ደስ ይላል ግን አስለቀሰኝ	Negative	Negative	1
5	ጎበዝ እንባዬ አልቆም አለኝ	Positive	Negative	0
6	አገራችንን አማኑኤል ይጠብቅልን ሆድያባባል	positive	Negative	0
7	ምነው ልባችነን አንጠለጠላችሁት ውይ	Neutral	Positive	0
8	እር በሰው ነብስ ባትጫወቱ	Neutral	Negative	0
9	መቼ ነው ቁጥር ሶስት የሚወጣው	Neutral	Positive	0
10	አፌ ቁርጥ ይበል ልክ እንዲህ ንገራቸው	Positive	Positive	1
11	የነበረሽ ሁሉ እንዳልነበር ሲሆን 😭 😭 😭	Positive	Positive	1
12	ምን ላርግሽ ❤ ❤ ❤ ❤ ❤ ❤	Positive	Positive	1
13	እሙ ❤ ❤ ❤ ❤	Positive	Positive	1
14	😂 😂 😂 😂 😂 😂 አንተ አዝግ	Negative	Positive	0

Table 5.9: Sample Similarity matrix of Automatic annotation and Human annotation

From 3000 human-annotated corpora, 2700 comments are found to be similar to that of

automatically annotated comments giving an overlap similarity measure of **90.67%.**

$$O(A,H) = \frac{|A \cap H|}{min(|A|,|H|)} \tag{5.7}$$

O(A,H)=2700/3000=0.9067

Where

A= automatically annotated

H= human annotated

O(A,H)= Overlap Similarity

Most of the comments are expressed explicitly either using positive or negative lexicons to express a positive or negative feeling about an event. For instance, the comments "**ምድረ ደነዝ አውቃችሁ ሙታችሀል ይድፋችሁ**" and "**ፊልሙ አልተመረቀም እንኳን ሊለቀቅ ምን እሚሉት ውሸት**" express negative feeling using negative lexicons (**ደነዝ, ይድፋችሁ, አልተመረቀም** and **ውሸት**) respectively. So, these types of comments are correctly identified by both the manual and automatic approach. In the same way, taking a sentence "**የትኛውን ሙዚቃ እንደማደንቅ ግራ ይገባኛል በጣም ነው እምወድልህ ሁሉ ሙዚቃህን ወንድም ቀጥልበት በጣም ልዩ ነው ድምፅህ**" and "**ስወድክ ወደር የለኝም ብሰማው ብሰማው የማልሰለቸው**" forward a positive sentiment using a positive lexicons (**እምወድልህ, ቀጥልበት** and **ስወድክ , ወደር , የማልሰለቸው**) respectively.

Sometimes people express sentiments toward more than one aspects in a social media which is treated as multi-aspect classification problem. For instance, "**ፊልሙ ጥሩ ነው ግን ፊልሙ ላይ የአንግሊዘኛ ሳውንድ ትራክ መጠቀሙ ያስነውራል**", "**እሙ ያምራል ግን ዳይሬክት ያደረገ ሳምሶን ምን ነካው ለወንድ ተዘፍኖ ልጁን አላስገባውም**" and "**ሁሌም በባህላዊ መሳሪያ ከነ ባህልሽ ዱቅ ስትይ ውድድ ግን ዜማው ተመሳሰለ**" are multi-aspect sentiments. In the first sentence "**ፊልሙ ጥሩ ነው** is one aspect and "**ፊልሙ ላይ የአንግሊዘኛ ሳውንድ ትራክ መጠቀሙ ያስነውራል**" is another aspect these two aspects are joined by the invertor **ግን**. These multi-aspect comments are correctly labeled as negative by both automatic and manual approach.

Some comments are mislabeled by the automatic approach due to various reasons. To

better understand why these reviews are not correctly labeled by the Emoji and lexicon based automatic labeling, we closely looked at the data. Sometimes people use positive words to give a negative feeling or negative words to give positive attitude. For instance, "**የኔ ባሪያ**" uses negative word "**ባሪያ**" to express his love. These kinds of expressions are difficult to annotate using lexicon-based system because it cannot handle context of a sentence. So, the automatic approach makes such kind of comments to wrongly labeled. Inconsistencies between automatic and human annotation could occur due to the following reasons: i) absence of a lexicon in the lexicon database. ii) text annotation is subjective. A positive text for one person may be negative for the other, and iii) Contextual expression of a sentence. One can express a positive felling using a negative word and vice versa.

From the results, most comments that are not correctly classified are context-based expressions. For example, "**መቼ ነው ቁጥር ሶስት የሚወጣው**" express ones positive excitation to see the next episode of certain film or action, "**ነበዝ እንባዬ አልቆም አለኝ**" express ones negative feeling about certain event using positive lexicon. It is to mean that "I cannot control my tears" these and other kinds of expression are context-based expressions. There are incorrectly labeled multi-aspect comments that lower the automatic labeling system. For instance, "**ነፃነት ግን መች ነው ገፅ ባህሪውን የሚቀይረው**" and "**ከብዙ አመት በኋላ አየሁት ግን አይሰለችም**" are example of incorrectly labeled comments.

The automatic approach does not perfectly match with the human annotator. However, it is very close. We believe that this is a good result as the automatic approach has minimal overhead. Annotating comments by human require a significant amount of time and energy. It is also subjected to human errors because many words are ambiguous. The words become positive in some cases and negative in the other cases. This results in some incorrect labeling by humans.

Disagreement between annotators is typically part of the annotation process. A sentence

may express multiple emotions simultaneously [51]. This results in disagreement between annotators, and even an individual is not always consistent with her/himself. In most comments, annotators resolve their disagreement through discussion. Here, we have examined the inter-annotators agreement using the 300 incorrectly labeled dataset. Inter-annotators agreement can be used for analysis of the label sets, to improve the annotation process, and resolve annotation difficulties.

Table 5.10 shows the inter-annotators disagreement occur while annotating a dataset for sentiment analysis.

Cohen's kappa coefficient (k) is used to measure inter-annotators agreement [51][52]. Each raters classify 300 items into 3 mutually exclusive categories. Inter-annotation coefficient Kappa(k) is calculated as given in Equation 5.8. Cohen's kappa coefficient(k) ranges from -1 to 1 [52]. According to Cohen the Kappa values <=0 indicates no agreement, values ranges from 0.01 to 0.20 show slight agreement, values between 0.21 and 0.40 are fair, values between 0.41 and 0.60 are moderate, values between 0.61 and 0.80 as substantial, and values ranges from 0.81 to 1.00 indicate a perfect agreement[52].

$$k = \frac{po - pe}{1 - pe} \tag{5.8}$$

where

po: relative observed agreement between raters.

pe: expected agreement. It is the proportion of items for which agreement is expected by chance when the items are random. Table 5.11 shows the agreement count between the annotators on the 300 comments.

No	comment	First annotator polarity	Second annotator polarity
1	**ወይኔ እንባዬን መቆጣጠር ነው ያቃተኝ**	negative	positive
2	**መቼ ነው ቁጥር ሶስት የሚወጣው**	positive	neutral
3	**አቤት አለባበስ**	positive	negative
4	**ወይኔ ቶማስ አደከምከኝ**	negative	positive
5	**አሪፍ ነው ግን ድምፁ በጣም ይደብራል እደኔ ድምፁ ያልተመቸው በተረፈ በርቱልን በጣም ደስ ይላል**	negative	positive
6	**ደስ አይልም ግን እራሄልን እወዳታለሁ**	positive	negative
7	**ሴትን ልጅ ታሳዝናታለህ እንጅ አታታልላትም ቤቱ በጣም ያምራል የልጅቷም ጥንካሬ እንደዛው ግን ፍቅር ያሸንፋል**	negative	positive
8	**እማይመጣ የለ**	neutral	negative

Table 5.10: Sample inter-annotators disagreement

	Annotator two (B)			
		positive	negative	neutral
Annotator one (A)	positive	106	10	3
	negative	7	149	6
	neutral	9	7	3

Table 5.11: Inter-annotator agreement count

The observed agreement is calculated as:

$$po = \frac{((Apositive, Bpositive) + (Anegative, Bnegative) + (Aneutral, Bneutral))}{total} \tag{5.9}$$

po=(149+106+3)/300=0.86

The expected agreement is calculated as:

$$pe = Ppositive + Pnegative + Pneutral \tag{5.10}$$

Here P refers the probability of being positive, negative or neutral.

$$Ppositive = \frac{((Apositive, Bpositive) + (Apositive, Bnegative) + (Apositive, Bneutral))}{total} * \frac{((Bpositive, Apositive) + (Bpositive, Anegative) + (Bpositive, Aneutral))}{total} \tag{5.11}$$

P positive=(106+10+3)/300*(106+7+9)/300=0.156

In the same way we calculate P negative and P neutral as:

P negative=(149+7+6)/300*(149+10+7)/300=0.298

P neutral=(3+9+7)/300*(3+6+3)/300=0.0025

pe=P positive+P negative+ P neutral=0.4565

where: Sample notations used in the equations:

P positive= probability of being positive from the total comments

(A positive, B negative)= A value that annotator one (A) says positive but annotator two (B) says negative.

The Cohen's kappa coefficient(k) using po=0.86,pe=1.999 and Equation 5.8 is:

k=(0.86-0.4565)/(1-0.4565)=0.7424. Cohen k= 0.7424 interpreted as a substantial agreement between the annotators.

From Table 5.10 and kappa coefficient(k=0.7424), this paper generalize there is no a total agreement between the inter-annotators in all comments they are assigned to label. However, most disagreements are resolved through discussions.

5.7 Threats to validity

In this research treat to validity may arise from the selection of feature size and classifiers algorithms. For investigating the effect of stemming on Amharic sentiment analysis, we have stemmed 1077 comments and develop one LSTM layer with eight neurons, and eight word embedding length. Other configurations of word embedding length, number of layers and number neurons on each layer may give a different result.

Deep neural networks are prone to overfitting to reduce this unwanted characteristics the paper used three techniques(early stopping, reducing neural network size and a dropout range from 0.5 to 0.8). However, other combination of overfitting reduction techniques may result in different result.

Chapter 6

Conclusions and Future Works

6.1 Conclusion

The main objective of this research is to investigate the effect of integrating stemming as part of preprocessing and to show the effect of adding labeled Emojis with lexicons for automatic labeling system on Amharic sentiment analysis.

The results in this paper show that stemming has reduced the accuracy from 82.36% to 75.93% and from 90.7% to 90.27% for both LSTM and Multinomial Naive Bayes Bi-gram (MNB) model respectively. This paper has concluded that stemming has no positive impact on Amharic sentiment analysis. The results on the paper show that integrating labeled Emojis with lexicons on automatic labeling system has increased the accuracy of LSTM model from 86.43% to 86.98%. A 0.55% improvement is insignificant, this leads to the conclusion that integrating emoji has a negligible effect on the labeling accuracy. A comparison of automatic annotation with the human-annotation is measured by taking 3000 comments from the total dataset without considering human annotators disagreement. The automatic and human annotation has 90.67 overlap similarity value. We have concluded that automatic approach can replace human annotation with certain modifications as the automatic annotation has minimal overhead in terms of time and energy required.

6.2 Future Works

Developing efficient fully-functional sentiment analysis requires coordinated team efforts that comprise linguistic professional, computer science professional and other people that can collect more comments from both public society and social media. Good coordination of these different professionals can result in a full functional sentiment analysis model. This research paper identifies the following direction as future work.

1. This work analyzes the effect of stemming on sentiment analysis using HornMorpho 2.5 Amharic morphological analyzer. In future we plan to test the approach using other Amharic morphological analyzers.

2. This research paper uses automatic annotation using labeled lexicons and Emojis. In the future we plan to use a dependency tree-based parser which consists direct relations between lexical items in a sentence. This relationship directly encode important information in a complex phrase structure. A dependency parse tree is created by consistently parsing an input sentence into a sequence of dependency relations which help to handle syntactic relation in a phrase. A dependency parse tree is more advantageous for morphologically rich languages.

3. This research paper on sentiment analysis is limited to comments written in Amharic language only. However, most users write a comment by transliterating Amharic alphabets to English alphabets and others write a comment by combining Amharic language with other languages like English, Tigrigna, and Oromigna. For next works adding these languages will help in developing a full model of sentiment analysis.

4. As a future work, we plan to develop an ensemble of CNN and LSTM,CNN and BI-LSTM networks or jointly train CNN-LSTM,CNN-BI-LSTM to improve the prediction accuracy.

Appendix A

Normalized characters

The following are some Normalized characters

ዐ=አ	ያ=አ	ኅ=ሀ	ሓ=ሀ	ጳ=ፃ	ሜ=ሴ
ዓ=አ	ዉ=ው	ኽ=ህ	ሔ=ሄ	ጼ:ፄ	ሥ=ስ
ዑ=ኡ	ኃ=ሀ	ሐ=ሆ	ኇ=ሆ	ጸ=ፆ	ሦ=ሶ
ዒ=ኢ	ጕ=ሁ	ሃ=ሀ	ጽ=ፅ	ሠ=ሰ	
ዓ=አ	ጒ=ሂ	ሐ=ሀ	ጸ=ፀ	ሡ=ሱ	
ዔ=ኤ	ኃ=ሀ	ሑ=ሁ	ጹ=ፁ	ሚ=ሲ	
ዕ=እ	ጔ=ሄ	ሒ=ሂ	ጼ=ፂ	ሣ=ሳ	

Word Normalization

The following list show Normalized words

ኢቲዮጲያ=ኢትዮጵያ	1ኛ=አንደኛ	ኡፍፍፍፍፍፍ=ኡፍፍ	ሰውደወ=ስወደው
ኮመንት=አስተያየት	ይመችክ=ይመችህ	ኡፍፍፍፍፍፍ=ኡፍፍ	
ሙቭ=ፊልም	ላይክ=ዉደድ	ውድድድድድ=ውድድ	
ኢትዬ=ኢትዮጵያ	ዶ/ር ዶክተር	ፍቅርርር=ፍቅር	
ፍልም=ፊልም	ዋውውውው=ዋው	ውድድድ:ውድድ	

Appendix B

LSTM model

```
topwords=15000
maximumlen=433
epoch=8
batch=14
embedding_size = 8
model=Sequential()
np.random.seed(0)
model.add(Embedding(topwords,8,input_length=maximumlen,dropout=0.5))
model.add(LSTM(8,return_sequences=True,input_shape=(x_train.shape[0],x_train.shape[1])))
model.add(Dropout(0.7))
model.add(Flatten())
model.add(Dense(16, activation='sigmoid'))
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
history=model.fit(x_train,y_train,validation_data=(x_val,y_val),epochs=epoch,batch_size=batch,ver
callbacks=[EarlyStopping(monitor='val_loss', patience=5, baseline=0.3)]) loss, accuracy =
model.evaluate(x_test, y_test,verbose = 2, batch_size=batch)
print("Training_Accuracy::.4f".format(accuracy))
plt.style.use('ggplot')
def plot_history(history):
```

```
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
x = range(1, len(acc) + 1)
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(x, acc, 'b', label='Training_acc')
plt.plot(x, val_acc, 'r', label='Validation_acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.subplot(1, 2, 2)
plt.plot(x, loss, 'b', label='Training_loss')
plt.plot(x, val_loss, 'r', label='Validation_loss')
plt.title('Training and validation_loss')
plt.legend()
plot_history(history)
```

Reference

[1]. Philemon, W. and Mulugeta, W., A Machine Learning Approach to Multi-Scale Sentiment Analysis of Amharic Online Posts.

[2]. Tilahun, T., 2014. Linguistic Localization of Opinion Mining from Amharic Blogs. International Journal of Information Technology Computer Sciences Perspectives, 3(1), p.890.

[3]. Gebremeskel, S., 2010. Sentiment Mining Model for Opinionated Amharic Texts. Unpublished MastersThesis, Addis Ababa University, Addis Ababa.

[4]. Alemu, Y. G. M. A. Deep Learning Approach For Amharic Sentiment Analysis University Of Gondar.

[5]. Yosef Arega, 2019 Trend Detection and Sentiment Analysis in Amharic Social Media,Unpublished MastersThesis, Addis Ababa University, Addis Ababa.

[6]. McCallum, A. and Nigam, K., 1998, July. A comparison of event models for naive bayes text classification. In AAAI-98 workshop on learning for text categorization (Vol. 752, No. 1, pp. 41-48).

[7]. Eyassu, S. and Gambäck, B., 2005. Classifying Amharic news text using self-organizing maps.

[8]. Kelemework, W., 2013. Automatic Amharic text news classification: A Neural networks approach. Ethiopian Journal of Science and Technology, 6(2), pp.127-137.

[9]. Gebre, B.G., 2010. Part of speech tagging for Amharic (Doctoral dissertation, University of Wolverhampton Wolverhampton).

[10]. Hidayatullah, A.F., 2015. The Influence of Stemming on Indonesian Tweet Sentiment Analysis. Proceeding of the Electrical Engineering Computer Science and Informatics, 2(1), pp.127-132.

[11]. Jabbar, H. and Khan, D.R.Z., 2015. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). Computer Science, Communication and Instrumentation Devices.

[12]. Na, J.C., Kyaing, W.Y.M., Khoo, C.S., Foo, S., Chang, Y.K. and Theng, Y.L., 2012, November. Sentiment classification of drug reviews using a rule-based linguistic approach. In International conference on asian digital libraries (pp. 189-198). Springer, Berlin, Heidelberg.

[13]. Yang, C.S. and Shih, H.P., 2012. A Rule-Based Approach For Effective Sentiment Analysis. In PACIS (p. 181).

[14]. Wahbeh, A., Al-Kabi, M., Al-Radaideh, Q., Al-Shawakfa, E. and Alsmadi, I., 2011. The effect of stemming on arabic text classification: an empirical study. International Journal of Information Retrieval Research (IJIRR), 1(3), pp.54-70

[15]. Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). "Efficient Estimation of Word Representations in VectorSpace". In: CoRRabs/1301.3781. arXiv:1301.3781.

[16]. Pennington, J., Socher, R., and Manning, C. D. (2014). "GloVe: Global Vectors for Word Representation". In: Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543

[17]. Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). "Enriching Word Vectors with Subword Information".In: Transactions of the Association for Computational Linguistics5, pp. 135–146

[18]. Hutto, C.J. and Gilbert, E., 2014, May. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Eighth international AAAI conference on weblogs and social media.

[19]. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. (2018). "Advances in Pre-Training Distributed Word Representations". In Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018).

[20]. Mandelbaum, A. and Shalev, A., 2016. Word embeddings and their use in sentence classification tasks. arXiv preprint arXiv:1610.08229

[21]. Alayba, A.M., Palade, V., England, M. and Iqbal, R., 2018, March. Improving sentiment analysis in Arabic using word representation. In 2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR) (pp. 13-18). IEEE

[22]. Olah, C. (2015) "Understanding lstm networks", http://colah.github.io/posts/2015-08-Understanding-LSTMs/. Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[23]. Lipton, Z.C., Berkowitz, J. and Elkan, C., 2015. A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019.

[24]. Gasser, M. (2012) 'HornMorpho 2.5 User's Guide', pp. 1–55

[25]. Dagimawi, D., 2017. Amharic Named Entity Recognition Using Neural Word Embedding as a Feature (Doctoral dissertation, AAU).

[26]. Mckinney, W. and Pydata Development Team (2014) "Pandas : powerful python data analysis toolkit release 0.13.1", Python package, p. 1211.

[27]. Asghar, M.Z., Khan, A., Ahmad, S., Qasim, M. and Khan, I.A., 2017. "Lexicon-enhanced sentiment analysis framework using rule-based classification scheme". PloS one, 12(2), p.e0171649.

[28]. Romero Llombart, Ò., "Using machine learning techniques for sentiment analysis".

[29]. Ain, Q.T., Ali, M., Riaz, A., Noureen, A., Kamran, M., Hayat, B. and Rehman, A., 2017. Sentiment analysis using deep learning techniques: a review. Int J Adv Comput Sci Appl, 8(6), p.424.

[30]. Huang, Q., Chen, R., Zheng, X. and Dong, Z., 2017, August. Deep sentiment representation based on CNN and LSTM. In 2017 International Conference on Green Informatics (ICGI) (pp. 30-33). IEEE.

[31]. Xue, W. and Li, T., 2018. Aspect based sentiment analysis with gated convolutional networks. arXiv preprint arXiv:1805.07043

[32]. Severyn, A. and Moschitti, A., 2015, August. Twitter sentiment analysis with deep convolutional neural networks. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 959-962). ACM.

[33]. Dahou, A., Xiong, S., Zhou, J., Haddoud, M.H. and Duan, P., 2016, December. Word embeddings and convolutional neural network for arabic sentiment classification. In Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers (pp. 2418-2427)

[34]. Alayba, A.M., Palade, V., England, M. and Iqbal, R., 2018, August. A combined CNN and LSTM model for arabic sentiment analysis. In International Cross-Domain Conference for Machine Learning and Knowledge Extraction (pp. 179-191). Springer, Cham

[35]. Pang, B., Lee, L. and Vaithyanathan, S., 2002, July. Thumbs up?: sentiment classification using machine learning techniques. In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10 (pp. 79-86). Association for Computational Linguistics.

[36]. Kamal, A., 2013. Subjectivity classification using machine learning techniques for mining feature-opinion pairs from web opinion sources. arXiv preprint arXiv:1312.6962.

[37]. Chockalingam, N., 2018. Simple and Effective Feature Based Sentiment Analysis on Product Reviews using Domain Specific Sentiment Scores. Polibits, 57, pp.39-43.

[38]. Dos Santos, C. and Gatti, M., 2014, August. Deep convolutional neural networks for sentiment analysis of short texts. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers (pp. 69-78).

[39]. Kralj Novak P, Smailović J, Sluban B, Mozetič I (2015) Sentiment of Emojis. PLoS ONE 10(12): e0144296. https://doi.org/10.1371/journal.pone.0144296

[40]. Ljubešić, N. and Fišer, D., 2016, August. A global analysis of emoji usage. In

Proceedings of the 10th Web as Corpus Workshop (pp. 82-89).

[41]. Shiha, M. and Ayvaz, S., 2017. The effects of emoji in sentiment analysis. Int. J. Comput. Electr. Eng.(IJCEE.), 9(1), pp.360-369

[42]. Ruangkanokmas, P., Achalakul, T. and Akkarajitsakul, K., 2016, January. Deep belief networks with feature selection for sentiment classification. In 2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS) (pp. 9-14). IEEE.

[43]. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), pp.1929-1958.

[44]. Kralj Novak, P., Smailović, J., Sluban, B. and Mozetič, I., 2015,Emoji Sentiment Ranking 1.0, viewed 30 April 2019, Retrieved from <http://kt.ijs.si/data/Emoji_sentiment_ranking>

[45]. Mroth, emojitracker: realtime emoji use on twitter, (July,2013),viewed 15 May 2019, Retrieved from <www.emojitracker.com>

[46]. Lebret, R.P., 2016. Word Embeddings for Natural Language Processing (No. THESIS). EPFL.

[47]. Duwairi, R. and El-Orfali, M., 2014. A study of the effects of preprocessing strategies on sentiment analysis for Arabic text. Journal of Information Science, 40(4), pp.501-513.

[48]. Prechelt, L., 1998. Early stopping-but when?. In Neural Networks: Tricks of the trade (pp. 55-69). Springer, Berlin, Heidelberg.

[49]. Minaee, S., Azimi, E. and Abdolrashidi, A., 2019. Deep-Sentiment: Sentiment Analysis Using Ensemble of CNN and Bi-LSTM Models. arXiv preprint arXiv:1904.04206.

[50]. Ruder, Sebastian. 2016a. "On word embeddings - Part 1." April 11, Retrieved from https://ruder.io/word-embeddings-1/

[51]. Bhowmick, P.K., Basu, A. and Mitra, P., 2008, August. An agreement measure for

determining inter-annotator reliability of human judgements on affective text. In Coling 2008: Proceedings of the workshop on Human Judgements in Computational Linguistics (pp. 58-65).

[52]. McHugh, M.L., 2012. Interrater reliability: the kappa statistic. Biochemia medica: Biochemia medica, 22(3), pp.276-282.

[53]. Arif, S.M. and Mustapha, M., 2017. The effect of noise elimination and stemming in sentiment analysis for Malay documents. In Proceedings of the International Conference on Computing, Mathematics and Statistics (iCMS 2015) (pp. 93-102). Springer, Singapore.

[54]. Krawczyk, B., 2016. Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence, 5(4), pp.221-232.

[55]. Branco, P., Torgo, L. and Ribeiro, R., A survey of predictive modelling under imbalanced distributions. arXiv 2015. arXiv preprint arXiv:1505.01658.