

Bilingual Social Media Text Hate Speech Detection For Afaan Oromo and Amharic Languages Using Deep Learning



Baharudin Sherif Kemal

A Thesis Submitted to the Department of Computer Science and Engineering,
School of Electrical Engineering and Computing

Presented in Partial Fulfillment for the Degree of Masters of Science in
Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

August, 2021

Adama, Ethiopia

Bilingual Social Media Text Hate Speech Detection For Afaan Oromo and Amharic Languages Using Deep Learning

By: Baharudin Sherif Kemal

Advisor: Teklu Urgessa (PhD.)

A Thesis Submitted to the Department of Computer science and Engineering,
School of Electrical Engineering and Computing

Presented in Partial Fulfillment for the Degree of Masters of Science in
Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

August, 2021

Adama, Ethiopia

Declaration

I hereby declare that this Master Thesis entitled “**Bilingual Social Media Text Hate Speech Detection For Afaan Oromo And Amharic Languages Using Deep Learning**” is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Baharudin Sherif

Name of the Student

Signature

Date

Recommendation

I, the advisor of this thesis, hereby certify that I have read the revised version of the thesis entitled **“Bilingual Social Media Text Hate Speech Detection For Afaan Oromo And Amharic Languages Using Deep Learning”** prepared under my guidance by Baharudin Sherif Kemal submitted in partial fulfillment of the requirements for the degree of Master’s of Science in Computer Science and Engineering. Therefore, I recommend the submission of the revised version of the thesis to the department following the applicable procedures.

Advisor

Signature

Date

Approval of Board of Examiners

I, the advisor of the thesis entitled “**Bilingual Social Media Text Hate Speech Detection For Afaan Oromo And Amharic Languages Using Deep Learning**” and developed by Baharudin Sherif Kemal, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

_____	_____	_____
Advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Baharudin Sherif Kemal have read and evaluated the thesis entitled “Bilingual Social Media Text Hate Speech Detection For Afaan Oromo And Amharic Languages Using Deep Learning” and examined the candidate during the open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

_____	_____	_____
Chairperson	Signature	Date

_____	_____	_____
Internal Examiner	Signature	Date

_____	_____	_____
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____	_____	_____
Department Head	Signature	Date

_____	_____	_____
School Dean	Signature	Date

_____	_____	_____
Office of Postgraduate Studies, Dean	Signature	Date

Acknowledgments

In the name of Allah, the Most Gracious and the Most Merciful Alhamdulillah, all Praises to Allah for the strengths and His blessing in completing this research. Then, my special thanks go to my advisor, Dr. Teklu Urgessa for his friendly approach, readiness, and willingness to advise on the thesis, his critical comments greatly improved this thesis.

Also, I would like to acknowledge the members of the Smart Software SIG of the CSE department. Including the SIG supervisor Dr. Mesifn Abebe for his continued commitment to guide and support this research from its inception to completion.

Then, I would like to thank all my classmates who have been working on their master's thesis for their timely help, idea, and support until the completion of my thesis.

Next, my special thanks go to Mr. Sisay, who is serving as a judge at Adama City higher Court, for providing me advice on hate speech laws, helping me to have deep insight into the Ethiopian hate speech laws. Also, I would like to thank Mr. Befekadu, who is working in the organization named “Center for Advancement of Rights and Democracy”, for giving his time to discuss with me hate speech on social media and providing me the anti-hate speech guideline which is prepared by their organization.

Then, I would like to acknowledge all data annotators for their willingness to perform the annotation process, which is boring and time taking, and they have a great contribution to the completion of my thesis.

After that, I would like to thank all my beloved family including my sisters, my brothers, my mother, and my Dad, for wishing me a bright future and helping me in all aspects they can.

TABLE OF CONTENTS

Declaration	i
Recommendation	ii
Approval of Board of Examiners	iii
Acknowledgments.....	iv
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF EQUATIONS	xii
LIST OF ABBREVIATIONS AND ACRONYMS	xiii
ABSTRACT.....	xiv
CHAPTER ONE	1
INTRODUCTION	1
1.1 Background of the Study	1
1.2 Motivation of the Study	3
1.3 Statement of the Problem.....	3
1.4 Research Questions	5
1.5 Objectives of the Study	5
1.5.1 General Objective	5
1.5.2 Specific Objectives	5
1.6 Scope and Limitations of the Study	5
1.6.1 Scope of the Study	5
1.6.2 Limitations of the Study.....	5
1.7 Significance of the Study	6
1.8 Organization of the Thesis	6
CHAPTER TWO	8

LITERATURE REVIEW AND RELATED WORKS	8
2.1 Hate Speech	8
2.1.1 Definition of Hate Speech.....	8
2.3 The Social Media	10
2.4 Hate Speech on Social Media	10
2.5 Techniques Used for Hate Speech Detection.....	11
2.5.1 Machine Learning Techniques.....	11
2.5.2 Deep Learning Techniques	12
2.5.3 Feature Extraction Techniques	16
2.6 Afaan Oromo Language.....	19
2.6.1 Writing system Of Afaan Oromo.....	19
2.6.2 Word Categories of Afaan Oromo.....	19
2.6.3 Afaan Oromo Sentence Structure	20
2.6.5 Afaan Oromo Punctuation	20
2.7 Amharic Language.....	21
2.7.1 Amharic Character Representation	21
2.8 Related Works on Hate Speech Detection	22
CHAPTER THREE	27
RESEARCH METHODOLOGY.....	27
3.1 Data Collection Method	27
3.1.1 Criteria for Facebook Page Selection	28
3.2 Data Preparation.....	28
3.2.1 Data Annotation	30
3.3 Bilingual Hate Speech Detection Modeling	31
3.3.1 Data Preprocessing.....	31

3.3.2 Feature Extraction Methods	32
3.3.3 Deep Learning Models.....	32
3.4 Evaluation Method.....	33
3.4.1 Inter Annotator Agreement (IAA)	33
3.4.2 Model Evaluation Methods.....	34
3.5 Model Prototyping	36
CHAPTER FOUR.....	37
THE PROPOSED SOLUTION FOR THE BILINGUAL HATE SPEECH DETECTION	37
4.1 The Proposed Architecture	37
4.2 Preprocessing	38
4.2.1 Cleaning	38
4.2.2 Normalization	39
4.2.3 Tokenization	40
4.2.4 Stopwords Removal	40
4.3 The Proposed Feature Representation	41
4.3.1 Word Embeddings	41
4.4 The Proposed Deep Learning Model Architectures.....	41
4.4.1 Bidirectional RNN Models	41
4.4.2 Attention Mechanism.....	45
CHAPTER FIVE	48
IMPLEMENTATION OF THE PROPOSED BILINGUAL HATE SPEECH DETECTION.....	48
5.1 Tools	48
5.1.1 Data Preparation And Preprocessing Tools	48
5.1.2 Package Managers and Environments	49
5.1.3 Modeling Tools and Packages.....	50

5.1.4 Deployment Tools.....	51
5.1.5 Hardware Tools.....	51
5.2 Data Preprocessing and Model Implementation	51
5.2.1 Loading Dataset	51
5.2.2 Preprocessing	52
5.2.3 Tokenization	53
5.2.4 Implementation For Word2vec	54
5.3 Model Implementation.....	56
5.3.1 Recurrent Neural Networks	57
5.3.2 Attention Mechanism.....	58
5.3.3 Prototyping.....	58
CHAPTER SIX.....	59
RESULTS AND DISCUSSION	59
6.1 Model Performance Results.....	59
6.1.1 BILSTM Confusion Matrix	61
6.1.2 Bi-GRU Confusion Matrix	61
6.1.3 Performance of Attention Based Models.....	62
6.2 Hyperparameter Tuning.....	64
6.3 Model Testing Using New Input Data	67
6.4 Discussions	67
CHAPTER SEVEN	70
CONCLUSION AND FUTURE WORKS	70
7.1 Conclusion	70
7.2 Future Works	71
REFERENCES	72

APPENDICES	i
4. APPENDIX A: ANTI-HATE SPEECH GUIDELINE.....	i
APPENDIX B: LIST OF STOPWORDS	iii
Appendix B.1 Sample Amharic Stopwords	iii
Appendix B.2 Sample Afaan Oromo Stopwords	iv
5. APPENDIX C: SAMPLE HATEFUL KEYWORDS IDENTIFIED BY CARD	v
APPENDIX D: MODEL SCORES PER-FOLD	vi
APPENDIX E: GRAPHICAL USER INTERFACE	viii
Appendix E.1 Predicting Texts as “Hate”	viii
Appendix E.2 Predicting Texts as “Free”	ix
APPENDIX F: SAMPLE CODE FOR PREPROCESSING	x
APPENDIX G: AMHARIC LANGUAGE WORD SIMILIARITY EXAMPLE IN THE SKIP-GRAM MODEL	xi
APPENDIX H: AFAAN OROMO LANGUAGE WORD SIMILIARITY EXAMPLE IN THE SKIP- GRAM MODEL	xii

LIST OF TABLES

Table 1.1 Sample Social Media Hate Speeches.....	4
Table 2.1 Afaan Oromo word class	20
Table 2.2 Afaan Oromo' Language Punctuation Marks	21
Table 2.3 Summary of Related Works.....	25
Table 3.1 Selected Facebook Pages and Collected Data	29
Table 5.1 Parameters For Training the Word2vec model.....	55
Table 6.1 Model Test Results Using 5-Fold Cross Validation	59
Table 6.2 Model Test Results Using 10-Fold Cross Validation	60
Table 6.3 Summary of Models Performance Using Word2vec and Without Using Word2vec.....	64
Table 6.4 Hyperparameter Tuning Trials.....	65
Table 6.5 Comparison of The Proposed BiLSTM and BiGRU+Attention Models.....	66

LIST OF FIGURES

Figure 2.1 Artificial Neural Networks	12
Figure 2.2 Recurrent Neural Networks	13
Figure 2.3 Architecture of Bidirectional LSTM	14
Figure 2.4 Continuous Bag of Words Architecture	18
Figure 2.5 Skip-gram Model Architecture.....	18
Figure 3.1 Strategy For Collecting Afaan Oromo Data	27
Figure 4.1 General Architecture for The Proposed Bilingual Hate Speech Detection Modeling.....	37
Figure 4.2 The Proposed Skip-gram Model Architecture.....	41
Figure 4.3 The Proposed BiLSTM Architecture.....	43
Figure 4.4 LSTM Cell	43
Figure 4.5 The proposed Bi-GRU Architecture.....	44
Figure 4.6 GRU Cell	45
Figure 4.7 The Proposed Attention Mechanism	46
Figure 5.1 Implementation For Loading Dataset	52
Figure 5.2 Sample Code For Data Cleaning and Stopwords Removal	52
Figure 5.3 Sample Code For Character Normalization	53
Figure 5.4 Implementation For Tokenization	54
Figure 5.5 Sample Code For Training Word2vec.....	55
Figure 5.6 Sample Code For Loading Word2vec and Assign The Weight to Variable	56
Figure 5.7 Importing Required Packages For Modeling	56
Figure 5.8 Sample Code to Create Bidirectional LSTM Model	57
Figure 5.9 Graphical User Interface.....	58
Figure 6.1 BiLSTM Confusion Matrix	61
Figure 6.2 Bi-GRU Confusion Matrix	62
Figure 6.3 Confusion Matrix for BiLSTM+Attention	63
Figure 6.4 Confusion Matrix for Bi-GRU+Attention.....	63
Figure 6.5 Hyperparameter Tuning Results.....	66

LIST OF EQUATIONS

Equation 2.1 Calculating Term Frequency in Document	16
Equation 2.2 Calculating Inverse Document Frequency	17
Equation 3.1 Calculating Accuracy	35
Equation 3.2 Calculating Precision.....	35
Equation 3.3 Calculating Recall	36
Equation 3.4 Calculating F1-score.....	36
Equation 4.1 Attention Weight Alignment	47
Equation 4.2 Softmax Calculation	47
Equatio 4.3 Context Vectors Calculation	47

LIST OF ABBREVIATIONS AND ACRONYMS

A

ANN · Artificial Neural Network, Artificial Neural Network
API · Application Program Interface
ASCII · American Standard Code for Information Interchange

B

BERT · Bidirectional Encoder Representations from Transformers
Bi-GRU · Bidirectional Gated Recurrent Unit
BiLSTM · Bidirectional Long Short Term Memory
BOW · Bag of Words

C

CARD · Center For Advancement of Rights and Democracy
CBOW · Continuous Bag of Words
CNN · Convolutional Neural Network

F

FN · False Negative
FP · False Positive

G

GPU · Graphics Processing Unit
GRU · Gated Recurrent Unit

H

HTTP · Hypertext Transfer Protocol

I

IAA · Inter Annotator Agreement

L

LASER · Language Agnostic Sentence Representation
LR · Logistic Regression
LSTM · Long Short Term Memory

M

MUSE · Multilingual Unsupervised and Supervised Embedding

N

NB · Naive Bayes
NLTK · Natural Language Toolkit

P

POS · Part of Speech Tag

R

RF · Random Forest
RNN · Recurrent Neural Network

S

SVM · Support Vector Machine

T

TF-IDF · Term Frequency Inverse Document Frequency
TN · True Negative
TP · True Positive

U

URL · Uniform Resource Locator

ABSTRACT

Hate speech on social media has become a challenging problem in the past few years. Despite the importance of social media for communication and information sharing purposes, it is observable that people are using it as a playground for hate speech against a targeted group based on their protected identity such as religion, ethnic group, and gender. Recently, hate speech have become a common problem also in Ethiopia. The problem can lead to violent actions in society when it is shared by a large community. Detecting hate speech texts on social media is a tedious and complex task due to the unstructured format of social media content. In the past recent years hate speech texts on social media have become more challenging and it attracted several researchers to work on hate speech detection. Due to the success of deep learning algorithms in natural language processing tasks, some researchers proposed deep learning models for hate speech detection. Even if the problem of hate speech is not language-specific, most of the studies are explored only for high resource languages like English except some studies recently proposed also for low resource languages. Also, most works on this problem are focused on a single language, and as a solution, this study proposed bilingual hate speech detection for Afaan Oromo and Amharic text on social media using deep learning. A bilingual dataset prepared from newly collected Afaan Oromo text from the Facebook platform and the existing binary Amharic dataset is adopted to develop models. The newly collected data is obtained by scraping selected Facebook pages using Fcepager graph API and annotated by 5 annotators. The prepared dataset contains binary classes “Hate” and “Free”. Bidirectional RNNs and attention mechanisms are implemented using Word2vec as feature representation. The word2vec model is trained based on skip-gram model due to its suitability for representing non frequent keywords of limited size dataset. Additionally LSTM and GRU networks are also implemented for model comparison. The models are trained using 5-fold and also 10-fold cross validation. The results shows that models achieved good performance when using 5-fold cross validation on our dataset. Then, several experiments are employed to select the best-performing model and finally, the BiLSTM model outperformed all other models with an accuracy of 94.3% and f1_score of 94.2%.

Keywords: Hate Speech Detection, Deep Learning, Natural Language Processing, Low-Resource Languages

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

The number of internet users is increasing daily since internet service is expanding which is attracting a lot of customers. According to (Clement, 2020) 59 percent of the global population were active internet users as of October 2020. The availability of social media platforms such as Facebook, Twitter, Instagram and YouTube playing an antwerping role in attracting the world to use the internet. In Ethiopia also, the user of social media has grown significantly in recent years. Social media platforms can disseminate information quickly and widely as a means of communication. It has opened up a new galaxy of opportunities for people to express their opinions freely. Nowadays people are using social media for online trading, dissemination of government policies, political campaigns, and religious preaching. The other importance of social media is that its application in areas like health, education, and business for sharing useful information (Akram & Kumar, 2017).

Despite its importance, social media created an opportunity for the propagation of hate speech and disinformation online (Nayel & Shashirekha, 2019). Having many audiences, some peoples use to share their hate speech propaganda online. Online hate speech can be accessed by a large number of the audience quickly which makes the problem very difficult. The term hate speech refers to a speech that promotes violence or attack on any protected category of people based on their race, religion, sex, ethnicity, national origin, sexual orientation, gender, or disability(Aluru et al., 2020). It includes all kinds of expressions that promote attack and discrimination or incites violence against a targeted group of people or individuals based on their protected identity. Hate speech towards targeted groups or individuals can lead to serious conflicts on the ground by motivating people for criminal actions. In Ethiopia, there is a diversity of cultures, religions, ethnic groups and languages. Any kind of hate speech against a targeted group or individuals can be a reason for conflict in Ethiopia.

Many countries in the world have laws that govern hate speech (Saksesi et al., 2018). Recently Ethiopia posted a hate speech prevention proclamation. It defined the term hate speech as ‘speech that deliberately promotes hatred, discrimination or attack against a person or a discernable group of identity, based on race, ethnicity, religion, gender or disability. There are also definitions given by different social media platforms and governmental organizations for the term hate speech. Definitions given by different sources for the term hate speech are almost similar in terms of their contents. Also, there are other concepts such as cyberbullying, discrimination, flaming and Extremism having related

meaning with hate speech. But, hate speech is differentiated from its related concepts in that hate speech is aimed to attack or diminish, incite violence, and targeted the specific identity of a person or group (Fortuna & Tec, 2020). Hate speech on social media is becoming a major problem in the world in recent years. USA grants companies of social media broad managing power of their content and enforcing the rule of hate speech. Companies in different countries like Germany were forced by the government to remove posts within certain periods¹. In Ethiopia, the result of hate speech posts on social media by different individuals made several conflicts (Mossie & Wang, 2018).

Social media platforms including Facebook, Instagram, and Twitter struggling to use artificial intelligence (AI) applications on their site to block hate speech automatically and make safe environments for their customers². In the past few year's researchers were focusing on the problem of hate speech detection and proposed different NLP and machine learning techniques to solve the problem. Most existing researches on the detection of hate speech are explored for some high-resource languages like English. But nowadays people are using social media as a playground for hate speech and insults against targeted groups or individuals. Hence, there should be a mechanism for hate speech detection also for under-resourced languages. (Kenenisa & Melak, 2019; Mossie & Wang, 2018) proposed a machine learning technique, and (Baweke, 2020; Tesfaye & Tune, 2020) proposed a deep learning approach for hate speech detection of Amharic texts on social media. There are also some other existing works on hate speech detection using machine learning (Aulia, 2019; Gaydhani et al., 2018; Nayel & Shashirekha, 2019), and deep learning (Badjatiya et al., 2017; Zhang, 1900).

However, there is no research conducted on bilingual hate speech detection for Afaan Oromo and Amharic languages. The research work proposed bilingual social media hate speech detection for Afaan Oromo and Amharic languages. The dataset is collected from the Facebook platform. The proposed hate speech detection classifies the dataset into two classes, which are 'Hate', and 'Free'. Bidirectional recurrent neural networks and attention mechanisms were implemented using word2vec feature representation for the experiment. The models are tested using 5-fold cross-validation. Best performance is achieved by the BiLSTM model with word2vec under 5-fold testing. The model performance is compared by training without word2vec but, using the trained word2vec achieved good performance for all models. Finally, even if no better performance is achieved manual hyperparameter tuning is performed for the BiLSTM model by adjusting 15 trials of hyperparameter combination.

¹ <https://www.cfr.org/background/hate-speech-social-media-global-comparisons>.

² <https://klingmarketing.com/ai-social-media>

1.2 Motivation of the Study

As a user of social media, we observed that hate speech posts, especially against specific ethnic group, become daily activity by many individuals in Ethiopia. It is our recent memory that in Ethiopia the problem even challenged the government and several times the Ethiopian government tried to manage the consequences through interruption of the internet service(Mossie & Wang, 2018).

Social media hate speech stayed the challenging problem for social media organizations like Facebook and Twitter despite they are struggling to develop AI for hate speech detection. Recently, the Ethiopian government posted a proclamation for hate speech prevention that enforces social media service providers to remove hate speech texts on their platforms. This is because any hate speech on social media can lead to serious conflict in society on the ground.

Social media texts hate speech detection remains a tedious task that the user has to report hate speech to the social network service providers for it to be removed from the platform. But, the unstructured format of social media data added difficulty to track it manually. Hence, we are inspired to work on the development of a hate speech detection model because it is important to simplify the task of hate speech detection on social media, which makes social media a safe environment for users. Also, we need to take a role in preparing benchmark datasets for future researchers and to take part in the improvement of the hate speech detection system.

1.3 Statement of the Problem

Since social media consists of a large amount of user-generated content, hate speech detection is far from perfect to detect hate speech texts on social media. Hate speech posted and shared on social media against any targeted group can cause conflicts in the community since it is accessed by many social media users.

The dissemination of hate speech on social media will not only affect online society, but its consequence will be more harmful when the information shared across social media start to cause evident problems to the public at large. Since there is a diversity of cultures, religions, and different identities in Ethiopia hate speech on any individuals or group based on their identity can contribute a lot in making conflicts. The following table shows some representative examples of social media hate speech in Ethiopia:

Table 1.1 Sample Social Media Hate Speeches in Amharic and Afaan Oromo

Afaan Oromo	Amharic
Biyya diiguuf Tigir** ta'uun gahaadha	ሃገርን ለማፍረስ ት**ሬ መሆን በቂ ነዉ.
Diqaal** biyya maalii qab*da asii	ድቃ* ከዚህ ምን ሃገር አ*ክ
Sanyii qur**i	የ ቆማ* ዘር

The posts are targeting some people just because of their ethnicity, religion, disability, or gender. This problem is challenging and makes serious conflict when peoples start sharing the posts. After observing the problem of hate speech propagation on social media, researchers focused on this area to discover an automatic hate speech detection mechanism.

Besides the success of deep learning algorithms on tasks related to image classification and object recognition, the abundance of natural language data and advancement in representing such data makes deep learning ideal for texts and speech processing which makes it now a state-of-the-art in many NLP tasks(Hassan & Mahmood, 2017). Several researchers focused on the area of hate speech detection for social media texts in past recent years and proposed different techniques for hate speech detection. Some works proposed hate speech detection systems by using machine learning techniques (Aulia, 2019; Mossie & Wang, 2018; Nayel & Shashirekha, 2019), and deep learning approaches(Baweke, 2020),(Gambäck & Sikdar, 2017),(Sutejo & Lestari, 2019).

But, most of the works are done for a mono language text. Further, the study explored until now on multilingual hate speech detection did not used the attention mechanism, while it brought the advantages of giving high weight for the useful keywords in the dataset.

As far as we know, there is no research conducted on bilingual social media hate speech detection for Afaan Oromo and Amharic languages. There is no dataset available to work on it for the Afaan Oromo indeed there is no bilingual dataset for Afaan Oromo and Amharic languages. This research proposed bilingual hate speech detection for Afaan Oromo and Amharic language texts on social media by using deep learning approaches. It involves preparing a bilingual dataset and developing deep learning models. The study aimed to develop a model to classify Afaan Oromo and Amharic social media texts as hate or free.

1.4 Research Questions

The following research questions are addressed and answered by this research work:

RQ1: Among the selected deep learning algorithms, which algorithm can achieve a better performance for the Afaan Oromo and Amharic bilingual text hate speech detection?

RQ2: How the performance of Afaan Oromo and Amharic bilingual hate speech detection can be improved?

RQ3: To what extent can we achieve hate speech detection using a deep learning model?

1.5 Objectives of the Study

1.5.1 General Objective

The general objective of this study is to develop and assess different deep learning models for Afaan Oromo and Amharic languages bilingual social media text hate speech detection.

1.5.2 Specific Objectives

- ☞ To prepare a bilingual hate speech dataset using Afaan Oromo and Amharic language texts.
- ☞ To develop a bilingual hate speech detection model by training deep learning algorithms on the prepared bilingual dataset.
- ☞ To evaluate the performance of different deep learning models and identify the best performing model.
- ☞ To tune hyperparameters for the selected deep learning model for further improvement.
- ☞ To develop a prototype using the best-performing model to demonstrate the model.

1.6 Scope and Limitations of the Study

1.6.1 Scope of the Study

The main goal of this study is to develop a deep learning model for detecting Afaan Oromo and Amharic hate speech texts on social media. There is no Afaan Oromo data to prepare a bilingual dataset. The Afaan Oromo dataset is prepared from scratch by scraping Facebook pages. This study includes developing bidirectional recurrent neural network models and using an attention mechanism by training models using the prepared bilingual dataset. Also, this work includes applying k fold cross-validation and tuning hyperparameters using different trials. The proposed bilingual hate speech detection classifies the dataset into two classes as Hate and Free.

1.6.2 Limitations of the Study

This study comes across limitations on a different phase of the research process. To build the dataset for this study we have only available Amharic dataset and Afaan Oromo data is collected from Facebook posts and comments and prepared from scratch. Due to the complexity of the task of data

annotation and also we have a limited number of annotators, we are unable to create a large dataset. The other big issue is concerning the computational resource which prohibited us from carrying out more GPU-intensive processes like automatic hyperparameters tunings, as we got a low-end GPU. Also, the classification of aspects of hate speech like political, religious, ethnic, and gender is not part of the study. Lack of time and dataset limited us also from developing models for non-textual data such as audio, and videos for hate speech detection.

1.7 Significance of the Study

The study proposed a deep learning-based bilingual hate speech detection for Afaan Oromo and Amharic texts on social media. The model classifies Afaan Oromo and Amharic texts as hate, and free. Deployment of this system has many significances. First, Detecting and removing hate speech is important for online communities for maintaining safe environments for their users. But, effectively detecting social media hate speech is difficult to do manually and it requires having an automatic detection system. The development of the hate speech detection model plays important role in simplifying the task of tracking online hate speeches. The development of a hate speech detector has also a vital role in the development of AI chatbots, public sentiment analysis, and managing contents of online discussions to identify and block hateful agendas. Also, it is helpful for organizations of social media platforms like Facebook and Twitter which are being challenged by the problem. They can use it as part of their extension so that hateful texts can be blocked. It has a great role to advance the value of democracy and dignity in the country since the hate speech detection system opposes hateful posts towards targeted groups or individuals.

1.8 Organization of the Thesis

The research paper is organized into seven chapters in the following ways:

Chapter one discussed above, includes the introduction of the study, the motivation, and the statement of problems, the research questions that would be answered in the proposed solutions, the scope and limitation, the objective of the study, and the significance of the study.

Chapter two, Presents the literature review and related works on hate speech detection, the definition of hate speech, social media definition, techniques used in hate speech detection: feature extraction, deep learning models, it discussed an overview of the Afaan Oromo and Amharic languages and finally, it discussed the related work.

Chapter three, Discusses the research methodology used in this research work, methods used to build the dataset, methods used to develop detection models which are preprocessing, feature extraction methods, and deep learning algorithms, and finally, it presents the evaluation methods.

Chapter four, Chapter describes the proposed solution for bilingual Afaan Oromo and Amharic language hate speech detection, which describes the overall architecture of the system, the proposed text preprocessing, feature representation technique, deep learning models architecture design, and model evaluation techniques.

Chapter five, Presents the implementation of the proposed solution for bilingual hate speech detection for Afaan Oromo and Amharic languages. It contains required tools, and the implementation of preprocessing, feature extraction implementation, model implementations.

Chapter six, Presents the result gotten from the experimentation with different deep learning models. Also, it presents the result of with and without using word2vec. Besides that, this chapter discusses and analyses the major results obtained from the experimentation.

Chapter seven, in this chapter we conclude the research and give something useful or necessary insight into the recommendations and valuable suggestions. The impacts of this research are also described in this final chapter.

CHAPTER TWO

LITERATURE REVIEW AND RELATED WORKS

This chapter reviews relevant literature to further comprehend the concept and investigate the research problem. The review defines hate speech, application areas of hate speech detection systems, and existing techniques used to detect hate speech. We start, by defining hate speech and social media. After that, the chapter presents a review of hate speech on social media, an overview of the Afaan Oromo and Amharic languages, techniques used in hate speech detection, and related works on social media hate speech detection.

2.1 Hate Speech

Many definitions are given for the term hate speech by the law of different countries and also there are definitions given by social media organizations. The term hate speech refers to any kind of expression that promotes hatred or attack against a targeted people because of who they are. But, there is no agreed-upon definition for the term hate speech. Therefore, it is difficult to identify whether texts contain hate speech or not. There are some challenges making hate speech detection difficult, even for humans. Those are, The creation of the dataset is not trivial because of its small number of hate speech compared to all social media data; Hate speech is user-generated content with a wide variety of writing styles (Aulia, 2019). Recently, hate speech is a crime that has been growing especially in online communications. The internet, social media platforms, and the increased willingness of people to express their opinions online contributing to the propagation of hate speech fastly (Fortuna & Tec, 2020).

2.1.1 Definition of Hate Speech

2.1.1.1 Definition From Social Media Platforms

Different social media platforms have their definitions for the term hate speech. But, most of the definitions are similar regarding their contents. Facebook, Twitter, and Youtube are the most common platforms that people use to post and share hate speech. Definition of the term hate speech given by Facebook, Twitter, and Youtube is given below:

Definition 1: Twitter³, “You may not promote violence against or directly attack or threaten other people based on race, ethnicity, national origin, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease. We also do not allow accounts whose primary purpose is inciting harm towards others based on these categories.”

³ <https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy>.

Definition 2: Facebook⁴, “Objectionable content that direct attack on people based on what we call protected characteristics such as race, ethnicity, national origin, religious affiliation, sexual orientation, caste, sex, gender, gender identity, and serious disease or disability. We also provide some protections for immigration status. We define attack as violent or dehumanizing speech, statements of inferiority, or calls for exclusion or segregation.”

Definition 3: YouTube⁵, “Hate speech is not allowed on YouTube. We remove content promoting violence or hatred against individuals or groups based on any of the following attributes: Age, Disability, Ethnicity, Gender, Nationality, Race, Immigration Status, Religion, Sex, Sexual Orientation, Veteran Status. We refer to Gender, Sex, and Sexual Orientation definitions mindful that society’s views on these definitions are evolving”.

2.1.1.1 Definition From Other Sources

There are also definitions for the term hate speech which is given by different organizations, and some of them are given below:

Definition 4: UN, International Committee on the Elimination of Racial Discrimination, “Hate Speech shall be understood as speech covering all forms of expression which spread, incite, promote or justify racial hatred, xenophobia, anti-Semitism or other forms of hatred based on intolerance, including intolerance expressed by aggressive nationalism and ethnocentrism, discrimination and hostility against minorities, migrants and people of immigrant origin”(Alhogbi, 2017).

Definition 5: American Convention on Human Rights (ACHR), “Hate speech can be defined by both its intent and its target. Concerning intent, hate speech is a speech designed to intimidate, oppress or incite hatred or violence. This speech must also be targeted directly against a person or group based on characteristics like race, religion, nationality, gender, sexual orientation, disability or another group characteristic”(Bertoni, 2006). For this study, we used the definition of hate speech presented on Ethiopian Hate Speech and Disinformation Prevention and Suppression Proclamation No.1185 /2020(ZEWUDIE, 2019). The definition is as follows:

Hate Speech: “Hate speech is the speech that intentionally promotes discrimination, hatred, or attack against a discernable group of identity or person, based on race, ethnicity, gender, religion or disability”.

Based on the definitions of hate speech above, hate speech can be differentiated from other kinds of speech in the following way:

- ☞ Hate speech is aimed to incite violence or hatred.

⁴ https://www.facebook.com/communitystandards/objectionable_content.

⁵ <https://support.google.com/youtube/answer/2801939%0Ahttps://support.google.com/youtube/answer/2801939?hl=en>.

- ☞ Hate speech includes promoting discrimination against a group or person based on their identities such as religion, gender, and ethnicity.
- ☞ Hate speech has a specific target which is characteristics owned by a person or group such as ethnicity, religion, gender, or disability.
- ☞ Hate speech is to attack or diminish based on protected identity specific targets.

In addition to those definitions, the guideline for details of indicators of hate speech is given in appendix A for simplifying the task of data annotation.

2.3 The Social Media

Social media is a platform people use to discuss their issues and opinions. Social media are platforms that allow people to share or exchange ideas, information, videos, images, texts, and even more with each other through a particular network(Akram & Kumar, 2017). Nowadays social media plays a great role in simplifying tasks such as electronic mails, online business, and education or tutorials online. Not only advantages but also social media has also disadvantages that can affect society negatively. The opportunity created by social media platforms allowing people to post and share different illegal pieces of information may negatively affect social media users and also other societies. Social media platforms like, Facebook, Twitter, and YouTube are being used as a tool for sharing hate speech and disinformation, which has a direct negative effect on the normal life of society.

2.4 Hate Speech on Social Media

The nature of social media allows people to create and share content or to participate online. It created an opportunity in which people can post and share their opinions and feelings online. Using this opportunity people can post and share illegal content such as hate speech, cyberbullying, and offensive speech. These online platforms are often exploited and misused to spread content that can attacks or diminishes, that incites violence or hate against targeted persons or groups. The nature of social media platforms for security and privacy, enables their users to hide their true identity behind the screen and express or spread hateful content than they might otherwise (Mossie & Wang, 2018). Recently, hate speech is considered a serious problem by different country authorities. Hate speech on social media is adverse not just to the wellness of users, also it has an impact on an open egalitarian society. Nowadays, online platforms such as Facebook and Twitter have connected billions of people and allowed them to share their ideas and opinions instantly. That being said, there are several intimidating consequences as well such as online harassment, trolling, cyber-bullying, and hate speech(Baweke, 2020).

In Ethiopia, hate speech on social media becoming a big problem with the increase of social media users in the country. Lack of legal law or recommendation that indirectly defines or tackles hate speech makes the problem more difficult. But, there is a law that is indirectly used for hate-related issues, which is anti-terrorism law, the law forbids “the use of any telecommunication network or apparatus to broadcast any terrorizing information” or “obscene message”, this includes the use of any social media platform and other forms of communication platform to disseminate terrorizing message(Kenenisa & Melak, 2019). The subjecting violators to a prison sentence of up to eight years. However, the law has been used to limit messages or speech that criticizes government policy and officials. This law has got a reaction from the national and international organization and academic community because of the use of the law contradict freedom of speech of human right law. Currently, law-maker, government officials, and politicians in Ethiopia have been aware of hate speech on social media, and they are looking for a solution to tackle this problem. The country lawmakers are drafting new hate speech and fake news laws which will be law soon.

2.5 Techniques Used for Hate Speech Detection

Since the problem of hate speech detection is related to text classification tasks, many researchers applying machine learning and deep learning techniques for the problem of online hate speech detection.

2.5.1 Machine Learning Techniques

Machine learning allows machines to learn and experience from data and perform tasks skillfully. Machine learning algorithms require data to learn, the discipline must have a connection with the discipline of the database. Due to the nature of some problems or because digital information is increasing in tremendous amounts, algorithms that can learn and adapt the patterns from large data are essential. The two main problems that require machine learning algorithms because of their capability to learn and improve from experience are tasks that are too complex to program. The other problem is adaptability requiring the ability to familiarize with user data. ML algorithms find the patterns in the data to better understand the data and help users in decision-making in their day-to-day activities. There is supervised and unsupervised machine learning category.

Supervised Learning: This type of ML is trained using the data with label. It uses a labeled dataset that contains matched sets of observed inputs, X’s, and the associated outputs, Y’s. Machine learning algorithm is applied to the dataset to infer the patterns between the inputs and outputs.

Unsupervised Learning: Instead of using labeled data, we need to allow the model to work on its own to discover information in the case of an unsupervised approach. It is used to train a model using a

dataset that only has inputs and analyzes to make sense and to structure or group the data. It uses data that are unlabeled and uncategorized to recommend to users by creating some structure to the data to the group. It draws inferences from the dataset by using input data without output data. On the other hand, it does not have labeled outputs, so its goal is to infer the natural structure present within a set of data points(Wiratunga, 2000).

2.5.2 Deep Learning Techniques

Deep learning methods use neural networks to automatically learn the multi-layers of features from the given data. It is a subset of machine learning approaches that attempts to learn the layered model of inputs. It enables computational models composed of different processing layers to learn representations of data with multiple abstraction levels. Deep learning algorithms are capable of discovering complex structures from large datasets using a backpropagation algorithm to update their internal parameters(Baweke, 2020). Neural network models show a promising future in text classification tasks. Deep learning models depend entirely on the ANN but with extra depth(Al-Hassan & Al-Dossari, 2019).

2.5.2.1 Neural Networks

Neural networks or artificial neural networks are pattern recognizable models fascinated by the working principle of the human brain. Its working strategy seems like the way that neurons transfer signals to one another in the human brain. ANN is a subset of machine learning and is at the heart of deep learning algorithms. An artificial neural network(ANN) contains the input layer, an output layer, and one or more hidden layers.

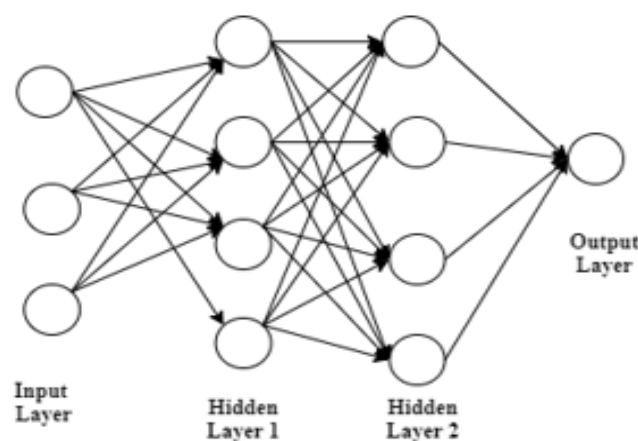


Figure 2.1 Artificial Neural Networks

Recently, deep learning algorithms got a high priority in the task of text classification due to their high accuracy. The following deep learning algorithms are used in text classification:

2.5.2.2 Recurrent Neural Network (RNN)

RNN is a type of ANN that uses sequential data or time-series data where the output from the previous step is used by the current step. It is designed in such a way that it can recognize the sequential characteristics of data and thereafter using the patterns to predict the coming scenario. Therefore, RNNs should remember the information about a sequence and have a hidden state that remembers some information about the sequence.

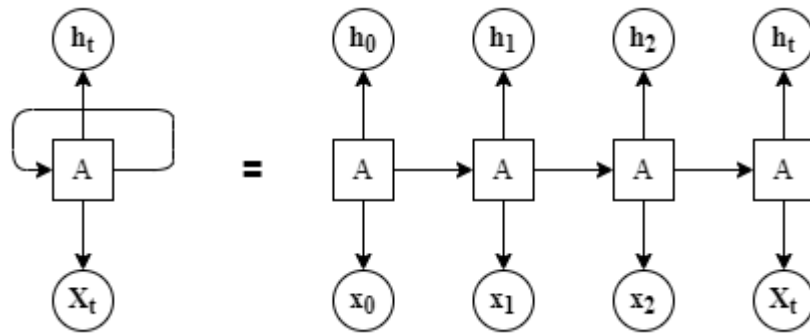


Figure 2.2 Recurrent Neural Networks

Given input state x_t , previous state h_{t-1} , new hidden state, and output at time step t are computed as:

$$h_t = \sigma_h (W_h x_t + U_h h_{t-1} + b_h)$$

$$y_t = \sigma_y (W_y h_t + b_y)$$

where:

- ☞ x_t is input vector at a time step, h_t is hidden layer vector, y_t is output vector at time step t .
- ☞ W, U, b are parameter matrices and vectors.
- ☞ σ_h, σ_y are activation functions.

In RNN hidden layers are recurrent layers, where every neuron in the hidden layer is connected. The hidden layer takes input from both input layer x_t and the hidden layer from the previous state h_{t-1} . Unlike feed-forward neural networks which accept fixed-size vectors as input and give fixed size-vector output, recurrent neural networks are designed for modeling sequences and they are capable of remembering past information.

However, recurrent neural networks(RNN) suffer from the problem of short-term memory. If the length of the input sequence is very long, they will have a hard time carrying the information from the earlier timesteps to later ones. LSTM and GRU solved this problem.

2.5.2.3 Long-short Term Memory(LSTM)

LSTM's are introduced by Hochreiter & Schmidhuber (1997). Long-short term memory is capable of learning long-term dependencies and they are special kinds of RNN networks. LSTM's are designed to avoid the problem of long-term dependency. There are three gates of LSTM: forget gate f_t , input gate i_t , and output gate o_t . The forget gate(f_t) is responsible for deciding what information to throw away from the cell state by looking at h_{t-1} and x_t , and gives an output a number between 0 and 1 for each number in the cell state in which 1 represents “keeping this completely” and 0 represents “ignore this”, and this is the first step in LSTM. The preceding operation is to decide what information to store in the cell state. Then, the input gate layer is responsible to decide which values to update. After that, a vector of new candidates $\tilde{C}_t$ is created by the tanh layer. Then, the old state c_{t-1} into c_t and that is the new cell state. After all, the output is based on the cell state but will be a filtered version. The sigmoid layer is responsible to decide what part of the cell to output.

2.5.2.4 Bidirectional Long Short Term Memory

BiLSTMs are an extension to typical LSTMs that are designed to capture information on a sequential dataset and maintain contextual features from the past and future. This network processes input sequences in both directions with two sub-layers to account for the full input context. The two RNN sub-layers are responsible to compute forward and backward hidden sequences(Mousa & Schuller, 2017). Then these are combined to compute the output sequence. It can able to exploit context in both directions. BiLSTM combined the forward and the backward hidden layers, which can perceive both the previous and subsequent contexts. However, LSTM only takes advantage of historical context. The architecture of BiLSTM is as shown below:

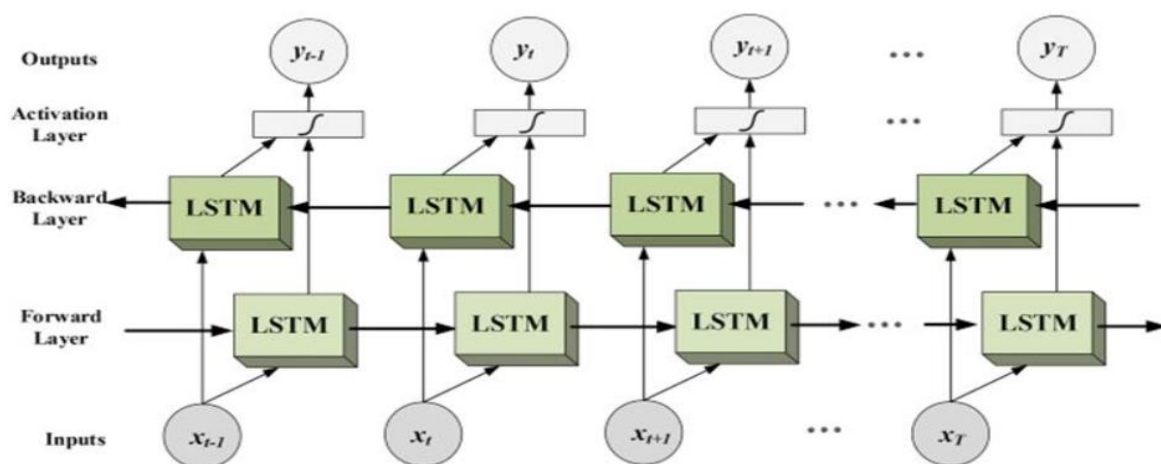


Figure 2.3⁶ Architecture of Bidirectional LSTM

⁶ <https://www.i2tutorials.com/deep-dive-into-bidirectional-lstm/>

2.5.2.5 Gated Recurrent Unit(GRU)

It is similar to LSTM but, it consists of two gates, which are the update gate and reset gate. These two gates are trained to filter out any inappropriate information and keeping essential information. The gates are containing values between 0 to 1 which can be multiplied with the input data or hidden state. In each case, 0 value in the gate vectors shows the corresponding data in the input or hidden state is irrelevant. on the other side, 1 value in the gate vector indicates that the corresponding data is essential and will be used. It has not a cell state rather it uses the hidden state to transfer information.

2.5.2.6 Convolutional Neural Network(CNN)

A convolutional neural network works in a ‘feed-forward strategy and connections between nodes do not form any cycle. Generally the model is used in the area of computer vision, however, they have shown promising results when applied to various NLP tasks as well. CNN’s are initially designed for the task of image processing that can make use of the 2D structure of image data through convolution layers, where each computation unit responds to a small region of input data. However, CNN has been getting great attention for text classification recently. Firstly words in a sentence are converted to low-dimensional word vectors through different techniques such as word embedding to use the model for text data. There is an input layer, convolution layer, and max-pooling layers in CNN. The input layer is used to embed words into low-dimensional vectors. After that, the operation of convolutions over the embedded word vectors using multiple filter sizes is performed by the convnet layer.

But, in the text classification modeling understanding the context of the data is necessary to provide a good prediction. For example, in case of hate speech detection, the model is expected to understand contexts of words from training data so that it can perform well on the new test data. But, since model is a feed-forward neural network applied on one-hot encoded vectors representing the dataset and CNN cannot model the context of words while RNNs are good in modeling text sequence.

2.5.2.7 Attention Mechanism

Deep learning approaches have become the driving force for NLP tasks. The most common neural architectures for natural language processing tasks are RNN and CNN. Recurrent neural networks are good for modeling long-term dependencies over input texts, and CNN can carry out high and efficient parallel computation(Liu et al., 2020). Although the encoder-decoder network performs quite well in NLP tasks like text classification, it is difficult for the model to maintain its performance as the input gets larger. Encoder-Decoder model encodes the input sequence to one fixed-length vector from which to decode the output at each time step.

But, when decoding long sequences it becomes a problem because it makes difficult for the neural network to cope with long sentences, especially those that are longer than the sentences in the training corpus (Agrawal, 2021).

The attention mechanism has spawned the rise of so many recent breakthroughs in natural language processing (NLP), including the Transformer architecture and Google's BERT. Attention Mechanism attempts to concentrate on a few relevant things while ignoring others in deep neural networks.

2.5.3 Feature Extraction Techniques

2.5.3.1 Bag of Words⁷

A bag-of-words is a representation of text that defines the occurrence of words within a document. A set of vocabulary is prepared based on the words which are in the training data, rather than a pre-defined set of words, like dictionaries. The steps involved in creating the BOW model for a piece of text are as follows:

1. Tokenize the text and store the tokens in a list.
2. Create a vocabulary out of the tokens.
3. Count the number of occurrences of tokens in each sentence and store the count.
4. Create a bag of words model by converting the text into vectors with a count of each word from the vocabulary.

However, in this technique, the word sequence doesn't consider, additionally, its syntactic and semantic content. Because of this, it can happen misclassification if the words are used in different contexts. The bag of word model focuses on the word count to represent a sentence. So, the order of the words in a sentence is not considered under the BOW model.

2.5.3.2 Term frequency-inverse document frequency (TF-IDF)⁸

TF-IDF can be defined as the calculation of how relevant a word in a series or corpus is to a text.

Term Frequency (TF): Given the document d , frequency represents the number of occurrences of a given word t . Hence, we can see that it becomes more relevant when a word appears in the text, which is rational.

$$TF(t) = \frac{\text{Numer of times } t \text{ appears in a document}}{\text{Total number of the term in the document}} \quad 2.1$$

⁷ <https://www.journaldev.com/46591/bag-of-words-model-in-python>

⁸ <https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/>

Inverse Document Frequency (IDF): The main operation of this IDF is it tests how relevant the word is. The key aim of the search is to locate the appropriate records that fit the demand. It is not only possible to use the term frequencies to measure the weight of the term in the paper since tf considers all terms equally significant.

$$IDF(t) = \log_e\left(\frac{\text{Total nuber of documents}}{\text{number of document with term } t \text{ in it}}\right) \quad 2.2$$

The term frequency is only the number of instances of a term in a single document; although the frequency of the document is the number of separate documents in which the term appears, it depends on the entire corpus. Now let's look at the definition of the frequency of the inverse paper. The IDF is the number of documents in the corpus separated by the frequency of the text.

Part-of-speech tag (POS): It is the approach used to improve the importance of the context and detect the role of the word in the context of a sentence. It comprises detecting the class of the word, such as personal pronoun, Verb non-third person singular present form, Adjectives, Determiners, Verb base forms. Generally, All the above-listed techniques fail to capture the context and semantics.

2.5.3.4 Word Embedding

Word embedding is a feature learning technique by which words from the vocabulary are mapped to real number vectors through capturing the contextual hierarchy. It is a way of representing texts where words that have the same meaning have a similar representation. Meaning that two similar words are represented by almost similar vectors that are very closely placed in a vector space. These are essential for solving most Natural language processing problems(Mujtaba, 2020). It represents words in a coordinate system which means related words, primarily based totally on a corpus relationship, are located nearer collectively.

Word2Vec is one of the most popular techniques to learn word embeddings developed by Tomas Mikolov in 2013 at Google. Word embeddings became the latest in natural language processing after the release of the word2vec toolkit and sparked a huge amount of interest in the topic. The Word2vec model takes text corpus as input and gives a set of vectors called feature vectors representing words in the text. The main feature and function of word2vec is enabling words having similar contexts to have similar embeddings. Word2vec has two architectural models known as CBOW (Continuous bag of words) and the Skip-gram model.

Continuous Bag of Words (CBOW):

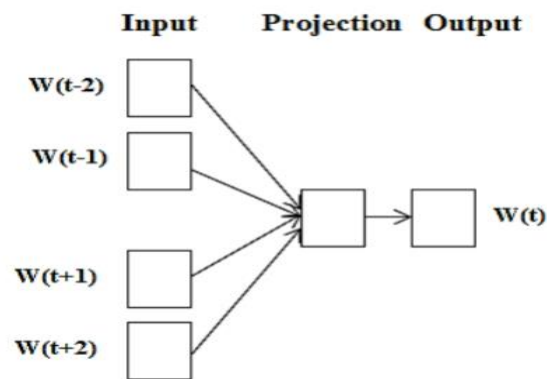


Figure 2.4 Continuous Bag of Words Architecture

CBOW given in figure 2.5 above, tends to find the probability of a word occurring in a context. Therefore, it generalizes over all the different contexts in which a word can be used. It predicts the probability of a word to occur given the surrounding words. The representation of vectors a given word is determined by its neighboring words, from where the semantics of the word can be captured. The output layer of the architecture holds the vector representation of a word after the model is trained over the average vectors of accompanying words.

Skip-gram Model: The model usually tries to achieve the reverse of the CBOW model. It predicts the source context words taking a target word as input. It can be concluded that the target is fed to the input and the output layer is replicated multiple times to accommodate the chosen number of context words. The error vector from all the output layers is added to adjust weights via a backpropagation method⁹.

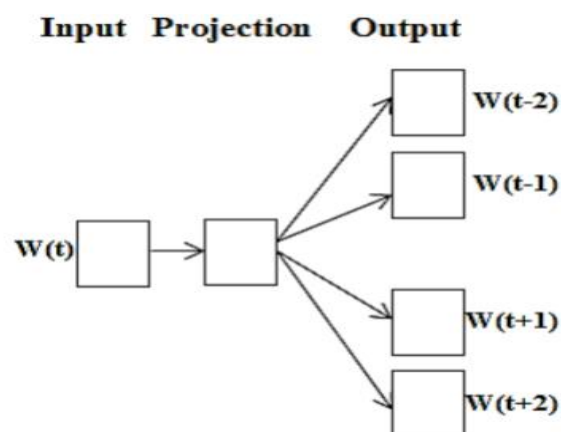


Figure 2.5 Skip-gram Model Architecture

⁹ <https://www.guru99.com/word-embedding-word2vec.html>.

GloVe (Global Vectors for Word Representation): Glove uses the advantage of the skip-gram model of word2vec and collaborative filtering algorithms aka matrix factorization(YARED YENEALEM AKLILU, 2019).. It is an unsupervised learning algorithm for word representation in vector space. It generates a word-word co-occurrence matrix from the overall document used for training and maps each word into a semantically appropriate point in space where the distance between similar words is kept small. In practice, both GloVe and Word2Vec are used to convert text into embeddings, and both show comparable performances.

2.6 Afaan Oromo Language

Afaan Oromo is one of most widely spoken and used in Ethiopia and neighboring countries like Kenya and Somalia. Afaan Oromo language is being used as an official language in Oromia regional state. Basically the language is used by Oromo people, who are the largest ethnic group in Ethiopia(BIRHANE TESFAYE, 2017). Latin-based alphabet known as Qubee has been adopted and became an official script of Afaan Oromo since 1991(Wubshet & Ejigu, 2014).

2.6.1 Writing system Of Afaan Oromo

The writing system of Afaan Oromo is nearly phonetic since it is written the way it is spoken, i.e. one letter corresponds to one sound. The Afaan Oromo language uses the Qubee (which is the Latin alphabet) which was formally adopted in 1991. Qubee has 33 consonants, seven of them are combined consonant letters: **ch, dh, ny, ph, sh, ts** and **zh**. The combined consonant letters are known as **qubee dachaa'**. The Afaan Oromo language has five short and five long vowels. The Afaan Oromo alphabet is characterized by capital and small letters like the English alphabet. vowels are sound makers in Afaan Oromo, as in the English language, and do stand by themselves(Genemo, 2015).

2.6.2 Word Categories of Afaan Oromo

Words are the basic unit of a language that has meaning and can be spoken or written. The Afaan Oromo language words are composed of two parts: the root (base morpheme), which generally consists of basic sound and provides the basic lexical meaning of the word, and the pattern, which consists of prefixes and/or suffixes and gives grammatical meaning to the word. For example, the root 'bar' combines with the pattern '-e' gives bare (learned), whereas the same root combines with the pattern '-te' gives barte (she learned). Afaan Oromo words can be placed into five grammatical categories: nouns, verbs, adverbs, adjectives, and adpositions (Ababa, 2020).

Table 2.1 Afaan Oromo word class

Gochima	Verb
Maqaa	Noun
Ibsa Maqaa	Adjective
Dabalgochima	Adverb
Raajeffannoo	Interjection
Walingaa	Conjunction
Durgala	Preposition
Maqdhaal	Pronoun

2.6.3 Afaan Oromo Sentence Structure

Afaan Oromo uses a subject-object-verb (SOV) structure, unlike English which has (SVO) structure. For instance, in the Afaan Oromo sentence “Gaaddiseen *barattuu* dha”, “Gadise” is the subject, “*barattuu*” is the object and “dha” is the verb. The translation of the sentence in English is “Gadise is a student”. There is also a difference in the formation of adjectives in Afaan Oromo and English. In the Afaan Oromo language, adjectives follow a noun or pronoun; their normal position is close to the noun they modify while in English adjectives usually precede the noun. For instance, in “ ilma gaarii” (good boy), gaarii (adjective) follows ilma (noun)(BIRHANE TESFAYE, 2017).

2.6.5 Afaan Oromo Punctuation

Other than apostrophes, punctuation marks used in both Afaan Oromo and English languages are the same and used for the same purpose. In Afaan Oromo language, Apostrophe mark (,) is used in writing to represent a glitch sound known as “hudhaa” while it is used to show possession in English language. The Apostrophe mark (,) in Afaan Oromo plays important role for reading and also writing system. Punctuation is placed in the text to make meaning clear and reading easier. Afaan Oromo follows the same punctuation pattern used in English and other languages that follow the Latin writing system.

Table 2.2 Afaan Oromo' Language Punctuation Marks

Tuqaa (Full stop (.)):	Used at the end of a sentence and in abbreviations.
Mallattoo Gaaffii, (Question Mark (?)):	Used in an interrogative or at the end of questions.
Raajeffannoo (Exclamation Mark (!)):	Used at the end of command and exclamatory sentences.
Qoodduu (Comma (,)):	It is used to separate listing in a sentence or to separate the elements in series.
Tuq-lamee (Colon (:)):	Used to separate and introduce lists, clauses, and quotations along with several conventional uses, etc.

2.7 Amharic Language

Not only Afaan Oromo but also Amharic language (አማርኛ) is another language having large speakers in our country. In addition to having a large audience, the language is the working language in the country officially. It comes from northern and central Ethiopia by the peoples of Amhara. The Amharic language uses an alphabet called fidäl (ፊደል), inherited from the Geez (Tiruneh, & Mulat Getaneh, 2020). Also, it is the 2nd most widely spoken Semitic language after Arabic and it is a state language in Ethiopia.

2.7.1 Amharic Character Representation

The Amharic language employs Geez characters. There are only consonants available in the Geez script, while the subsequent variants of the characters represent phoneme pairs of consonant-vowel. In Amharic language characters are formed by consonant-vowel combination like in Geez. 34 basic consonant and vowel characters are there which have six additional different forms in the Amharic language. The 34 basic characters are ሀ, ለ, ሐ, መ, ሠ, ረ, ሰ, ሸ, ቀ, ቦ, ተ, ቸ, ኀ, ነ, ኘ, አ, ከ, ኸ, ወ, ዐ, ዘ, ዠ, የ, ደ, ጀ, ገ, ጠ, ጪ, ጰ, ጹ, ፀ, ፈ, ፒ. When one of the remaining six vowels follows these basic characters, they change their shape into some predefined forms (Argaw & A, Abebe, 2019). All the above 34 basic characters are not unique in their pronunciation or sound representation. Some characters in the Amharic language represent similar sounds, even if all of them have their usage. The sets {ሀ, ሐ, ኀ}, {ሠ, ሰ}, {አ, ዐ} and {ጰ, ፀ} represent similar sounds. This adds complexity to the Amharic writing scheme.

2.7.1 Challenges in Amharic Language Writing Scheme

Amharic writing scheme has some issues that are difficult to process Amharic text. One of these challenges is the redundancy of characters used in Amharic, more than one character to represent 19 the same sound. The various forms have their meaning in Ge'ez, but there is no clear rule that shows their purpose in Amharic. The problem of the same sound with various characters is not only observed with core characters but also exhibited in the same order of characters (Kenenisa & Melak, 2019). A word formed by using those characters has the same meaning. For example, the word 'sun' could be written in a different way like ጸሀይ, ፀሀይ, ጸሃይ, ፀሃይ, ጸሐይ, ፀሐይ, ፀሓይ, ፀኅይ, or ጸኅይ. This is why we need to apply character normalization for the Amharic language.

2.8 Related Works on Hate Speech Detection

This section presents a comprehensive review of basic related works to the area of hate speech detection on social media to clearly understand the general technique, method, and result of existing studies.

The work by (Kenenisa & Melak, 2019) proposed a hate speech detection model for Amharic language using machine learning and text-mining feature extraction techniques. They collected data from selected Facebook public pages and manually labeled them into three classes and then converted them into binary classes to build binary and ternary datasets. SVM, NB, and RF models trained using the whole dataset with the extracted feature based on word unigram, bigram, trigram, combined n-grams, TF-IDF, combined n-grams weighted by TF-IDF and word2vec for both datasets. The models were evaluated using 5-fold cross-validation. The models based on SVM with word2vec perform better in classification results with a 73% F1-score, and it shows slightly better performance than the NB and RF models for both binary and ternary models. However, the researchers were limited to using only machine learning algorithms and they developed the models only for the Amharic language. Deep learning models can be implemented for better classification performance.

(Baweke, 2020) proposed Amharic text hate speech detection using deep learning approaches. The author collected data from Facebook and Twitter and labeled them into four classes as hate, offensive, both (hate and offensive), and neutral. Word embedding using Keras and Word2Vec embedding used as feature extraction. For the experiment LSTM, CNN, BiLSTM, combined CNN-LSTM, and GRU models were trained using the whole dataset. The researcher used (80,20) train-test split with performance metrics of precision, recall, and f1-score for model comparison. The researcher applied data augmentation techniques and formed two datasets (original and augmented). Five different models were developed using both datasets and the model based on BiLSTM with word2vec achieved better performance than the other models for both augmented and original datasets with an accuracy of

88.89% and an f1-score of 89% for the original dataset. the authors are limited to only using RNNs, BiRNNs, and CNN models, while attention mechanisms can be implemented, also they didn't use cross-validation technique which helps to minimize the problem of overfitting. They developed the models only for the Amharic language.

(Gaydhani et al., 2018) presented a mechanism for developing a model for the detection of hate speech and offensive language on Twitter by using Naive Bayes, Logistic Regression, and Support Vector Machines with n-gram features weighted with TF-IDF values using a publicly available Twitter dataset. The authors used a very large number of datasets combined from three different publicly available datasets from Crowd flower containing tweets that have been manually classified into Hateful, Offensive, and Clean. Logistic Regression outperformed other models with the optimal n-gram range 1 to 3 for the L2 normalization of TF-IDF and the model achieved 95.6% accuracy.

The work presented by (Sutejo & Lestari, 2019) developed models to detect hate speech in the Indonesian Language from input text and speech by using a deep learning approach. The authors used both textual and acoustic features and compare their accuracies. From the experiment, better results for hate speech detection are obtained using only textual features is better than that of using acoustic features and both combined features models. The best model using textual feature obtained an F1-score of 87.98% which is higher than the model of using acoustic feature only (F1-score 82.5%) and the model of using acoustic and lexical features (F1-score 86.98%)

(Tesfaye & Tune, 2020) Developed recurrent neural network models for automated hate speech post-detection for Amharic posts on Facebook. The long short-term memory and gated recurrent unit with word n-grams for feature extraction were used by the authors. They used word2vec to represent each unique word by vector representation. The authors split the dataset into a train set, validation set, and test set of 80%, 10%, 10% respectively for the experiment. LSTM based RNN with Batch size 128, learning rate 0.001 with 0.5 dropouts, and RMSProp optimizer achieved an accuracy of 97.9% to classify posts as free or hate speech by training with 100 epochs. Despite they achieved good accuracy, they are limited to develop a model for a monolingual dataset. further, other deep learning models such as attention mechanism and bidirectional recurrent neural networks can be tested instead of using only LSTM and GRU.

(Del Vigna et al., 2017) proposed an Italian online hate campaign on the social network. They used data collected from Italian public Facebook pages. The dataset is annotated into three classes as no hate, weak hate, and strong hate, and by merging weak and strong hate as hate they form the second dataset. The author design and implement two classifiers algorithms for the Italian language, SVM and LSTM algorithm

leveraging morpho-syntactical features, sentiment polarity, and word embedding lexicons. Conducting two different experiments with both datasets that a least 70% of the annotator agreed on the class of the data. F1-score of 80% is achieved by using SVM and LSTM achieved an f1-score of 79% for binary classification. For the ternary classification SVM and LSTM achieved f1_score of 64% and 60% respectively. They used only the conventional SVM and RNN model LSTM, while more deep learning models can be implemented to improve the classification performance.

(Aljarah et al., 2020) proposed Cyberhate speech detection based on Arabic context over the Twitter platform, by applying NLP and machine learning techniques. The work focused on a set of tweets related to sport, racism, terrorism, journalism, sports orientation, and Islam. The processed dataset is experimented with using Decision Tree (DT), Support Vector Machine (SVM), Naive Bayes (NB), and Random Forest (RF). In their experiment, RF with TF-IDF and profile-related features achieved a better result with an accuracy of 91.3%. Since hate as a term is subjective and can be expressed in a wide range of areas not restricted to the sport, religious or racial issues they recommended further work for the more generalized dataset and effective detection models.

Likewise, the deep learning approach for automatic cyber hate speech detection on Twitter is presented by (Faris et al., 2020). The dataset for the study was collected from Twitter and the collected data captures different hate expressions in the Arabic region. The author's used word embedding mechanisms for feature extraction. The hybrid of CNN and LSTM network is implemented for model development. The proposed approach aimed to classify tweets as hate and normal and achieved promising results, 66.564%, 79.768%, 65.094%, 71.688% regarding the accuracy, recall, precision, and F1 measure respectively. The study is limited to binary classification while it is important to distinguish offensive expressions from hate speech. The study also recommended a more standardized dataset and high-performance deep learning approaches.

(Aluru et al., 2020) proposed multilingual hate speech detection using deep learning. The authors conducted the study for 9 languages; Arabic, English, German, Indonesian, Italian, Polish, Portuguese, Spanish, and French. They used a dataset from 16 publicly available sources in 9 different languages. They used to classify the dataset as "Hate", and "Free". They combined all the datasets and split into train, test, and validation set with the ratio of 70%, 20%, and 10% respectively. F1_score is used for model performance measure. They applied four models; MUSE + CNN-GRU (), Translation + BERT, LASER + LR, mBERT, in which for a given input sentences MUSE is generated and fed to CNN-GRU, the sentence is translated to English and BERT used and LASER is used to create embedding and it is fed into Logistic Regression model, and directly fed into BERT respectively. They experimented with both monolingual and multilingual scenarios. The result shows that models show

different performances for different languages. LASER+LR and BERT performed best for low resource language and high resource respectively. But, BERT performs best in the case huge data is available.

Table 2.3 Summary of Related Works

Authors and Year	Data source, size, and classes	Best classifier and Performance	Limitations
Y. Kenenisa and T. Melak (2019)	Facebook: 5,000, They experimented with both binary and ternary class [Hate, Normal] & [Hate, Offensive, Neither]	SVM with word2vec F1-score: 73% (for binary) F1-score:53% (for ternary class)	They are limited to machine learning models and only Amharic language.
F. Del Vigna, A. Cimino, F. Dell’Orletta, M. Petrocchi, and M. Tesconi (2017)	Facebook: 3,575 [no hate, weak-hate, and strong-hate]	LSTM with word embedding F1-score: 75.2%	The authors tried only SVM and LSTM. They developed the models only for the Italian language
I. Aljarah et al (2020)	Twitter: 3696 [Hate, not-hate]	RF with TF-IDF Accuracy: 91.3%	Only machine learning models and Tf-IDF feature extraction are considered. The authors didn’t tried deep learning algorithms and feature representation mechanism like word embeddings.
E. Baweke (2020)	Facebook and Twitter: 27495 [Hate, offensive, both and neither]	BiLSTM with word2vec Accuracy:88.89% f1-score:89%	Limited to the Amharic language. Different algorithms and hyperparameters can be tested.
H. Faris, I. Aljarah, M.	Twitter: 3696 [Hate, normal]	LSTM- CNN Accuracy:66.564%,	Besides they developed a model for the mono language they are

Habib, and P. A. Castillo (2020)		F1-score:71.688%	limited to the basic RNNs and CNN models.
S. G. Tesfaye and K. K. Tune (2020)	Facebook: 30,000 [Hate, free]	LSTM Accuracy: 97.9%	Although they got good accuracy, they are limited to the Amharic language. The authors did not use the attention mechanisms and hyperparameters such as spatial dropout are not included.
T. L. Sutejo and D. P. Lestari (2018)	Facebook, Twitter, YouTube, and Line Today Audio:2469 Text: 2273 [Hate, no-hate]	LSTM: using textual features F1-score 87.98%	Even if they tried to use acoustic and text data they didn't used multilingual data and attention mechanisms.

As presented in the table above, people tried different techniques for the task of hate speech detection. The problem is a supervised learning task as we need a labeled hate speech dataset for classification. The lack of a publicly available dataset is one of challenges limiting researchers to focus on a limited number of languages only. Even though people proposed several techniques for the problem of hate speech detection, bilingual text hate speech detection for Afaan Oromo and Amharic languages stayed an open problem.

CHAPTER THREE

RESEARCH METHODOLOGY

In this chapter, we discuss a research methodology for the development of bilingual hate speech detection for Afaan Oromo and Amharic languages. We discuss the ways of data collection, and data preprocessing techniques used to build hate speech datasets. Also, the actual procedure used in building and training models, the model evaluation method used in the research is stated. The processes are briefly described in the next section.

3.1 Data Collection Method

The dataset used for this study contains both Afaan Oromo and Amharic languages. For Afaan Oromo Language the dataset is prepared from scratch by scraping posts and comments from the Facebook platform. After that, it is merged with the Amharic dataset and prepared the bilingual dataset for this study which contains both Afaan Oromo and Amharic languages. To prepare the Afaan Oromo hate speech dataset we followed the following steps:

- ☞ Facebook pages selection and scraping pages.
- ☞ Preprocessing the data and consolidating it into one file dataset.
- ☞ Annotating the data.

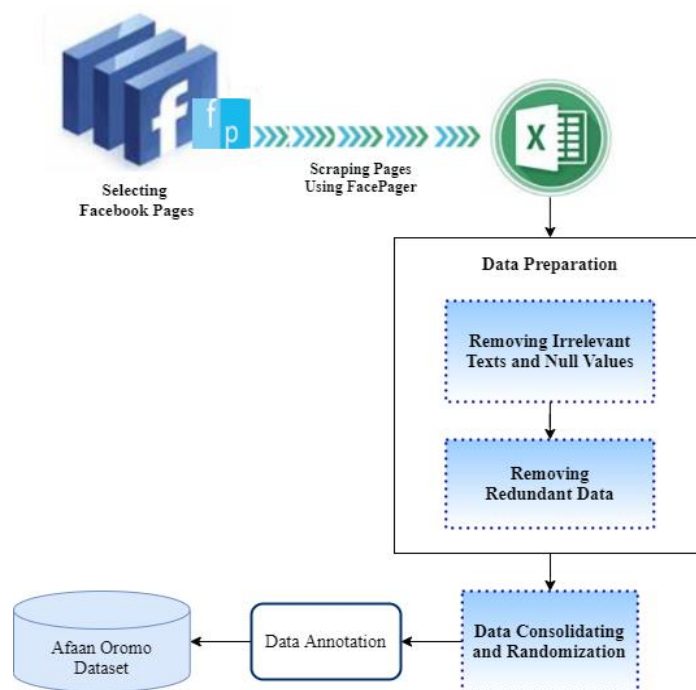


Figure 3.1 Strategy For Collecting Afaan Oromo Data

The posts and comments were gathered from different categories of a popular public page because Facebook's privacy policy does not allow access to the public content of a private page. We decided to use Facebook from other social media platforms because:

- ☞ Facebook is the most widely accessible social media over the world (Clement, 2020)
- ☞ According to multiple statistics preform in Ethiopia's social media, Facebook is the most widely accessible social media in Ethiopia having many users¹⁰.
- ☞ To get the required textual data since peoples use Facebook to post any kind of expression to express their feelings and there are several reactions in comment sessions.

Because of a lack of resources needed to manage and crawl all public Facebook pages in selected categories, we used only a limited number of Facebook pages. To manage the size of the data, we collected only posts and comments posted from September 2020 until May 2021. We used a Facepager which is Facebook Graph API-based data scraping tool. It is an HTTP-based API used to programmatically query data and performs a wide variety of other tasks. To have a representative dataset we identified criteria for Facebook page selection. The collected data is annotated based on an anti-hate speech guideline that is prepared by the Center for Advancement of Rights and Democracy. The data annotation guideline is presented in appendix A. In addition to the guideline, the CARD organization provided us a list of selected hateful keywords that we used during the data annotation process. The Amharic language dataset used for this study is contributed by (Getachew, 2020), which is prepared by collecting posts and comments from the Facebook platform and published on Mendeley. It is available at DOI: 10.17632/ymtmxx385m.1.

3.1.1 Criteria for Facebook Page Selection

- ☞ A page that mostly uses the Afaan Oromo language for posts.
- ☞ Pages having likes and followers greater than 20,000.
- ☞ Pages of religious media, famous vloggers, politicians and broadcasting media are selected to get many and more representative data.

3.2 Data Preparation

Filtering and Cleaning the data primarily use for the next stage, which is the annotation of the posts and comments in the dataset, and then used for training models. The following tasks were performed to prepare the dataset for annotation:

- Removing non-textual posts and comments.
- Removing irrelevant characters.

¹⁰ <https://gs.statcounter.com/social-media-stats/all/ethiopia>

Removing null, blank values, and extra whitespace.

Combining the data into a single file.

Removing duplication to ensure the uniqueness of each text in a dataset.

Based on these criteria, the selected pages, page categories, and the number of data collected from each page are given in the table below:

Table 3.1 Selected Facebook Pages and Collected Data

No	Page Name	Page Catagories	Followers	Likes	Collected Data
1	Oromia Media Network	Media/News Company	1,705,780	1,385,961	1,379
2	Ortodoksii page	Religious Organization	29,294	28,876	196
3	Oromia Broadcasting Service - OBS	Broadcasting & Media Production Company	259,218	243,762	2,878
4	Tvislaamaa	Broadcasting & Media Production Company	209,030	138,812	187
5	OBN Afaan Oromoo	Media/News Company	710,698	565,487	4,396
6	FBC Afaan Oromoo	Media/News Company	573,280	529,342	6,262
7	Raayyaa Abbaamacca	Interest	615,652	539,219	742
8	VOA Afaan Oromoo	Broadcasting & Media Production Company	901,579	795,725	320
9	Ahmedin Jebel official	Personal Blog	705,848	636,905	2,334
10	Andualem Bafakadu Demelce 1	Personal Blog	306,679	293,425	4,092
11	Kello Media	News & Media Website	273,999	182,038	496
12	Hawi – Anole	Personal Blog	153,625	147,166	4,889
13	BBC News Afaan Oromoo	Media/News Company	786,299	644,027	2,515
14	Taye Dendea Aredo	Politician	412,812	381,139	2,973

15	Ustaz Kamil Shamsu	Public Figure	290,541	258,343	595
16	Addisu Arega Kitessa	Politician	170,474	156,555	2,604
17	Mana Lubummaa Oromiyaa	Religious Organization	55,236	51,231	3,690
18	ODP Official	Personal Blog	79,734	76,788	4,088
Total Number of Data Filtered					44,616
Total Number of Unique Data Filtered After Removing Redundancy					40,000

3.2.1 Data Annotation

The dataset is labeled into different classes in data annotation process. The data annotation task is time taking, and it needs budgets needed for annotators. we did not annotate all the collected data because of our limited time and budget, rather we used a random selection of texts to be annotated. Five annotators participated in the annotation task, four MSc students and one Ph.D. student. We selected them based on their willingness to perform the task and their background in the Afaan Oromo language, they can read, write and understand the language. The annotation task is performed based on the annotation guideline presented in appendix A. Based on this guideline, the annotators were instructed to label each post and comment to two different classes ‘hate’ and ‘free’. All annotators were instructed to annotate 52,00 instances of datasets among that 3700 instances are unique and 1500 instances are the same for all annotators. We used the same 1500 instances annotated by all annotators to measure the inter-rater agreement. The inter-rater agreement gave a kappa value of 0.664 and which is a good level of agreement (3.4.1 Inter Annotator Agreement (IAA)). The annotation results by each annotator are given in the table below:

Table 3.2 Unique Instances annotated by Each Annotator

Labels	Annotator1	Annotator2	Annotator3	Annotator4	Annotator5	Total
Free	1836	1846	1848	1,860	1,892	9,282
Hate	1864	1854	1852	1,840	1,808	9,218
Total	3,700	3,700	3,700	3,700	3,700	18,500

Tab 3.3 Common Instances Annotated By Each Annotator

Labels	Annotator1	Annotator2	Annotator3	Ammotator4	Annotator5	Total by Voting
Free	685	700	709	706	701	703
Hate	815	800	791	794	799	797
Total	1,500	1,500	1,500	1,500	1,500	1,500

From annotation results, as shown above, a total of 20,000 instances are annotated where 9,985 of them are labeled as “Free” and the rest 10,015 are labeled as “Hate” for the Afaan Oromo language. Finally, the bilingual dataset is prepared by merging the Afaan Oromo dataset with the Amharic language dataset. Since the Amharic language dataset contains 30,000 instances and the prepared Afaan Oromo dataset contains 20,000 instances, totally we got a dataset with 50,000 instances. From the 50,000 instances of the bilingual dataset 24,036 of them are labeled as “Free” and the rest are labeled as “Hate”. The distribution of the dataset by class is given in the table below:

Table 3.4 Dataset Distribution By Class

Labels	Total Number Of Instances
Free	24,036
Hate	25,964
Total	50,000

3.3 Bilingual Hate Speech Detection Modeling

3.3.1 Data Preprocessing

Preprocessing refers to removing noises from the data or cleaning, normalization, and tokenization that helps for effective feature selection. In this stage, we perform the process of cleaning input data for further analysis. Since texts on social media contain many unnecessary contents such as special characters, HTML scripts, and URLs, we remove such unwanted components to get cleaned texts for use. We used the following methods in preprocessing:

- ☞ Eliminating punctuations, numbers, symbols, blank value and whitespace, URL, and emojis.
- ☞ Text normalization.
- ☞ Tokenization

3.3.2 Feature Extraction Methods

The feature extraction technique is applied for reducing redundant detection problems, features, and dimensionality. For this study word embedding technique is used for feature representation.

3.3.2.1 Word embedding

Word embedding allows feature representation in such a way that words with similar contexts have similar values in the vector space. Simply put, it is a mapping of words into vectors of real numbers. The way word embeddings work makes them ideal for the current state-of-the-art deep learning algorithms, and this brought an improvement in the performance of many natural language processing challenges. There are various word embeddings models like FastText, Glove, and word2vec. Word2vec is trained for this study using the whole dataset without the label.

3.3.2.2 Word2vec

Word2vec is used to create word embeddings using shallow neural networks (having only one hidden layer). It can capture the semantic similarity between words. It does so by projecting words into an n-dimensional space and giving words with similar contexts similar places in this space. Simply the model gives relationships between the words in the text. Training Word2vec helps to get a feature model from the unlabeled dataset. The CBOW and skip-gram models are two mechanisms for training word2vec and they are differentiated in the following ways¹¹:

- ☞ The skip-gram model works well when we have smaller training data and it better represents words that are less frequent in the text.
- ☞ CBOW model requires large training data and more frequent words to work better.

To get the advantage of achieving a better model with a small dataset and less frequent data, we used the skip-gram model for the proposed feature representation.

3.3.3 Deep Learning Models

3.3.3.1 Recurrent Neural Networks (RNN)

RNN is a type of ANN that is useful to model sequential data (Salehinejad et al., 2017). Unlike feedforward neural networks where the output is based on only the current input, RNN considers the output from the previous layer making it best for modeling sequential data. RNN is named as feedback neural networks since the connections between neurons can form directed cycles. This network making use of sequential information. RNN has a memory about what has been calculated so far and uses it on current output computation to store the semantics of all the past content in a hidden layer. A

¹¹ <https://towardsdatascience.com/comparing-word-embeddings>

recurrent neural network is modeled to remember each information throughout time which is necessary for any time series predictor. However, RNN faces a problem when it is necessary to remember long-term data. As training samples gradients get propagated, it gets weaker by the time it gets to the neurons representing older data points in the time series it is unable to adjust them properly. It is a vanishing gradient problem and can be solved by LSTM and GRU.

GRU and LSTM networks use a memory cell to store long-term information. For controlling the flow of information in the network GRU's and LSTM's make use of gates. Gates are used for learning which input in the sequence is important and store the information in the memory unit. LSTM has an input gate, output gate, and forget gate. The input gate decides the extent of information to be written onto the internal cell state, and the forget gate decides to erase the cell or not. The output gate determines how much to reveal of a cell. GRU contains reset gate and update gate. The update gate is used to determine how much the past information should matter now, and the reset gate decides to drop the previous information or not.

However, LSTM and GRU networks use only the previous context of a word. But, there is bidirectional RNN enabling to store past and future information. In bidirectional LSTM and bidirectional GRU, the first recurrent layer is replicated, where the replicated RNN is used to store the context accepting reversed copy of the input sequence. This makes the BiRNN better than simple RNN networks. Hence, we used BiLSTM and Bi-GRU networks.

3.3.3.2 Attention Mechanism

Although RNNs provide admirable results on different NLP tasks like text classification, they are not intuitive enough for poor interpretability (Qian et al., 2020). Usually, emotions and meanings vary to different parts of a sentence. Some words in the sentence are more informative and the others are not. An attention mechanism is introduced to attend informative words and aggregate their representations to form a sentence vector (Yao, 2017). It can learn how much contribution of each word and sentence for the classification judgment, which is beneficial for applications and analysis through the two levels of attention. The attention mechanism can improve the performance with interpretability for text classification, which makes it popular.

3.4 Evaluation Method

3.4.1 Inter Annotator Agreement (IAA)

Inter annotator agreement evaluation computes the agreement of human annotators of the dataset by calculating how similar the annotators labeled posts and comments in the dataset. The study used Fleiss's kappa for calculating the agreement level. The annotators are instructed to annotate the same

amount of similar subsets of posts and comments in the dataset. To calculate the kappa using the dataset that receives annotation from all annotators. The Kappa value varies from 0 to 1. The standard interpretation of Kappa values are listed as follows.

Table 3.5 Standard Interpretation of Kappa Values

Agreement	Kappa Values
Poor agreement	Between 0 and 0.2
Fair agreement	Between 0.2 and 0.4
Moderate agreement	Between 0.4 and 0.6
Good agreement	Between 0.6 and 0.8
Very good agreement	Between 0.8 and 1.0
Perfect agreement	1.0

3.4.2 Model Evaluation Methods

Model evaluation should be done to see if the proposed model is performing well on unseen data. Evaluating models is a necessary step in machine learning to check model generalizability on unseen data. two mechanisms of model evaluation using unseen data, cross-validation and holdout method(Mutuvi, 2019). The data is divided into training, validation, and testing sets in a holdout. Most of the time holdout method is used because of its simplicity, speed, and flexibility. But, it is associated with high variability of the data in the training set and test set, resulting in low validation accuracy. Cross-validation is the other most widely used technique. Commonly used cross-validation is k-fold cross-validation. This method is known for its skill in the less biased or less optimistic estimation over the previous simple train/test split approach. This technique significantly reduces variance, hence we used k-fold cross-validation for this study.

3.4.2.1 K-Fold Cross-Validation

The dataset is split into k number of folds where the k-1 fold is used for training and 1 fold for testing. K value represents the number of folds that a given data sample is to be split into. It works as follows:

1. Randomly split the data into k-folds for training
2. Model is trained using K -1 folds and validate the model using the remaining Kth fold.
3. Step 2 is repeated until every K-fold serves as the test set.
4. Averages of the recorded scores will be the performance metric for the model

3.4.2.2 Classification Metrics

Besides the cross-validation testing, this study used different classification metrics appropriate for model performance evaluation. They are discussed below:

Confusion Matrix: Sometimes classification models can have a good accuracy score even if there is misclassification in prediction. This happens especially in case we have unbalanced data, where the model will predict only the major class without any calculation. At this time we need to have deep insight into the model prediction to check misclassification on each class, and this is where the confusion matrix comes into the picture. It is a table of $n \times n$ matrix where n is the number of classes. In our case we have two classes, hence the confusion matrix is 2×2 .

- ☞ **True Positive (TP):** This is the case when the actual class of the data point was True and the predicted class is also True.
- ☞ **False Positive (FP):** This is the case when the actual class of the data point was False and the predicted class is True.
- ☞ **False Negative (FN):** This is the case when the actual class of the data point was True and the predicted class is False.
- ☞ **True Negative (TN):** This is the case when the actual class of the data point was False and the predicted class is also False.

Table 3.6 Confusion Matrix Format

		Actual Label	
		Hate	Free
Predicted Label	Hate	True Hate	False Hate
	Free	False Free	True Free

Accuracy: The accuracy of the model indicates the rate at which it predicts results correctly.

$$Accuracy = \frac{TP+TN}{Total\ Instances\ (TP+TN+F+FP)} \quad 3.1$$

Precision: Precision calculates the number of values that are correctly assigned concerning all automatically classified sentences.

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

Recall: is used to measure the amount of correctly identified actual positives.

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

F1-Measure: F1-measure is a harmonic mean of precision and recall. Sometimes it is known as the f1-score. F1-score is computed using the following formula.

$$F1 - score = 2 \times \left(\frac{Recall \times Precision}{Recall + Precision} \right) \quad 3.4$$

3.5 Model Prototyping

After model development by training the models and applying cross-validation, the best-performing model is selected and deployed on the webserver, so that users can access the webserver to check posts/comments whether it is “free” or “hate”. First, the trained model is saved as a file. Then, the saved model is loaded on the Flask web server so that it became reachable through the WSGI HTTP server to the users.

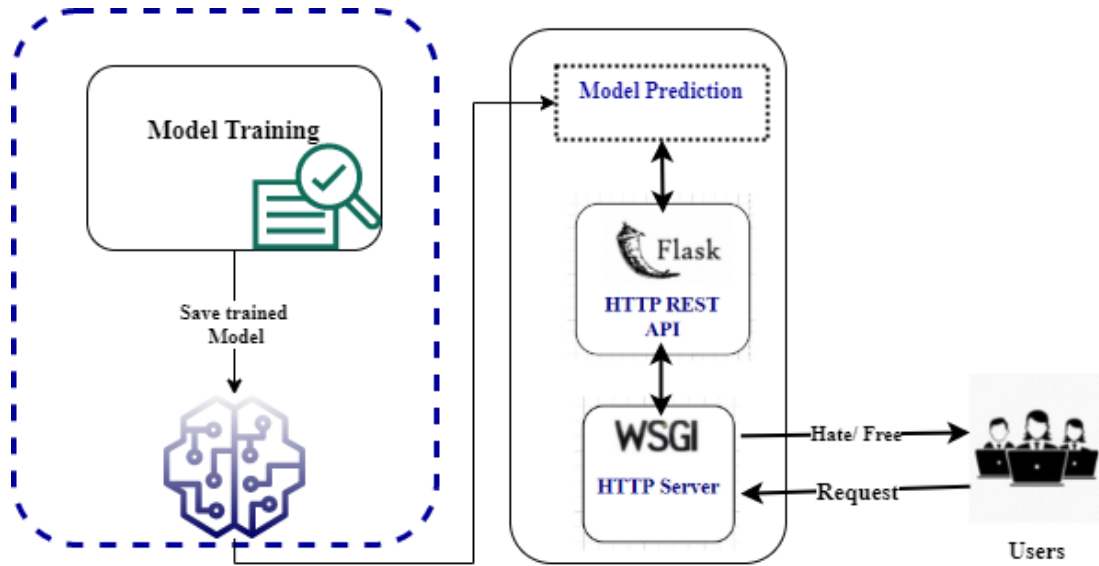


Figure 3.2 Deployment Architecture of the Proposed Bilingual Hate Speech Detection

CHAPTER FOUR

THE PROPOSED SOLUTION FOR THE BILINGUAL HATE SPEECH DETECTION

This chapter discusses the proposed solution for bilingual hate speech detection for Afaan Oromo and Amharic languages using deep learning approaches. First, the general architecture of the proposed bilingual hate speech detection is presented. Then, this chapter discusses the proposed preprocessing techniques, feature representation techniques, and design of deep learning model architectures used for this study.

4.1 The Proposed Architecture

The proposed bilingual hate speech detection is aimed to classify Afaan Oromo and Amharic posts and comments on social media as hate, and free. As shown in Figure 4.1, the proposed architecture contains components of preprocessing, feature representation, model building, and model evaluation. It takes the bilingual dataset containing Afaan Oromo and Amharic languages and the preprocessing techniques are applied to the dataset. Preprocessing methods include cleaning, normalization, and tokenization before representing the dataset as the feature vector.

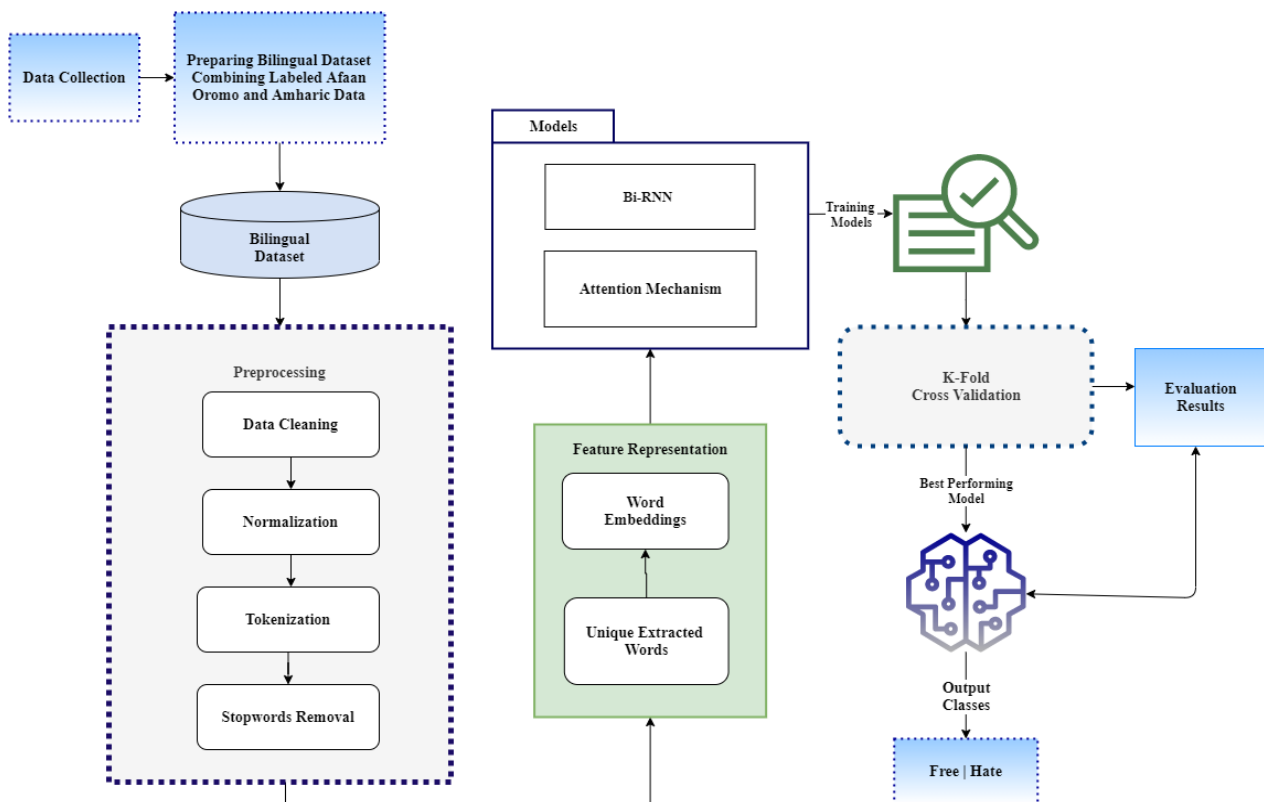


Figure 4.1 General Architecture for The Proposed Bilingual Hate Speech Detection Modeling

After preprocessing word embedding is used for representing the unique extracted words obtained from tokenization as a feature vector. Bidirectional recurrent neural networks (Bi-RNN), and attention-mechanism are used for model development. To select the best detection model the models were evaluated using 5-fold cross-validation. The best performing model is selected for classifying the data into the classes of “hate” or “free” based on the evaluation results. Finally, the selected best-performing model is used to develop a prototype of the detection model that can take a new Afaan Oromo and Amharic texts as input and classifies the input as free, or hate speech. Each subcomponent of the proposed architecture is explained next to this section.

4.2 Preprocessing

Why preprocessing? Because the real world data contains different unwanted contents, inconsistency and format that is difficult for the machine to analyze it. Therefore, we need to represent the data in such a way that it can be analyzed by machine, and this is what preprocessing steps do. The preprocessing step concerned with removing noises from the data or cleaning, normalization, and tokenization. As social media texts contain several less useful contents such as links, punctuations, and other special characters those should be removed during preprocessing for effective feature representation. Under this component preprocessing of Afaan Oromo and Amharic text is performed. This preprocessing component includes cleaning, normalization, and tokenization.

4.2.1 Cleaning

To remove unnecessary content and make the data more representable by the word embedding to be employed we shall go over some data cleaning steps. This cleaning procedure will get rid of all irrelevant special characters, symbols, and emojis. Pseudocode to remove irrelevant characters (cleaning) is as given below:

Algorithm 4.1 Data Cleaning Algorithm

BEGIN:

1. Read the text in the dataset;

2. While (! end of the text in a dataset):

If the text contains special characters and symbols [' ', '!', '"', ':', ')', '(', '-', '!', '?', '|', ';', '""', '\$', '&', '/', '[', ']', '>', '%', '=', '#', '*', '+', '\\', '•', '~', '@', '£', ' ', ' ', '{', '}', '©', 'ª', '®', '™', '<', '→', '°', '€', '™', '›', '♥', '←', '×', '§', '™', '™', 'ˆ', ' ", '½'] then

Remove special characters and symbols

If the text contains number [0-9] & [፩ ፪ ፫ ፬ ፭ ፮ ፯ ፰ ፱ ፲ ፳ ፴ ፵ ፶ ፷ ፸ ፹ ፺ ፻ ፼ ፽ ፿] then

Remove Arabic and Ethiopic numbers

If a text contains emojis [😊, 😄, 🐼, 😄...] then

Remove emojis

If a text contains extra white space, then

Remove extra space

3. Return clean_text;

END:

4.2.2 Normalization

Why normalization? Normalizing text is attempting to reduce its randomness and bring it closer to a predefined standard. It is very useful to reduce the amount of different information the computer has to deal with and therefore improves efficiency. Its main goal is to group related tokens, where tokens are usually the words in the text. The task of text normalization includes converting texts to a similar case for the Afaan Oromo language. It is best to convert characters into lowercase since most of the time the users use lower case without dealing with the capitalization. Hence, all characters changed to lower case. For example, the word “UMMATA” is changed to “ummata” after changing to lower. Handling extra whitespaces is important to group the related tokens, therefore we applied those operations. For the Amharic language, the morphology is more complex and needs more normalization techniques. Some Amharic letters have the same sound but are spelled differently. Therefore people can write the same thing with a different spelling. For Example, በልተዋል and በልቶካል could be normalized to በልቷል. We applied normalization on such Amharic letters.

Algorithm 4.2: Algorithm For Normalization

INPUT: Unprocessed dataset

OUTPUT: Normalized dataset

BEGIN:

1: Read the dataset

2: WHILE(it is not the end of file):

 IF text contains [ሃጎኃሐሐኸ] THEN replace with 'ሀ'

 IF text contains [ሐኑኸኸ] THEN replace with 'ሁ'

 IF text contains [ኸሐኸኸ] THEN replace with 'ኸ'

 IF text contains [ኃሐኸኸ] THEN replace with 'ኸ'

 IF text contains [ሐኸ] THEN replace with 'ሀ'

```

IF text contains [ፖሎኸ] THEN replace with 'ሆ'
IF text contains [ዓኣዐ] THEN replace with 'ኣ'
IF text contains[ሠ ሡ ሢ ሣ ሤ ሥ ሰ] THEN replace with [ሰ ሱ ሲ ሳ ሴ ስ]
IF text contains [ዑ ዒ ዓ ዔ ዕ ዖ ኣ] THEN replace with [ኣ ኤ ኦ ኢ ኦ ኦ ኦ ኦ]
IF text contains [ጸ ጹ ጺ ጻ ጼ ጽ ጺ] THEN replace with [ፀ ፁ ፊ ፍ ፈ ፅ ፉ]
IF text contains [ሉ[ዋኣ]] THEN replace with [ሊ]
IF text contains [ሙ[ዋኣ]] THEN replace with [ሚ]
IF text contains [ቱ[ዋኣ]] THEN replace with [ቷ]
IF text contains [ሩ[ዋኣ]] THEN replace with [ሯ] . . .
Return normalized text
ENDIF

```

3 END

4.2.3 Tokenization

Why tokenization? Tokenization is the process of converting text into tokens so that they can be considered as discrete values before transforming it into vectors. It is also easier to deal with and filter out unnecessary tokens. For example, a document into paragraphs or sentences into words. It detects the boundaries of a written text. In this step, input texts are tokenized into a stream of characters using white spaces and punctuation which helps to convert into a list of words. The process detects the boundaries of a written text. The input to deep learning models should be tokenized as text $T = \{t1, \dots tN\}$, where each word is represented by the word embeddings. For example:- The word “walqixxummaa sabaaf sablammii hundaa” tokenized as [“walqixxummaa”, “sabaaf”, “sablammii”, “hundaa”], and [“እርግጠኛ መሆን አለብህ”] tokenized as [“እርግጠኛ”, ”መሆን”, ”አለብህ”].

4.2.4 Stopwords Removal

Stop words removal is also another crucial step in preprocessing. There are many stopwords in Amharic and also Afaan Oromo language that is redundant in the dataset. Removing such stopwords is advantageous since we can get the core idea of the text even if they are removed, and it minimizes redundancy in the dataset which could improve classification performance. For example, in Afaan Oromo words like “akka”, “irra”, “hin” are redundantly appeared in the dataset, and removing those texts cannot change the idea of the text. In Amharic also words like “እኔ”, “እኛ”, “ነህ” are mostly redundant and less informative in the dataset. hence, we applied this preprocessing step for both languages Amharic and Afaan Oromo.

4.3 The Proposed Feature Representation

4.3.1 Word Embeddings

The proposed feature representation component converts the dataset to feature vectors. As computers can't understand text, feature representation techniques are used to change the text data to numerical form that could be understood by a machine. We used word embeddings for feature representation. Word embeddings are unsupervised learning of word representation whose relative similarity correlates with semantic similarity(Rushdy, 2021). Hence, we used word embedding to get the advantage that they can model semantic similarity of words. We trained word2vec for the proposed bilingual hate speech detection by using posts and comments in the dataset without labels. Word2vec can be trained whether using skip-gram or continuous bag-of-words. For this study, the skip-gram model is used to create our word2vec model, in which the neural network finds the context words given the target word.

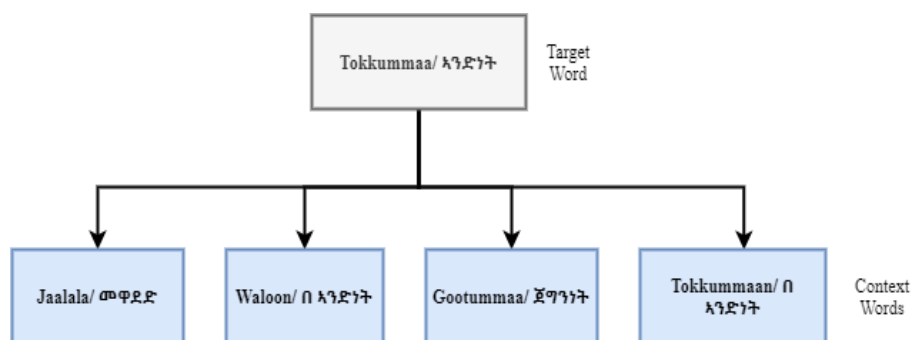


Figure 4.2 The Proposed Skip-gram Model Architecture

4.4 The Proposed Deep Learning Model Architectures

4.4.1 Bidirectional RNN Models

Bidirectional recurrent neural networks contain a combination of two recurrent neural network layers the backward and forward layers. The RNN added in the forward layer is used to read the input sequence beginning from start to end of the sequence and stores forward information (past) only. Then, the backward RNN layer is added to read the input sequence in reverse order from the end of the input sequence back to its beginning and store future information. After that, the combination of the forward information and the backward information is used as an output of the recurrent layer.

The final output of the recurrent layer of bidirectional RNN contains the past and future information which helps to remember long-term information. For this study, Bidirectional LSTM and bidirectional GRU networks are proposed since both LSTM and GRU networks work to eliminate the long-term dependency problems in simple RNNs. Architectures of both BiLSTM(Figure 4.3) and Bi-GRU(Figure 4.5) are the same except the recurrent layer is changed that is LSTM and GRU network

respectively. The other layers, embedding layer, spatial dropout, dense layer, and output layers are the same for both networks.

Embedding Layer: For the embedding layer, word embedding is used to represent the uniquely extracted words. The word embedding with 300 dimensions is applied for this study. Also, the prepared word2vec is trained by using the same 300 dimensions so that it is supported by this embedding layer. The embedding layer is similar for both BiLSTM and Bi-GRU models.

Recurrent Layer: BiLSTM and BiGRU is used in the recurrent layer for BiLSTM and BiGRU respectively. The bi-directional LSTM is obtained by adding the LSTM layer in forwarding and backward directions to understand the past and future information. The operation is the same for BiGRU unless the GRU gates are used instead of LSTM in the case of bidirectional GRU.

Spatial Dropout layer: SpatialDropout1D is used after embedding layer. Dropout and SpatialDropout1D come to the picture when the neurons in the embedding layer are correlated. Dropout randomly removes elements in the embedding. But Dropout is not enough since the other information is still correlated even if some elements dropped. SpatialDropout1D drops the 1D feature maps from embedding rather than dropping some feature elements. In our case also using SpatialDropout1D in addition to Dropout is very important since there are many redundant and less informative features or words in the dataset. In such a case, the SpatialDropout1D layer drops those 1D features which help to minimize the complexity of the dataset and also has a great role in enhancing the model performance. For example if you see the input layer of BiLSTM and Bi-GRU given below, there are words like malee/ᱠᱟᱨᱢᱟᱝ and hundi/ᱠᱟᱨᱢᱟᱝ are less informative relative to the other words in the sentence. So, the spatial dropout will drop such features and so that more informative words only fed in to the recurrent layer. Dropout is also used after the recurrent layer to disable some feature elements before being fed into the dense layer. Finally, the sigmoid layer is applied to classify the data into two classes as “Hate”, and “Free”.

Output layer: After the dropout and dense layers the sigmoid layer is used for the output prediction. The sigmoid function predicts value 0 or 1 and it is used for binary classification. Our system has binary classes and hence, we applied sigmoid for the output layer. The sigmoid function works using the formula:

$$\sigma(x)=1/(1+e^{-x}).$$

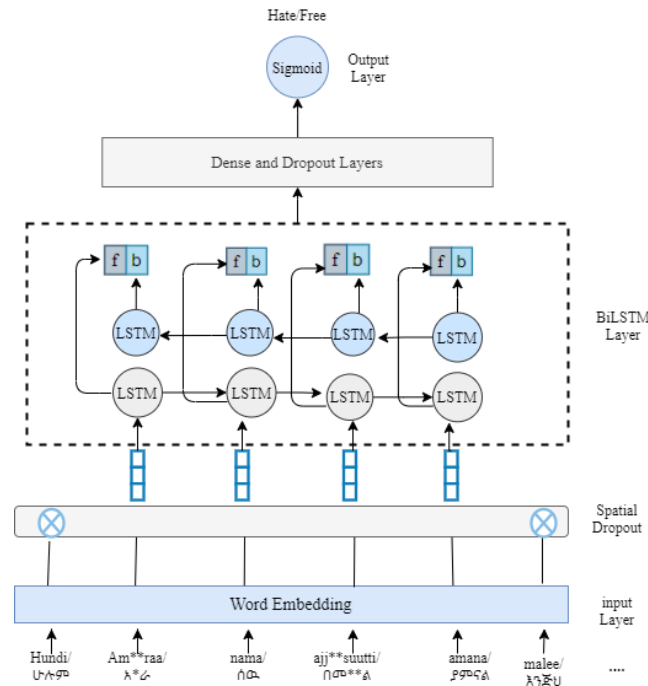


Figure 4.3 The Proposed BiLSTM Architecture

The recurrent layer of the BiLSTM is combination of LSTM cell. Each LSTM cell contains forget gate (f_t), input gate (i_t), and output gate (o_t). Gates are used to let optionally information through.

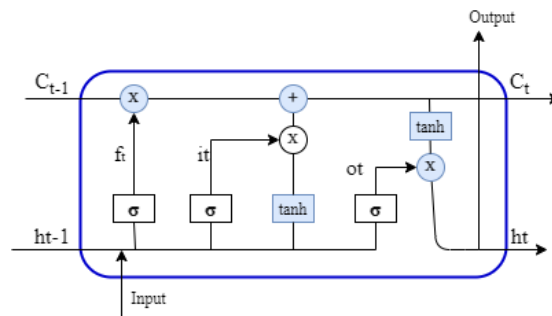
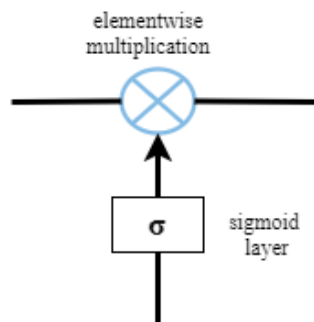


Figure 4.4 LSTM Cell

Each gate composed out of sigmoid layer and pointwise multiplication.



The working strategy of the LSTM cell for the proposed BiLSTM model is explained below:

First, the forget gate looks at the h_{t-1} and the input x_t to decide the information to be thrown away from the cell state. Then it gives output 0 meaning throw, or 1 to keep the information. It is calculated as:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f)$$

Then, the information to be stored in the cell state is decided by the input gate. It identifies the value that will be updated. Then the $\tanh$ creates $\hat{C}_t$ i.e., the new candidate to be added to the cell state. The input gate it and the candidate $\hat{C}_t$ calculated as:

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i)$$

$$\hat{C}_t = \tanh(W_c.[h_{t-1}, x_t] + b_c)$$

Then, the old state c_{t-1} is updated to c_t . by forgetting the information earlier decided to, f_t is multiplied by the c_{t-1} and add multiplication of it and $\hat{C}_t$ to get the new cell state.

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t$$

Finally, the last step is deciding what should be the output which is done by the output gate.

$$O_t = \sigma(W_o.[h_{t-1}, x_t] + b_o)$$

The final output layer is obtained from the tanh layer. Which is $h_t = o_t * \tanh(C_t)$.

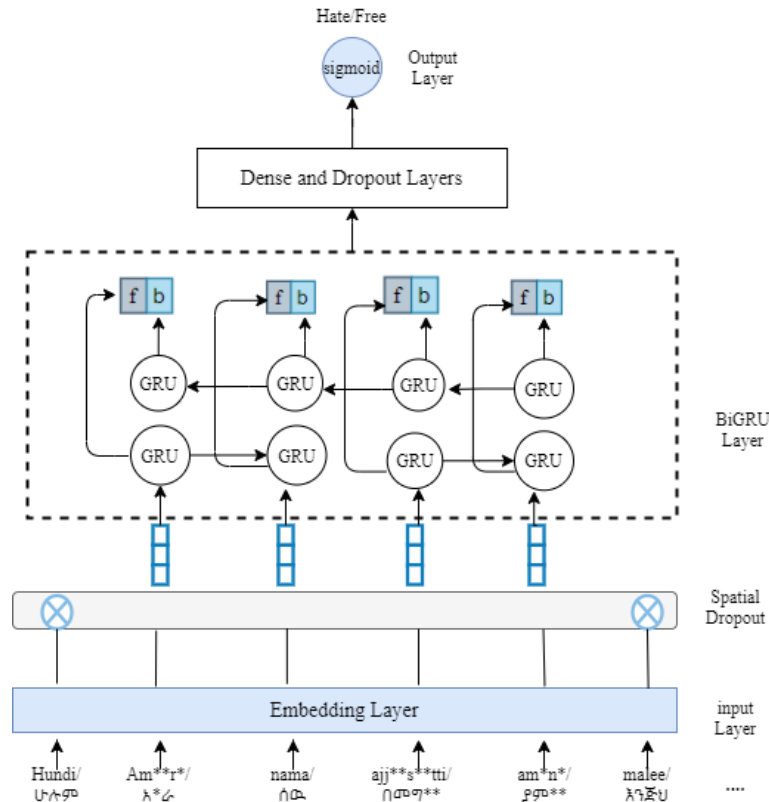


Figure 4.5 The proposed Bi-GRU Architecture

The recurrent layer of the Bi-GRU is combination of GRU cell. The GRU cell simplifies the LSTM by merging forget gate and input gate into update gate z_t , which controls the forgetting factor and the decision to update the state unit. It contains also the reset gate r_t that controls which parts of the state get used to compute the next target state.

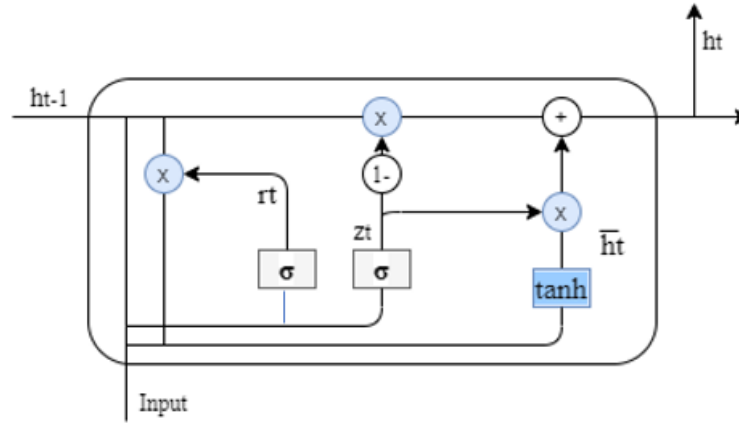


Figure 4.6 GRU Cell

First, calculate z_t at timestamp t . Then, how much past information to be forgotten is decided by the reset gate r_t . After that, the new content to be stored is calculated and the final output h_t is calculated by the tanh layer.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

$$\bar{h}_t = \tanh(W_h \cdot [r_t \cdot h_{t-1}, x_t] + b_h)$$

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \bar{h}_t$$

4.4.2 Attention Mechanism

The layers of the proposed attention mechanism models are the same as the BiRNNs. But, the attention layer is added after the recurrent layer. (Bahdanau et al., 2015) presented attention mechanism as a solution for the problem of the encoder-decoder model. In the case of the encoder-decoder model, the input is input sequences to one fixed-length vector by the input network, and based on that the output is decoded. However, when we need to focus only on a certain set of inputs that are responsible to provide a particular output encoder-decoder model cannot do that. This is the case why we need an attention mechanism. The attention mechanism is useful to focus on different positions of a single sequence (Vaswani et al., 2017). The attention layer is added in between the encoder layer and the decoder layer to calculate the importance of each input and focus on certain inputs with high weight.

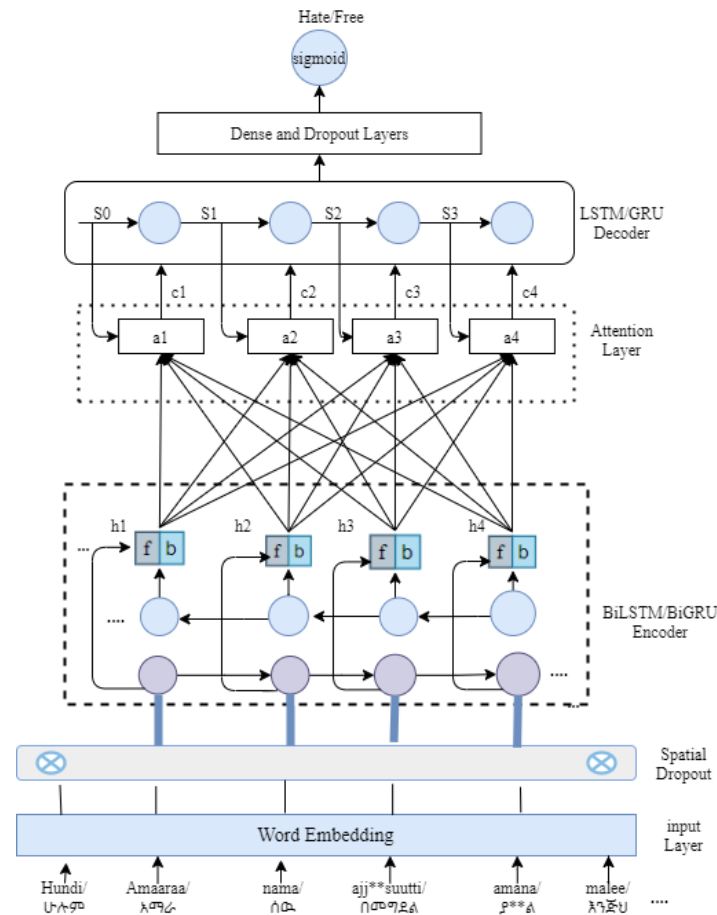


Figure 4.7 The Proposed Attention Mechanism

The encoder layer is the network accepting input values which is a bidirectional LSTM and GRU in our case. The decoder layer accepts the output of the attention layer and provides an output based on the learned context. In a simple encoder-decoder model the decoder layer decides the final output only based on the output from the encoder layer. The attention mechanism works in such a way that the model can identify what kind of inputs are responsible to give a particular output. The main function of the attention layer is to generate context vectors. The attention layer contains the following three main processes:

- ☞ Alignment
- ☞ Weighting (softmax calculation)
- ☞ Context vectors generation

Alignment: in this step, the output of the decoder layer passed onto perhaps more layers before outputting the final prediction. The alignment operation is putting together the output of the decoder layer with the input of the attention layer h_j which are words and output from the recurrent layer. $S_0, s_1, s_2, s_3..$ represents the feedback of the decoder layer and h_j represents the $h_1, h_2, h_3, \dots, h_{Tx}$ word

features. Based on its importance for providing the output of the decoder the input h_j is aligned with the output of the decoder layer. The value of “e” represents the attention annotation and it can be calculated by using the following equation:

$$e_{ij} = a(S_{i-1}, h_i) \quad 4.1$$

where:

- ☞ a represents attention
- ☞ S_{i-1} output from the decoder from the previous output time step
- ☞ h_i is the output of the decoder layer

Weighting (softmax calculation): Allows threatening scores like probabilities indicating the likelihood of each encoded input time step being relevant to the current decoder output. It is known as weight annotation. The link from the output of encoder layers $h_1, h_2, h_3, \dots, h_{T_x}$ to the attention $a_1, a_2, a_3, \dots, a_t$ is assigned a weight α . For example for the link from h_1 to a_1 the α_{ij} is α_{11} and so on. The $h_1, h_2, h_3, \dots, h_{T_x}$ are word features obtained from the recurrent layer. Generally, the weight is represented as α_{ij} and it is calculated as follows:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \quad 4.2$$

Where:

- ☞ T_x is the number of attention inputs or the number of inputs to be focused on by the neural network

Context vectors: Context vectors are the word features we get from the attention layer after calculating their importance. This is a weighted sum of the annotations and normalized alignment scores and it can be calculated using the following formula:

$$C_i = \sum_{k=0}^n \alpha_{ijk} h_k \quad 4.3$$

Where: h_i is the output from the encoder layer

CHAPTER FIVE

IMPLEMENTATION OF THE PROPOSED BILINGUAL HATE SPEECH DETECTION

This chapter presents the implementation of the proposed models for bilingual hate speech detection for Afaan Oromo and Amharic languages. Mainly this chapter presents the implementation of the proposed methodologies and the required tools for experimentation. We start by discussing implementation tools and packages and next, we discuss the actual implementation.

5.1 Tools

5.1.1 Data Preparation And Preprocessing Tools

Facepager 4.3.3¹²

FacePager is an open-source tool for collecting publicly available data from social media platforms such as Facebook, Twitter, YouTube, and other websites based on APIs and web scraping. It is useful in the data collection process to build the dataset by collecting data from social media. The collected data could be stored on a standard database like SQLite and can be extracted in a CSV file. We used Facepager since it simplifies the data collection process and it can easily extract the collected data in the form of a CSV file.

Scikit-learn 0.21.3¹³

Scikit-learn is a free machine learning library for Python. Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy, and Matplotlib.

Pandas 1.2.3

It is an open-source Python library that provides high-performance data manipulation and analysis tools using its powerful data structures. Pandas present a diverse range of utilities, ranging from parsing multiple file-formats to converting an entire data table into a NumPy matrix array. It is a high-

¹²<https://github.com/strohne/Facepager>

¹³ https://www.tutorialspoint.com/scikit_learn/scikit_learn_introduction.htm

performance, easy-to-use, and data analysis tool. We used pandas for data reading, manipulation, writing, and handling the data frame.

Numpy 1.19.5¹⁴

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object and tools for working with these arrays. We used NumPy for handling converting the text to numeric data for features and training and testing the model.

LibreOffice 7.1.3¹⁵

LibreOffice is a powerful and free office suite and it provides a clean interface and feature-rich tools that help to unleash your creativity and enhance your productivity. LibreOffice includes Writer (for word processing), Calc (spreadsheets), Impress (presentations or PowerPoint), Draw (vector graphics and flowcharts), Base (databases), and Math (formula editing).

5.1.2 Package Managers and Environments

Anaconda Navigator 4.10.0

Anaconda Navigator is a GUI that contains the Anaconda individual edition, which makes it easy to launch applications and manage packages and environments without using command-line commands. It helps with creating different environments, training with different python and package versions easily.

Google Colab¹⁶

It is a free Jupyter notebook environment that runs entirely in the cloud. Colab is a free cloud service that helps us to use free GPU. It supports many popular machine learning libraries which can be easily loaded into your notebook. It helps to run very resource and time-consuming deep learning codes efficiently and also it doesn't need to install any package explicitly.

Jupyter Notebook 6.0.1

Jupyter notebooks is An open-source web application that helps us to make and contribute documents that contain live code, equations, visualizations, and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, and machine learning. It has been incorporated with an anaconda navigator and we used it for model implementation.

¹⁴ <https://cs231n.github.io/python-numpy-tutorial>

¹⁵ <https://www.libreoffice.org/discover/libreoffice/>

¹⁶ https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm

5.1.3 Modeling Tools and Packages

Python 3.7.4

Python is an object-Oriented, interpreted, high-level programming language with dynamically rich in semantics. It is a powerful programming language to develop a machine learning application to process linguistic data, easy to learn. Many reasons making python a really good programming language choice for NLP tasks. It has a simple syntax and transparent semantics that make it an excellent choice for Natural Language Processing tasks. Moreover, python provides developers with an extensive collection of NLP tools and libraries that enable us to handle a great number of NLP-related tasks such as document classification, topic modeling, part-of-speech tagging, word vectors, and sentiment analysis¹⁷.

RegEx 2.2.1

RegEx (regular expression or RE) specifies a set of strings that matches it; the functions in this module let you perform string matching, removal, and replacement. We used this package for preprocessing of text.

Gensim 3.8.0

Gensim (Generate Similar) is a Python library for topic modeling document indexing and similarity retrieval with a large dataset. This study uses it for the constricting word2vec model.

Matplotlib 3.1.1

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits. In this study, we used it for data and results visualization.

TensorFlow 2.1.0¹⁸

TensorFlow is an end-to-end open-source platform for machine learning tasks. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in deep learning. The flexible architecture of TensorFlow allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of

¹⁷ <https://sunsrapers.com/blog/8-best-python-natural-language-processing-nlp-libraries/>

¹⁸ <https://www.tensorflow.org>

servers to mobile and edge devices(Pavithra & Prakash, 2019). It has an accessible and readable syntax, but at the same time with more flexibility, it's preferred over the other deep learning platforms. Aside from its flexibility, TensorFlow is given the researcher more network control and understanding of operations done by a specific model.

Keras 2.3.1¹⁹

Keras is one of the python libraries which a high-level neural network that runs on top of TensorFlow. It is a fast experimentation deep learning API written in python. Because of its consistent & simple high-level API, working with Keras is speedier and allows us to do more experiments with less strain.

5.1.4 Deployment Tools

Flask 1.1.1

After the experiments are done the best performing model is deployed on a web server, where it can be used to make a real-time prediction by providing the story needed to be verified. To realize this, we used a flask micro web framework. Since we are deploying a deep learning model that is based on a simple HTTP request-reply in text form, Flask is more favorable over other Heavyweight frameworks.

5.1.5 Hardware Tools

The tools which are discussed above in section 5.1 have been deployed on a personal computer equipped with a processor Intel® Core™ i5-4310M, CPU 2.70GHz, 2 Core(s), 8 Gigabyte of physical memory, 1TB (1000GigaByte) hard disk storage capacities. The operating system is Windows 10 Pro, 64 bits.

5.2 Data Preprocessing and Model Implementation

5.2.1 Loading Dataset

To implement the preprocessing method, we used the Python programming language. Before doing any pre-processing on the data, the data should be made available to the python environment we are doing. We used Pandas to load the dataset since Pandas supports reading data as DataFrames. Pandas is very efficient with small data usually from 100MB up to 1GB²⁰. The Pandas method read_excel reads the file and load it as DataFrames, where instances of the data are accessible via column name.

¹⁹ <https://keras.io>

²⁰ <https://towardsdatascience.com/why-and-how-to-use-pandas-with-large-data>


```

def char_normalization(x):
    h1=['U', 'U', 'U', 'U', 'U', 'U', 'U']
    h2=['H', 'H', 'H', 'H', 'H', 'H', 'H']
    h3=['7', '7', '7', '7', '7', '7', '7']
    a1=['h', 'h', 'h', 'h', 'h', 'h', 'h']
    a2=['o', 'o', 'o', 'o', 'o', 'o', 'o']
    s1=['n', 'n', 'n', 'n', 'n', 'n', 'n']
    s2=['w', 'w', 'w', 'w', 'w', 'w', 'w']
    t1=['t', 't', 't', 't', 't', 't', 't']
    t2=['x', 'x', 'x', 'x', 'x', 'x', 'x']
    for i in range(len(h1)):
        x=x.replace(h3[i],h1[i])
        x=x.replace(h2[i],h1[i])
        x=x.replace(t2[i],t1[i])
        x=x.replace(s2[i],s1[i])
        x=x.replace(a2[i],a1[i])
    return x
df['posts'] = df['posts'].apply(lambda x: char_normalization(x))

```

Figure 5.3 Sample Code For Character Normalization

5.2.3 Tokenization

The data that has been cleaned and normalized should be tokenized so that the model could understand the context. To get the token or word of the text in the dataset, the NLTK python module is used. We used the python splits function and NLTK method for word tokenization. The output of this function is tokens or words, which is input for feature extraction methods or used by feature extraction methods.

The fit_on_texts Method from the Tokenizer class generates the vocabulary index based on word frequency. Next, the method texts_to_sequence creates a sequence of representative numbers based on the word index generated from the fit_on_texts method. Finally, as neural networks require inputs with the same shape and size, we need to pad the sequence. After the tokenization process, we got 179583 unique tokens that are used to create the embedding layer.

```

# The maximum number of words to be used. (most frequent)
#and Max number of words in each complaint.
MAX_NB_WORDS = 500000
MAX_SEQUENCE_LENGTH = 250
EMBEDDING_DIM = 300
tokenizer = Tokenizer(num_words=MAX_NB_WORDS,
                      filters='!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~', lower=True)
tokenizer.fit_on_texts(df['posts'].values)
word_index = tokenizer.word_index
print('Found %s unique tokens.' % len(word_index))
#Dependent and independent variable
X = tokenizer.texts_to_sequences(df['posts'].values)
X = pad_sequences(X, maxlen=MAX_SEQUENCE_LENGTH)
Y=df['label']

print('Shape of data tensor:', X.shape)
print('Shape of label tensor:', Y.shape)

```

Figure 5.4 Implementation For Tokenization

5.2.4 Implementation For Word2vec

After the data is preprocessed there should be a feature representation mechanism to represent the data as numeric vectors that is understandable by the machine as computers can understand only numbers. We used word embedding for the feature representation mechanism for this study. Pretrained word embeddings are a mechanism to build an embedding vector in which similar and related words are represented closely in the vector space. Word2Vec is one of the mechanisms to create pre-trained word embedding and it is a particularly computationally efficient predictive model for learning word embeddings from raw text. For the Afaan Oromo language, there are no pre-trained word embeddings available to use for weight initialization for the model that could enhance model performance. We trained a word2vec model using the posts and comments collected and saved the model that we used for initializing the weight of our model. To implement word2Vec we used a python Gensim module which is used to implement different embedding methods.

First import and instantiate word2Vec class with necessary parameters and builds the vocabulary, and training the Word2Vec model. The training resulted in a feature vector is also known as the embeddings. Embeddings are features that describe the target word. Finally, the weight of the trained word2vec model is loaded and used to initialize the weight of our deep learning models.

```
# train w2vec model
model = Word2Vec(sentences, size=300, sg=1, window=10, min_count=1)
# summarize vocabulary
words = list(model.wv.vocab)
# save model
model.save("C:/Users/hp/Desktop/Thesis/Word2Vec/w2vec_model.bin")
```

Figure 5.5 Sample Code For Training Word2vec

Table 5.1 Parameters For Training the Word2vec model

Size	Represent the vector dimension and we used 300 embedding dimensions. In case we have limited data, the size should be a much smaller value since you would only have so many unique neighbors for a given word.
Window	The window parameter represents the maximum distance between the target word and its neighboring word. If your neighbor's position is greater than the maximum window width to the left or the right, then, some neighbors would not be considered as being related to the target word. In theory, a smaller window should give you terms that are more related. Again, if your data is not sparse, then the window size should not matter too much, as long as it's not overly narrow or overly broad. If you are not too sure about this, just use the default value.
Min_count	The min_count represents the minimum frequency count of words. Words that do not satisfy the min_count would be ignored by the model.
Sg	Defines the training algorithm, that's used to define the algorithm whether it is a skip-gram or continuous bag of words model(CBOW. If sg=0 it means the model is CBOW, and sg=1 represents Skip-Gram Model. The CBOW model predicts the target word from the surrounding context words. The Skip-gram model is the inverse of CBOW. The Skip-Gram model predicts context words from the target words, training speed is lower than that of the CBOW model. Hence, we used the skip-gram model to create the word2vec model.

After the model is created and saved, it should be loaded, so that we can assign its weight to a variable that we can use for initializing the weight of the model.

```
#load w2vec
from gensim.models import Word2Vec
word_model = Word2Vec.load('c:users/hp/Desktop/Thesis/w2vec/w2vec_model.bin')
w2vecmodel = word_model.wv.syn0
print(w2vecmodel.shape)
```

Figure 5.6 Sample Code For Loading Word2vec and Assign The Weight to Variable

5.3 Model Implementation

After data preprocessing and embedding layer preparation, the next step is developing neural network models for the experiment. To implement the proposed models we have to Import the required packages from the sci-kit learning library.

```
import pandas as pd
import numpy as np
# text preprocessing
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import re
# plots and metrics
from sklearn import metrics
from keras.layers import Dense, Dropout, Activation, Embedding, Flatten
from keras.layers import Embedding, Flatten, Concatenate, Input, Dropout
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix
# preparing input to our model
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.layers import Embedding, Bidirectional, LSTM, GRU, Dense, SimpleRNN
from keras.callbacks import ModelCheckpoint, EarlyStopping
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import precision_recall_fscore_support
from sklearn.model_selection import KFold, cross_val_score
#from keras.utils import to_categorical
# keras layers
from keras.models import Sequential
from keras.layers import Embedding, Bidirectional, LSTM, GRU, Dense
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_recall_fscore_support
import string
import seaborn as sns
import tensorflow as tf
```

Figure 5.7 Importing Required Packages For Modeling

5.3.1 Recurrent Neural Networks

We implemented bidirectional RNN models (BiLSTM, and BiGRU) for RNN models, and we used Keras bidirectional(LSTM/GRU) layer for adding the recurrent layers. To prepare the embedding layer we used the Keras Embedding layer and we specified the required parameters.

`Embedding(MAX_NB_WORDS, EMBEDDING_DIM, weights=[w2vecmodel], trainable=True, input_length=X.shape[1])`. `MAX_NB_WORDS` is set to the size of vocabulary in the prepared hate speech dataset, `EMBEDDING_DIM` represents the embedding dimension and it is of the same size as our trained word2vec model which is 300. `Max_seq_len` is the maximum sequence length supported by the embedding layer which is input length. To add the recurrent layer we used `bidirectional(LSTM)` and `bidirectional(GRU)` Keras API to create bidirectional RNN models.

```
bilstm = Sequential()
bilstm.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM, weights=[w2vecmodel],
                    input_length=X.shape[1], trainable=True))
bilstm.add(SpatialDropout1D(0.5))
bilstm.add(Bidirectional(LSTM(128, return_sequences=True)))
bilstm.add(Bidirectional(LSTM(100)))
bilstm.add(Dropout(0.4))
bilstm.add(Dense(64))
bilstm.add(Dense(1, activation='sigmoid'))
bilstm.compile(loss='binary_crossentropy', optimizer='adam',
               metrics=['accuracy', keras.metrics.Precision(),
                       keras.metrics.Recall()])
```

Figure 5.8 Sample Code to Create Bidirectional LSTM Model

`bilstm.add(Bidirectional(LSTM(100, return_sequences=True)))`, this line of code is used to add a recurrent BiLSTM layer where 100 represents the number of hidden units, or nodes and `return_sequence` is set to `True` to enable adding more recurrent layers. To add Bi-GRU the LSTM layer is changed to GRU. To prevent the overfitting problem we added dropout to our models using `bilstm.add(Dropout(0.4))`, where 0.4 is the dropout rate. Dropout is used to randomly set the fraction rate of input units to 0 during training time, which is important to ignore less important elements and enhance model performance by reducing overfitting. However, the dataset contains unbalanced instances in which some instances are more redundant and they are less important, in such cases Dropout is not enough to prevent overfitting. Hence we used **SpatialDropout1D()** after the embedding layer. SpatialDropout is the same as Dropout but, it drops entire feature maps rather than individual elements, hence less important features are ignored.

5.3.2 Attention Mechanism

In the attention mechanism, unlike a simple encoder-decoder model the importance of each encoded vector is calculated and remembered during the decoding process. This helps the decoder to pick up only important vectors for decoding. For the attention model, we created an attention layer using `class Attention(Layer)` : unlike in the RNNs. The architecture of the attention-based model is the same as that RNN architecture except the attention layer is added and we have RNN decoders after the attention. We defined also the build and call functions in the attention class. Then, the layer is added using `model.add(Attention(return_sequences=True))`. The other parameters such as `SpatialDropout1D`, and `Dropout` are also used for the attention-based models same to the bidirectional RNNs.

5.3.3 Prototyping

After implementation and model evaluation are performed, the best-performing model is deployed on the webserver using FLASK API.

Afaan Oromo and Amharic Languages Hate Speech Detector

Users Post:

Bakka Kanatti Barreessaa/ ቋህፍን እዝህ ጋር ያስገቡ:

Check

Figure 5.9 Graphical User Interface

CHAPTER SIX

RESULTS AND DISCUSSION

In this chapter, the result of the experiments of the proposed solution for bilingual hate speech detection for Afaan Oromo and Amharic languages using deep learning is discussed. Finally, we discuss the significates of the result obtained by each experiment in this study.

6.1 Model Performance Results

The experiment involves developing six models and evaluating the models. We evaluated each model using 5 fold and 10 fold cross-validation to see the performance of models under both 5 fold and 10 fold testing. In addition to the proposed bidirectional recurrent neural networks and attention mechanism, two models LSTM and GRU are implemented and evaluated. The LSTM and GRU models are implemented and tested to compare their performance with the proposed models for the proposed bilingual hate speech detection problem. The word2vec model is applied for the embedding layer. The experiment results are given in **Table 6.1** and **Table 6.2**.

Table 6.1 Model Test Results Using 5-Fold Cross Validation

Model Test Results Using 5-Fold Cross-Validation				
Models	Evaluation Metrics			
	Accuracy(%)	Precision(%)	Recall(%)	F1-score(%)
LSTM	94	94	94	94
GRU	94	94	94	94
BiLSTM	94.3	94.2	94.2	94.2
Bi-GRU	94.2	94	94	94
BiLSTM+Attention	94	94	94	94
Bi-GRU+Attention	94.21	94.1	94.1	94.1

Table 6.2 Model Test Results Using 10-Fold Cross Validation

Model Test Results Using 10-Fold Cross-Validation				
Models	Evaluation Metrics			
	Accuracy(%)	Precision(%)	Recall(%)	F1-score(%)
LSTM	93.8	93.992	93.992	93.992
GRU	94.007	94	94	94
BiLSTM	94.2	94.16	94.16	94.16
Bi-GRU	94.1	94.105	94.105	94.105
BiLSTM+Attention	93.82	93.825	93.825	93.825
Bi-GRU+Attention	94.025	94	94	94

The experiment results confirm that using 5 fold cross-validation is slightly better than that of 10 fold for our problem. The test result using 10 fold cv shows that the BiLSTM performs best with an accuracy of 94.2 and an F1-score of 94.16. The second-best performing model is Bi-GRU with an accuracy of 94.1 and an F1-score of 94.105. The performance of the BiLSTM and BiGRU with attention mechanism didn't show any performance improvement. The accuracy and F1-score of BiLSTM are minimized to 93.8 and 93.899 respectively when adding attention mechanism to the model. For the Bi-GRU model when adding the attention layer the performance is comparable and didn't show a big difference, but still, the performance didn't exceed the BiGRU without attention. Further, we observed that using basic LSTM and GRU models cannot perform better than the proposed bidirectional RNNs and attention mechanism for the proposed system.

The test results using 5 fold cross-validation also show that the performance of BiLSTM exceeds the other models. Except for the GRU and BiLSTM+attention, all the other models performed better. The BiLSTM model accuracy and F1-score are improved to 94.3% and 94.2% respectively. The Bi-GRU+attention model is the second-best performing with an accuracy of 94.21% and an F1-score of 94.1%. Also, the result shows the performance of BiLSTM and BiGRU beats that of LSTM and GRU under 5 fold testing. Generally, the experiment confirms that the proposed bidirectional RNN models scored promising results than basic LSTM and GRU using both 5 fold and 10 fold testing. Further,

using 5-fold testing also made us advantageous by improving model performance relative to 10 fold testing.

6.1.1 BiLSTM Confusion Matrix

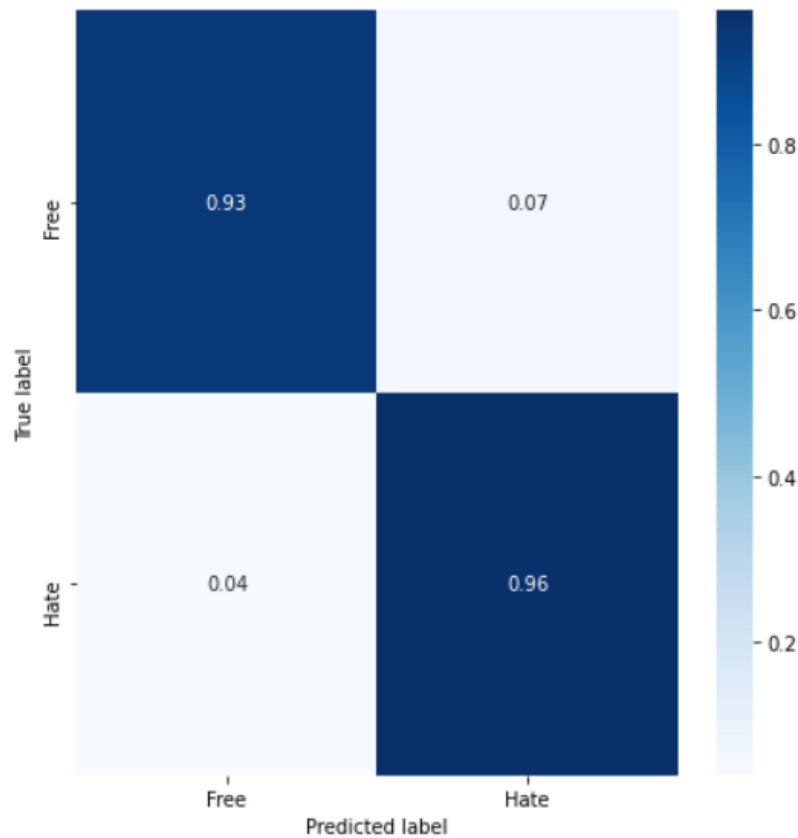


Figure 6.1 BiLSTM Confusion Matrix

The confusion matrix given in Figure 6.1 shows that the BiLSTM model correctly classifies 47,221 instances of the total 50,000 instances in the dataset and only 2,779 instances are misclassified. From the total of 25964 hate instances in the dataset, 96% are classified as true positive and 4% are false negatives. But, for the class “Free” 7% of the total 24036 instances are classified as false negatives and the left 93% of class “Free” instances are correctly classified.

6.1.2 Bi-GRU Confusion Matrix

The other proposed recurrent neural network model we implemented is bidirectional GRU. The model classified 97% of the “Free” instances as true positives and 3% of them as false negatives. For the “Hate” instances, 92% of them are classified as true positives and 8% as false negatives from the total 25964 instances of “Hate” in the dataset.

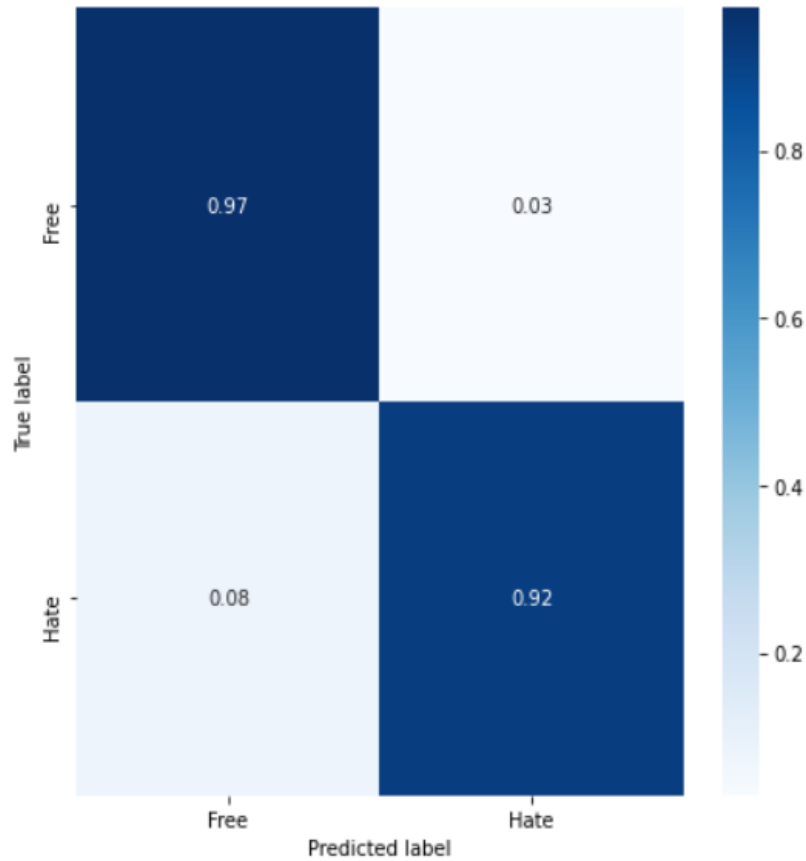


Figure 6.2 Bi-GRU Confusion Matrix

Generally, the model correctly classified 47,202 instances from the whole dataset. The model is a little bit more confused than the bidirectional long short-term memory as it misclassified 2798 instances while the BiLSTM model misclassified only 2779 instances.

6.1.3 Performance of Attention Based Models

Then, we applied the attention mechanism by adding an attention layer to both bidirectional RNN models, BiLSTM and BiGRU. Using BiLSTM+Attention, 97% of the total 25964 “Hate” instances in the dataset are classified correctly and the left 3% are false negatives. But, for “Free” instances, only 91% are true positives, and the rest 9% are false negatives as classified by the BiLSTM+Attention model.

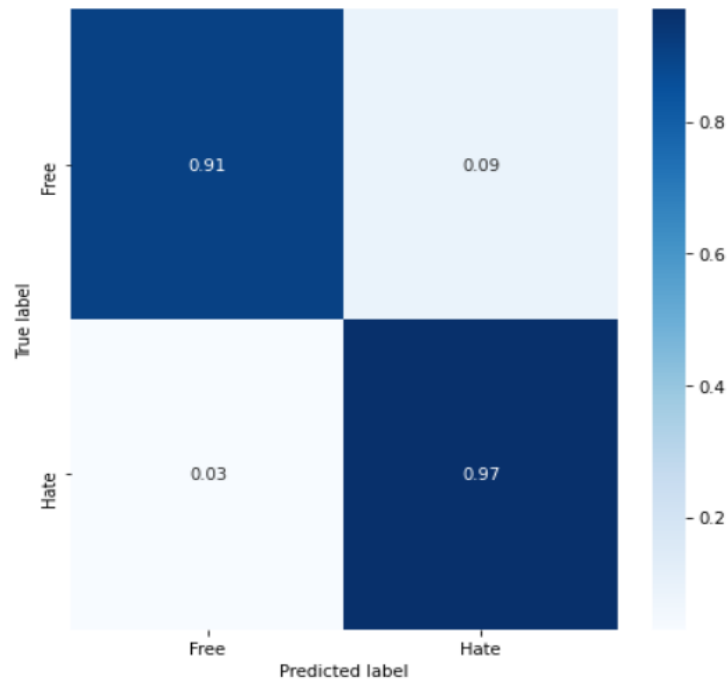


Figure 6.3 Confusion Matrix for BiLSTM+Attention

The Bi-GRU+Attention model is the second-best performing model when evaluated by 5 fold cross-validation next to the BiLSTM model. The model classified 96% of the total 24,036 “Free” instances in the dataset as true positives and only 4% of them are classified as false negatives. But, for the class “Hate” from the total of 25,964 instances, 93% of them are classified as true positives and the rest are false negatives.

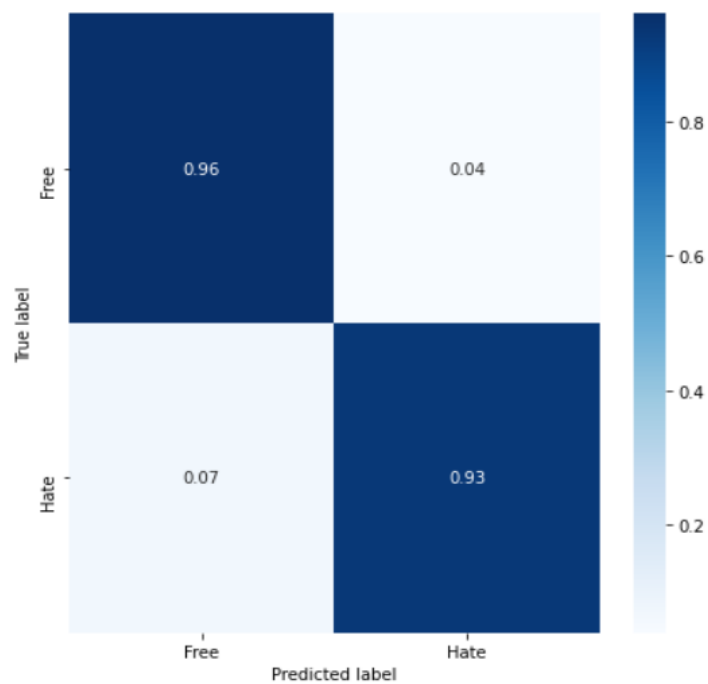


Figure 6.4 Confusion Matrix for Bi-GRU+Attention

To compare the performance of the proposed models with and without the proposed skip-gram word2vec model, the models are trained also without the word2vec. The overall summary of classification performances of the proposed models using word2vec and without word2vec in terms of accuracy, F1_score, precision, and recall based on 5 fold testing is given in the table below:

Table 6.3 Summary of Models Performance Using Word2vec and Without Using Word2vec

Models Performance using Word2vec				
Models	Accuracy	Precision	Recall	F1-score
BiLSTM	94.3	94.2	94.2	94.2
Bi-GRU	94.2	94	94	94
BiLSTM +Attention	94	94	94	94
Bi-GRU+Attention	94.21	94.1	94.1	94.1
Models Performance Without Using Word2vec				
BiLSTM	94.2	94	94	94
Bi-GRU	94	94	94	94
BiLSTM+Attention	93.89	93.89	93.89	93.89
Bi-GRU+Attention	94	94	94	94

The experiment results presented in (Table 6.3) confirm that applying the proposed word2vec model for feature representation slightly improved the performance. From the summary of the model's performance above, we observed that BiLSTM with word2vec performance is better than the other models. The BiLSTM best performed with an accuracy of 94.3% and the model BiGRU+Attention achieved an accuracy of 94.21%, which is the second-best performing model.

6.2 Hyperparameter Tuning

The main goal of tuning hyperparameters is to find the optimum hyperparameter that results in low validation error and high model performance. Hyperparameters can be tuned automatically but, it needs huge computational resource. Hence, we used manual hyperparameter tuning. Manual hyperparameter tuning is a trial and error process. Since there is no efficient algorithm to select

optimum hyperparameter the process requires adjusting several trials to find the parameter combination that provides low validation error. Manually we adjusted hyperparameters and prepared 15 trials for the BiLSTM model. The BiLSTM model is trained and tested on the 15 trials of the parameter combinations. We observed that changing the number of batch sizes and increasing the number of epochs did not show performance improvement. But, selecting the right optimizer and learning rate value shows a significant difference in model performance. For example, using SGD and RMSprop optimizers by adjusting the other parameter the same, the SGD performs very lower than that of RMSprop. Learning rate also has a great role in performance improvement. Selecting an optimal learning rate like 0.001, show good performance rather than using too small a value or large value for learning rate.

Table 6.4 Hyperparameter Tuning Trials

Trial	Hyperparameters				
	Dropout	Optimizer	Batch Size	Epochs	Learning Rate
1	0.5	RMSprop	256	30	0.002
2	0.5	RMSprop	64	15	0.001
3	0.4	RMSprop	256	10	0.0001
4	0.5	Adamax	128	20	0.0001
5	0.5	Adamax	256	30	0.002
6	0.2	Adamax	128	5	0.1
7	0.5	Adam	256	15	0.01
8	0.4	Adam	32	10	0.1
9	0.5	Adam	256	30	0.002
10	0.5	Adadelata	128	7	0.01
11	0.2	Adam	128	10	0.0001
12	0.4	Adadelata	64	15	0.01
13	0.5	Adadelata	256	30	0.002
14	0.5	SGD	256	25	0.00001
15	0.2	SGD	128	10	0.0001

As shown in (Figure 6.5), trial 1 and trial 9 achieved performance first obtained by the BiLSTM model. But, no trial results exceed the performance first achieved by the BiLSTM model.

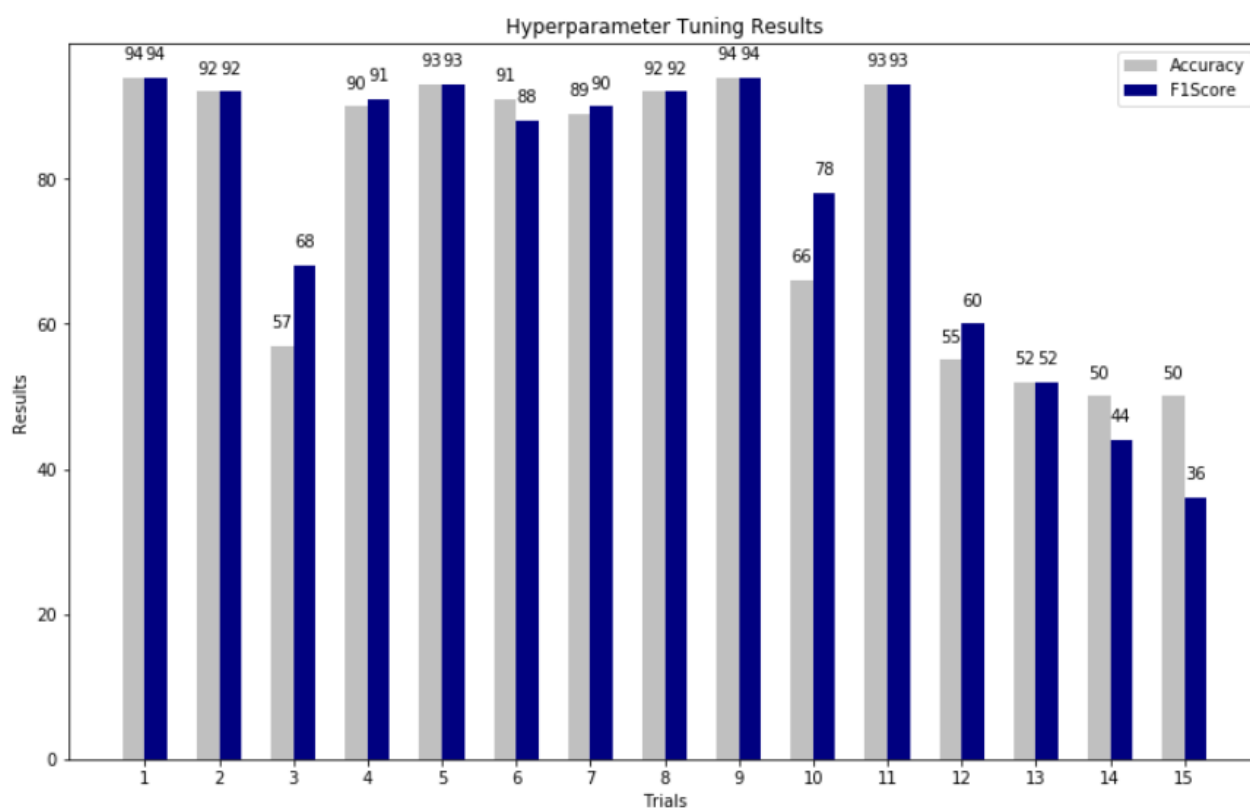


Figure 6.5 Hyperparameter Tuning Results

In addition to the BiLSTM model, the BiGRU+Attention is also implemented by using the parameter combination of trial 1 and trial 9 for model comparison. The BiGRU+Attention model is the second-best performing model as presented by the experiment above.

Table 6.5 Comparison of The Proposed BiLSTM and BiGRU+Attention Models

Models	Trial 1		Trial 9	
	Accuracy(%)	F1-score(%)	Accuracy(%)	F1-score(%)
BiLSTM	94	93.846	94.2	94.2
BiGRU+Attention	93.56	93.56	94	94

From the comparison of BiLSTM and attention-based BiGRU models, we observe that the performance of the BiLSTM model stayed unbeaten by BiGRU+Attention. Finally, the experiment confirms that the BiLSTM model performs best for the proposed bilingual social media hate speech detection on the prepared bilingual dataset.

6.3 Model Testing Using New Input Data

To evaluate the model prediction, new posts and comments are collected. The data for evaluation is collected from five peoples where each of them are instructed to prepare 10 instances; five Afaan Oromo and five Amharic, from Facebook posts and comments. As a result, we got 50 new instances. We observed that only five instances are misclassified by the model after checking the model prediction for all 50 instances. Observing this, we tried error analysis to identify the reason for model misclassification. There are some reasons for misclassification that we identified. The first challenging problem we observed is that peoples use the name of ethnic group, name of religion, and other names of the protected category to offend the targeted group when they post hate speech. Therefore most of the hate speech texts in the training data contain the name of the protected identity of the targeted group.

Hence, the model sometimes classifies such texts as hate speech. The other one is misclassification due to abusive words in the text. Peoples use abusive words mostly to offend the targeted group. The hate speech dataset contains a lot of abusive words which are labeled as hate speech. Due to that the model mostly classifies text containing abusive words as hate speech even if it is written in another context. The dataset is also limited to social media data. Therefore lack of a huge dataset other than social media data is another problem making the model unable to correctly classify some instances.

6.4 Discussions

The main goal of this study is to develop a deep learning model for bilingual hate speech detection for Afaan Oromo and Amharic texts. To the best of our knowledge, this study is the first to propose bilingual social media hate speech detection for Afaan Oromo and Amharic languages using deep learning approaches. To implement the models we used a bilingual dataset prepared from newly collected Afaan Oromo texts from the Facebook platform and the Amharic hate speech dataset prepared by (Getachew, 2020). Before the model implementation, we applied text preprocessing including cleaning and normalization. We normalized Amharic characters those having the same meaning but spelled differently. Also as we stated in (4.2.2 Normalization), there are ambiguous Amharic words in addition to characters, and we applied also the normalization for them. We used

word embeddings and we trained word2vec for the feature representation to develop the models. The models are evaluated based on the proposed evaluation metrics (3.4.2 Model Evaluation Methods).

Six models are implemented and evaluated using the prepared bilingual dataset. Looking for the advantages of using bidirectional RNN models to store past and future information, we implemented BiLSTM and BiGRU models. In addition to that, attention-based BiLSTM and attention-based BiGRU are implemented expecting the advantages of using an attention mechanism that helps to calculate the significance of each input as stated in (Sun & Lu, 2020). The LSTM and GRU models are also implemented to compare their performance with the proposed models. First, the models are tested using 5 fold and 10 fold cross-validation. Best model performance is obtained using the BiLSTM model using 5 fold evaluation. The experiment shows that using 5-fold is better than using 10-fold for our bilingual dataset. The experiment results using 5 fold testing shows that the BiLSTM model with word2vec performed better on our bilingual dataset (Table 6.3). The attention-based BiGRU model is the second-best performing model. For the BiLSTM adding the attention layer, unable to outperform the BiLSTM baseline while the BiGRU+attention performed slightly better than the BiGRU. Based on the evaluation results, the BiLSTM performed better than all the other models.

For performance improvement and to mitigate the problem of overfitting we used SpatialDropout1D and a Dropout rate of 0.5. Also, EarlyStopping is applied during model training to enabling the model to stop learning at optimum epochs. Also, we prepared a trained word2vec model based on the skipgram model. The models are implemented using the trained word2vec and without using the trained word2vec, the result shows that the developed word2vec model enhanced the model performance slightly (Table 6.3). Further, hyperparameter tuning is performed to come up with the optimum hyperparameter combination providing high accuracy and low validation error. To do that, the selected BiLSTM is tested on 15 trials of parameter combination. Since hyperparameter tuning requires more GPU devices and is time taking process only 15 trials are manually adjusted. By testing the model performance using the 15 parameter combination trials, no trial result exceeds the performance first obtained by the BiLSTM model. But, the best performance is obtained using trials 1 and 9 which is accuracy and f1_score of 94%, and that is the same as the performance first achieved by the BiLSTM model.

To compare the model performance, the BiGRU+attention model is selected, which is the second-best performing model and compared with the BiLSTM model. The attention-based BiGRU is trained using trial 1 and trial 9 parameter combinations. The result shows that the performance of the BiLSTM model stays better for the prepared bilingual dataset.

Finally, the accuracy of 94.3% and an f1 score of 94.2% achieved by the BiLSTM model stayed the best performance for the proposed bilingual social media hate speech detection for Afaan Oromo and Amharic languages. The prototype is developed using the BiLSTM model.

We developed GUI for the prototype and deployed the model on a webserver using FLASK API. Then, the error analyses process is performed based on newly collected 50 instances to test model predictions using new input data and identify the reason for misclassification. Most misclassification problem is related to the effect of abusive words. Since the hate speech dataset contains a lot of abusive words the model mostly understands any text with abusive words as hate speech. Also, the other big challenge is there are names of the protected identity that come with hate speech in the training data. Hence, the model understands and classifies such words as hate speech. Classifying hate speech correctly requires huge training data with correct labels.

CHAPTER SEVEN

CONCLUSION AND FUTURE WORKS

7.1 Conclusion

This study proposed bilingual social media texts hate speech detection for Afaan Oromo and Amharic languages using deep learning approaches. To accomplish this work first the bilingual dataset is prepared using Afaan Oromo and Amharic texts. Different preprocessing techniques are applied to the dataset before representing it as a feature vector. Word embedding is used for feature representation to develop the proposed models. Word embedding is selected since it can represent words based on their semantic contexts. We prepared also a trained word2vec using the Amharic and Afaan Oromo texts. The word2vec is trained using the skip-gram model, which is good for modeling non-frequent words and even if there is no huge size data.

For the experiment, six models are developed using the bilingual dataset. The LSTM and GRU algorithms are implemented in addition to the proposed bidirectional RNNs and attention mechanism for model comparison. Also, the models are compared by using the trained word2vec and by training models without using our trained word2vec. The BiLSTM model with word2vec outperformed all models by evaluating models using 5 fold cross-validation. The BiLSTM model achieved accuracy and f1-score of 94.3% and 94.2% respectively. Despite its advantages of focusing on important keywords, adding the attention mechanism to the BiLSTM model is unable to outperform the BiLSTM baseline. But, the Bi-GRU model with an attention mechanism shows slightly better performance than the BiGRU baseline. The experiment shows that the Bi-GRU with attention mechanism performed best next to the BiLSTM model while the performance of the LSTM and GRU was unable to beat the proposed BiRNNs and attention mechanism.

Hyperparameter tuning is performed by adjusting 15 trials for hyperparameter combinations. In the tuning process, we were able to see the effect of different hyperparameters on model performance. The tuning process is done for the BiLSTM model which outperformed the others. Even if we did not get significant improvement, two trials able to achieve comparable performance to the performance first obtained by the BiLSTM. By selecting the two best trials we trained the attention-based Bi-GRU, which is second best-performing model, and compared it with the BiLSTM. Finally, the study got BiLSTM is the best performing model for the proposed bilingual hate speech detection of Afaan Oromo and Amharic language texts using our dataset. Also, the study proved that using pre-trained word2vec models is useful for performance improvement.

7.2 Future Works

Although the study implemented and conducted an experiment on recurrent neural networks and created a baseline attention mechanism for bilingual hate speech detection, future research can include custom large pre-trained word embedding using social media data for further performance improvement. Since deep learning models like CapsNet also achieving good performance in NLP tasks, ensemble deep learning models can be implemented for the problem by integrating the BiLSTM model with those models. Further, the BiLSTM model can be tested on non-textual data such as audio and video by future researchers for the problem of hate speech detection. Also, the model can be tested on multilingual data by preparing datasets for other languages in Ethiopia other than Amharic and Afaan Oromo.

This research was funded by Adama Science and Technology University with a grant number ASTU/SM-R/224/21

REFERENCES

- Ababa, A. (2020). *Bidirectional Amharic-Afaan Oromo Machine Translation Using Hybrid Approach Gelan Tulu Heyi A Thesis Submitted to the Department of Computer Science in Partial Fulfilment for the Degree of Master of Science in Computer Science* (Issue March). AAU.
- Agrawal, N. (2021). *Understanding Attention Mechanism: Natural Language Processing*. <https://medium.com/analytics-vidhya/https-medium-com-understanding-attention-mechanism-natural-language-processing>
- Akram, W., & Kumar, R. (2017). A Study on Positive and Negative Effects of Social Media on Society. *International Journal of Computer Sciences and Engineering*, 5(10), 351–354. <https://doi.org/10.26438/ijcse/v5i10.351354>
- Al-Hassan, A., & Al-Dossari, H. (2019). *Detection of Hate Speech in Social Networks: a Survey on Multilingual Corpus*. 83–100. <https://doi.org/10.5121/csit.2019.90208>
- Alhogbi, B. G. (2017). Tracking hatred: An international dialogue on hate speech in the social media. *Journal of Chemical Information and Modeling*, 53(9), 21–25.
- Aljarah, I., Habib, M., Hijazi, N., Faris, H., Qaddoura, R., Hammo, B., Abushariah, M., & Alfawareh, M. (2020). Intelligent detection of hate speech in Arabic social network: A machine learning approach. *Journal of Information Science*. <https://doi.org/10.1177/0165551520917651>
- Aluru, S. S., Mathew, B., Saha, P., & Mukherjee, A. (2020). *Deep Learning Models for Multilingual Hate Speech Detection*. 1–16. <http://arxiv.org/abs/2004.06465>
- Argaw, M., & A, Abebe, D. S. L. (2019). *Amharic Parts-of-Speech Tagger using Neural Word Embeddings*. ADDIS ABABA UNIVERSITY.
- Aulia, N. (2019). *Hate Speech Detection on Indonesian Long Text Documents Using Machine Learning Approach*. 164–169.
- Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). *Deep Learning for Hate Speech Detection in Tweets **. August 2020, 7–9. <https://doi.org/10.475/123>
- Bahdanau, D., Cho, K. H., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.
- Baweke, E. (2020). *AMHARIC TEXT HATE SPEECH DETECTION IN SOCIAL MEDIA USING DEEP LEARNING APPROACH*. BAHIR DAR UNIVERSITY.
- Bertoni, E. (2006). Hate Speech Under The American Convention On Human Rights. *ILSA Journal of*

International and Comparative Law, 12(2), 569–574.

BIRHANE TESFAYE, Y. A. (2017). *Afaan Oromo Word Sense Disambiguation Using WordNet*. AAU.

Clement, J. (2020). Global digital population as of October 2020. *Statista.Com*, 212, 49016428.

Colah. (2021). *Understanding LSTM Networks*. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Del Vigna, F., Cimino, A., Dell’Orletta, F., Petrocchi, M., & Tesconi, M. (2017). Hate me, hate me not: Hate speech detection on Facebook. *CEUR Workshop Proceedings*, 1816(January), 86–95.

Faris, H., Aljarah, I., Habib, M., & Castillo, P. A. (2020). Hate speech detection using word embedding and deep learning in the Arabic language context. *ICPRAM 2020 - Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods*, *Icpram*, 453–460. <https://doi.org/10.5220/0008954004530460>

Fortuna, P., & Tec, I. (2020). *A Survey on Automatic Detection of Hate Speech in Text*. 51(4).

Gambäck, B., & Sikdar, U. K. (2017). *Using Convolutional Neural Networks to Classify Hate-Speech*. January, 85–90. <https://doi.org/10.18653/v1/w17-3013>

Gaydhani, A., Doma, V., Kendre, S., & Bhagwat, L. (2018). *Detecting Hate Speech and Offensive Language on Twitter using Machine Learning : An N-gram and TFIDF based Approach*.

Genemo, A. S. (2015). *School of Graduate Studies College of Natural Sciences Department of Computer Science Afaan Oromo Named Entity Recognition Using School of Graduate Studies College of Natural Sciences Department of Computer Science* (Issue March). AAU.

Getachew, S. (2020). *Amharic Facebook Dataset for Hate Speech detection*. <https://doi.org/10.17632/ymtmxx385m.1>

Hassan, A., & Mahmood, A. (2017). Efficient deep learning model for text classification based on recurrent and convolutional layers. *Proceedings - 16th IEEE International Conference on Machine Learning and Applications, ICMLA 2017, 2017-Decem*(February 2018), 1108–1113. <https://doi.org/10.1109/ICMLA.2017.00009>

Kenenisa, Y., & Melak, T. (2019). Hate Speech Detection for Amharic Language on Social Media Using Machine Learning Techniques. In *Hate Speech Detection for Amharic Language on Social Media Using Machine Learning Techniques By:* (Issue September). ASTU.

Liu, Z., Huang, H., Lu, C., Lyu, S., & Tao, Z. (2020). Text classification based on Multi-granularity attention Hybrid Neural Network. *ArXiv*, 1–15.

Mossie, Z., & Wang, J.-H. (2018). *Social Network Hate Speech Detection for Amharic Language*. 41–

55. <https://doi.org/10.5121/csit.2018.80604>

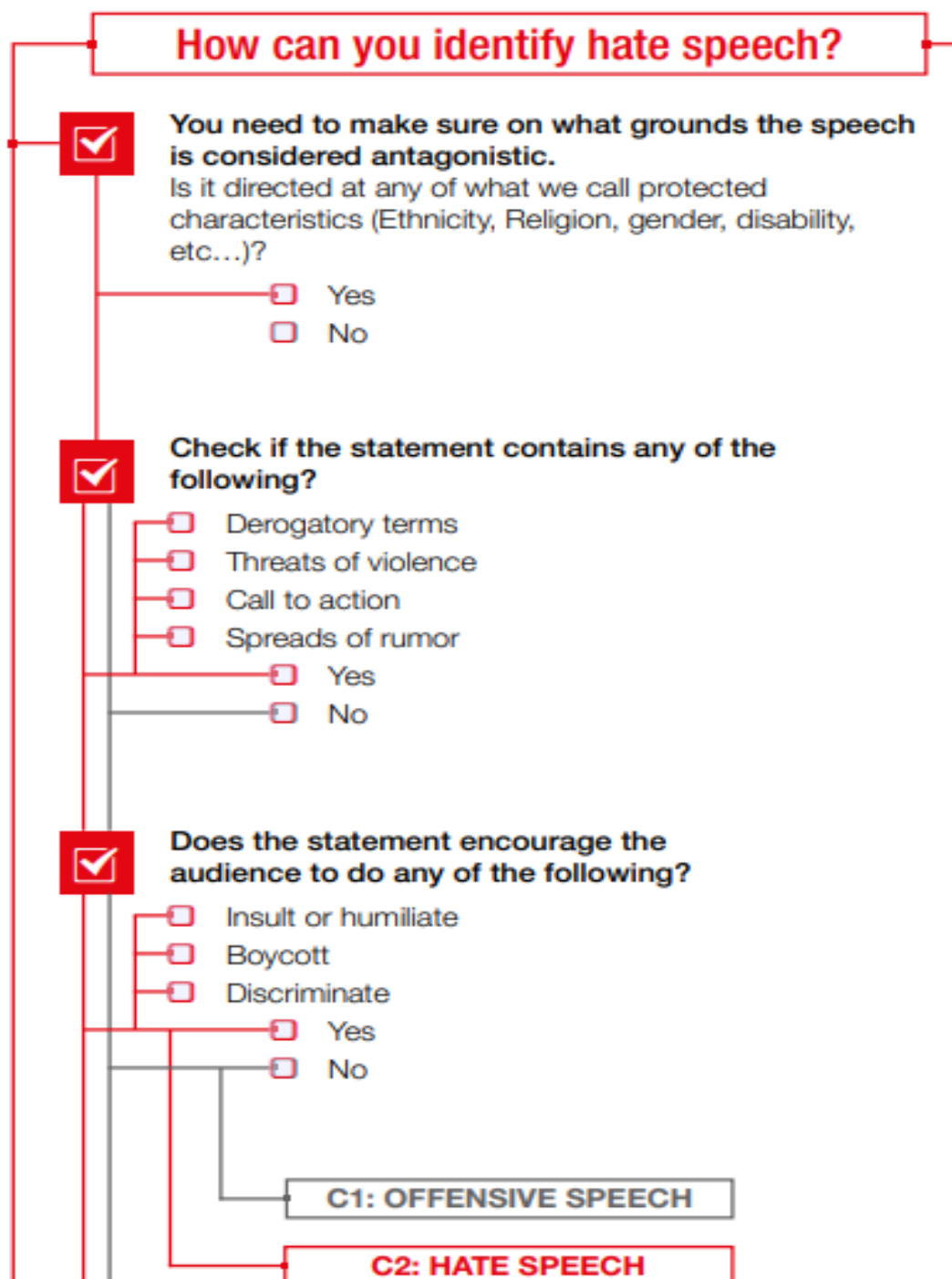
- Mousa, A. E. D., & Schuller, B. (2017). Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis. *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference, 1*, 1023–1032. <https://doi.org/10.18653/v1/e17-1096>
- Mujtaba, H. (2020). *Embedding | GloVe What is Word Embedding ? Why Word Embeddings are used ?* <https://www.mygreatlearning.com/blog/word-embedding/>
- Mutuvi, S. (2019). *Introduction to Machine Learning Model Evaluation | by Steve Mutuvi | Heartbeat.* <https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f>
- Nayel, H. A., & Shashirekha, H. L. (2019). DEEP at HASOC2019 : A machine learning framework for hate speech and offensive language detection. *CEUR Workshop Proceedings, 2517*(December 2019), 336–343.
- Pavithra, M. A., & Prakash, M. S. M. (2019). TensorFlow in Deep learning. *International Journal of Innovative Research in Technology*, 5(9), 142–147.
- Qian, L. I., Peng, H., Jianxin, L. I., Congyin, X. I. A., Lifang, H. E., Lichao, S. U. N., Yang, R., & Philip, S. Y. U. (2020). A Survey on Text Classification: From Shallow to Deep Learning. *ArXiv*, 37(4).
- Rushdy, A. (2021). *Aadhil Rushdy Word Embeddings for Sentence Classification*. 1–6.
- Saksesi, A. S., Nasrun, M., & Setianingsih, C. (2018). Analysis Text of Hate Speech Detection Using Recurrent Neural Network. *Proceedings - 2018 International Conference on Control, Electronics, Renewable Energy and Communications, ICCEREC 2018*, 242–248. <https://doi.org/10.1109/ICCEREC.2018.8712104>
- Salehinejad, H., Sankar, S., Barfett, J., Colak, E., & Valaee, S. (2017). Recent advances in recurrent neural networks. *ArXiv*, 1–20.
- Sun, X., & Lu, W. (2020). *Understanding Attention for Text Classification*. 1999, 3418–3428. <https://doi.org/10.18653/v1/2020.acl-main.312>
- Sutejo, T. L., & Lestari, D. P. (2019). Indonesia Hate Speech Detection Using Deep Learning. *Proceedings of the 2018 International Conference on Asian Language Processing, IALP 2018*, 39–43. <https://doi.org/10.1109/IALP.2018.8629154>
- Tesfaye, S. G., & Tune, K. K. (2020). *Automated Amharic Hate speech Posts and Comments Detection Model using Recurrent Neural Network*.
- Tiruneh, Mulat Getaneh, Y. A. (PhD), & This. (2020). *Amharic WordNet construction Using Word*

Embedding. Addis Ababa University.

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention is all you need*. Advances in Neural Information Processing Systems.
- Wiratunga, N. (2000). *Supervised vs . Unsupervised Learning Contents*. 1–9.
- Wubshet, D., & Ejigu, D. (2014). *Bidirectional English – Afaan Oromo Machine Translation Using Hybrid Approach* (Issue November). AAU.
- Yao, X. (2017). Attention-based BiLSTM neural networks for sentiment classification of short texts. *Proceedings of Science, 2017-Decem*(December 2017), 1–8. <https://doi.org/10.22323/1.300.0014>
- YARED YENEALEM AKLILU. (2019). *Exploring Neural Word Embeddings*.
- ZEWUDIE, S. (2019). Hate Speech and Disinformation Prevention and Suppression Proclamation. *Proclamation, Labour No.1156/2019 Labour*, 35.
- Zhang, Z. (1900). *Hate Speech Detection : A Solved Problem ? The Challenging Case of Long Tail on Twitter*. 1(0), 1–5.

APPENDICES

APPENDIX A: ANTI-HATE SPEECH GUIDELINE





Check if the statement contains any of the following and if the audience have the means to carryout violence.?

- ☐ Riot
- ☐ Loot
- ☐ Beat
- ☐ Evict
- ☐ Kill
- ☐ Calls for other forms of physical violence

☐ Yes

☐ No

C3: DANGEROUS SPEECH WITH LIMITED RISK



Is the target able to call upon institutions?

☐ Yes

☐ No

C3: DANGEROUS SPEECH WITH LIMITED RISK

C4: DANGEROUS SPEECH WITH RISK OF VIOLENCE

APPENDIX B: LIST OF STOPWORDS

Appendix B.1 Sample Amharic Stopwords

እኔ	የእነሱ	ማድረግ	ከ	እንደዚህ
የእኔ	ራሳቸው	ሀ	ወደ ላይ	ብቻ
እኔ ራሴ	ምንድን	አንድ	ታች	የራሱ
እኛ	የትኛው	የ	ውስጥ	ተመሳሳይ
የእኛ	ማን	እና	ውጭ	ስለዚህ
እኛ ራሳችን	ይህ	ከሆነ	በላይ	ይልቅ
አንቺ	የሚል ነው	ወይም	እንደገና	እንዲሁ
ነህ	ያ	ምክንያቱም	ተጨማሪ	ት
አላችሁ	እነዚህ	እንደ	ከዚያ	ይችላል
እርስዎ	እነዚያ	እስከ	አንድ ጊዜ	ይገባል
ትፈልጋለህ	ነኝ	እያለ	እዚህ	ይገባኛል
ያንተ	ናቸው	በ	እዚያ	አሁን
ራስህን	ነበር	ለ	መቼ	መ
እራሳችሁ	ነበሩ	ጋር	የት	ም
እሱ	ሁን	ስለ	ለምን	ኦ
የእሱ	ቆይቷል	ላይ	እንዴት	ዳግም
ራሱ	መሆን	መካከል	ሁሉም	መሆን
እሷ	አላቸው	ወደ	ማንኛውም	ሁለ
እሷናት	አለው	በኩል	ሁለቱም	ሁለም
የእሷ	ነበረው	ወቅት	እያንዳንዳቸው	ሕዝብ
እራሷ	ያለው	ከዚህ በፊት	ጥቂቶች	ሀሙስ
ነው	መስራት	በኋላ	በጣም	ለመሆ
እነሱ	ያደርጋል	ከላይ	ሌላ	
እነሱን	አደረገ	ከታች	አንዳንድ	

Appendix B.2 Sample Afaan Oromo Stopwords

aanee	sana	sun	tun	yommuu
gidduu	nuti	an	keenya	eegasii
itti	ala	isaa	utuu	isii
narraa	illee	kee	booddee	malee
akka	kan	tanaaf	iseen	yoo
gubbaa	siin	ani	keessatti	fi
ittuu	alatti	isaaf	waan	isin
natti	immoo	keenna	dura	na
akkam	kana	tanaafuu	ishii	yookaan
hanga	silaa	ati	kiyya	gama
jala	amma	isaanirraa	warra	isiin
nu	inni	keessa	duuba	naaf
akkasumas	kanaafi	ta’ullee	ishiif	yoom
henna	kanaaf	bira	koo	kun
jara	sitti	isatti	yeroo	
nurraa	ammo	keessan	eega	
akkuma	irra	teenya	ishiirraa	
hogguu	kanaafuu	booda	kun	

APPENDIX C: SAMPLE HATEFUL KEYWORDS IDENTIFIED BY CARD

No.	Term Or Phrase	Language	Transliteration	Meaning or Context
1	ጋላ	Amharic	Galla	It is a derogatory name/identifier against the ethnic Oromo of Ethiopia. It used to be an official reference of Oromos before 1974 and banned since then.
2	nafxanya	Afaan Oromo	Neftegna	It means gun-man in Afaan Oromo but is mostly used against ethnic Amharas in prejudice. It is historically used to refer to people who have conquest today's southern Ethiopia by then Abyssinia.
3	ሰፋሪ	Amharic	Sefari	It means 'Settler' and is a derogatory term referring to Amharas based on the belief that Amharas settled "on the lands" of non-Amharas
4	qomcee	Afaan Oromo	komche	A Derogatory word against Amhara ethnic group usually used with derogatory attack.
5	የአህያ ዘር	Amharic	Yeahya zer	It means 'Donkey race' used against Amharas. Tigreans use "adgi" (አድጊ) Tigrigna equivalent for donkey to refer to Amharas.
6	ትምክህተኛ	Amharic	Timkehetegna	It is a politically loaded term and is usually used against Amhara ethnic members.
7	መንጋ	Amharic	Menga	It means Herd and also MOB.
8	ቆማጣ	Amharic	Komata	It's an Afaan Oromo/ Amharic word which means whose literal meaning is 'Leprosy infected'
9	Qurcii	Afaan Oromo	Qurchi	It's an Afaan Oromo word which means whose literal meaning is 'Leprosy infected'
10	ባንዳ	Amharic	Banda	it is a word currently used by every group to refer/label people as sell outs to other groups.
11	Maxantuu	Afaan Oromo	Mutantu	It means Parasite and is often used to people who work or have any cooperation with non oromos
12	ዲቃላ	Amharic	Dikala	Bastard
13	አህባሽ	Amharic	Ahbash	It is one sect of Muslim religion, usually Sunni Muslims used this term to refer one groups or individuals whom they think they are less Muslim

APPENDIX D: MODEL SCORES PER-FOLD

BiLSTM

Score per fold

> Fold 1 - Loss: 0.16452501714229584 - Accuracy: 94.1100001335144%

> Fold 2 - Loss: 0.17594434320926666 - Accuracy: 94.20999884605408%

> Fold 3 - Loss: 0.15561871230602264 - Accuracy: 94.37999725341797%

> Fold 4 - Loss: 0.1792638599872589 - Accuracy: 93.48000288009644%

> Fold 5 - Loss: 0.15600022673606873 - Accuracy: 94.34999823570251%

Average scores for all folds:

> Accuracy: 94.10599946975708 (+- 0.3278151274976172)

> Loss: 0.16627043187618257

> Precision: 94.10599946975708

> Recall: 94.10599946975708

BiLSTM+Attention

Score per fold

> Fold 1 - Loss: 0.17412269115447998 - Accuracy: 93.790000067710876%

> Fold 2 - Loss: 0.17018628120422363 - Accuracy: 93.8700020313263%

> Fold 3 - Loss: 0.18481707572937012 - Accuracy: 93.91999840736389%

> Fold 4 - Loss: 0.16738387942314148 - Accuracy: 94.27000284194946%

> Fold 5 - Loss: 0.17250044643878937 - Accuracy: 94.24999952316284%

Average scores for all folds:

> Accuracy: 94.02000069618225 (+- 0.20039989985124362)

> Loss: 0.17380207479000093

> Precision: 94.02000069618225

> Recall: 94.02000069618225

Bi-GRU

Score per fold

> Fold 1 - Loss: 0.15996244549751282 - Accuracy: 93.97000074386597%

> Fold 2 - Loss: 0.1500823050737381 - Accuracy: 94.19000148773193%

> Fold 3 - Loss: 0.14019903540611267 - Accuracy: 94.30000185966492%

> Fold 4 - Loss: 0.1833246648311615 - Accuracy: 93.9300000667572%

> Fold 5 - Loss: 0.16476605832576752 - Accuracy: 94.23999786376953%

Average scores for all folds:

> Accuracy: 94.12600040435791 (+- 0.14840491385901095)

> Loss: 0.15966690182685853

> Precision: 94.12600040435791

> Recall: 94.12600040435791

BiGRU+Attention

Score per fold

> Fold 1 - Loss: 0.13703685998916626 - Accuracy: 94.2900002002716%

> Fold 2 - Loss: 0.15137869119644165 - Accuracy: 94.44000124931335%

> Fold 3 - Loss: 0.16046537458896637 - Accuracy: 94.01999711990356%

> Fold 4 - Loss: 0.15598241984844208 - Accuracy: 94.12000179290771%

> Fold 5 - Loss: 0.15417274832725525 - Accuracy: 94.08000111579895%

Average scores for all folds:

> Accuracy: 94.19000029563904 (+- 0.15388381817127872)

> Loss: 0.15180721879005432

> Precision: 94.19000029563904

> Recall: 94.19000029563904

APPENDIX E: GRAPHICAL USER INTERFACE

Appendix E.1 Predicting Texts as “Hate”

Afaan Oromo and Amharic Languages Hate Speech Detector

Dubbii Jibbiinsaati/ የጥላቻ ንግግር ነዉ

Users Post:

እረ ወያኔ አተዋሽ ምን አቅም ኑሮሽ ነዉ ተቆጣጠርኩ ምትይዉ አንች ቅማላም የ ቆማጣ ዘር መጀመሪያ አቅምሽን እወቁ አሁንም 1970ዎች ላይ ናችዉ እንዴ እር ንቁ እንናተ ማትረቡ ትግሬዎች ሰልጥኑ እስኪ ያለነዉ 22 ክፈለ ዘመን ላይ ነን ማለትም የሰለጠነ ህዝቦች ነን እና መቸም ቢሆን የትም አድርሱም በቅረቡ ትጠፋላችዉ ሁላችሁም

Check

Afaan Oromo and Amharic Languages Hate Speech Detector

Dubbii Jibbiinsaati/ የጥላቻ ንግግር ነዉ

Users Post:

Warra qalmaaf qophaa'e kana juntaan haa fixxu maaltu nu dhibe tokkummaa biyyaaf osoo hin taane garaa nama tokkootiif jecha tigireefi amaarri wal ajjeesaa jirti jarri lamaanuu nuuf diina nun ilaallatu

Check

Appendix E.2 Predicting Texts as “Free”

Afaan Oromo and Amharic Languages Hate Speech Detector

Dubbii Jibbiinsaa Miti/ የጥላቻ ንግግር አይደለም

Users Post:

ኪዳነምህረት ከክፉ ትሰውራችሁ መላውን የ ኢትዮጵያ ህዝብ ትጠብቀው ፈጣሪ ሰላማችንን እንድመልስልን በየሃይማኖታችን ጸሎት እናድርግ ሰላም ፍቅር ያስፈልገናል

Check

Afaan Oromo and Amharic Languages Hate Speech Detector

Dubbii Jibbiinsaa Miti/ የጥላቻ ንግግር አይደለም

Users Post:

Mootummaafi ummanni naannoo Oromiyaa ejjennoo cimaa, "Itoophiyaa Baraaruuf Nan Duula; Eessattuu, Yoomiyyuu, Waan Hundaan!" jedhuun hojii qabatamaa keessatti argamu.

Check

APPENDIX F: SAMPLE CODE FOR PREPROCESSING

```

1. h1=['ᵀ', 'ᵁ', 'ᶆ', 'ᵁ', 'ᶆ', 'ᵁ', 'ᵀ']
2. h2=['ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ']
3. h3=['ᵀ', 'ᵁ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ']
4. a1=['ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ']
5. a2=['ᵀ', 'ᵁ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ']
6. s1=['ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ']
7. s2=['ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ', 'ᵁ', 'ᵀ']
8. t1=['ᵀ', 'ᵁ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ']
9. t2=['ᵀ', 'ᵁ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ', 'ᶆ']
10.
11. l1=['ᵀ']
12. l2=['ᵀᵁ']
13. l3=['ᵀᵁ']
14. tu1=['ᵀ']
15. tu2=['ᵀᵁ']
16. tu3=['ᵀᵁ']
17. m1=['ᵀ']
18. m2=['ᵀᵁ']
19. m3=['ᵀᵁ']
20. r1=['ᵀ']
21. r2=['ᵀᵁ']
22. r3=['ᵀᵁ']
23.
24. for i in range(len(h1)):
25.     x=x.replace(h3[i],h1[i])
26.     x=x.replace(h2[i],h1[i])
27.     x=x.replace(t2[i],t1[i])
28.     x=x.replace(s2[i],s1[i])
29.     x=x.replace(a2[i],a1[i])
30.
31.     x=x.replace(l2[i],l1[i])
32.     x=x.replace(l3[i],l1[i])
33.     x=x.replace(tu2[i],tu1[i])
34.     x=x.replace(tu3[i],tu1[i])
35.     x=x.replace(m2[i],m1[i])
36.     x=x.replace(m3[i],m1[i])
37.     x=x.replace(r2[i],r1[i])
38.     x=x.replace(r3[i],r1[i])
39.
40.     x = x.replace('.', ' <PERIOD> ')
41.     x = x.replace(',', ' <COMMA> ')
42.     x = x.replace('"', ' <QUOTATION_MARK> ')
43.     x = x.replace(';', ' <SEMICOLON> ')
44.     x = x.replace('!', ' <EXCLAMATION_MARK> ')
45.     x = x.replace('?', ' <QUESTION_MARK> ')
46.     x = x.replace('(', ' <LEFT_PAREN> ')
47.     x = x.replace(')', ' <RIGHT_PAREN> ')
48.     x = x.replace('--', ' <HYPHENS> ')
49.     x = x.replace('?', ' <QUESTION_MARK> ')
50.     x = x.replace(':', ' <PERIOD> ')
51.     x = x.replace(':', ' <PERIOD> ')
52.     # text = text.replace('\n', ' <NEW_LINE> ')
53.     x = x.replace(':', ' ')
54.     x = x.replace(':', ' <SEMICOLON> ')
55.     x = x.replace(':', ' <HYPHENS> ')
56.     x = x.replace('[', ' <SQUARE_LEFT_PAREN> ')
57.     x = x.replace(']', ' <SQUARE_RIGHT_PAREN> ')
58.
59.     return x
60. df['posts'] = df['posts'].apply(lambda x: clean_text(x))
61.

```

APPENDIX G: AMHARIC LANGUAGE WORD SIMILIARITY EXAMPLE IN THE SKIP-GRAM MODEL

```
# Print word similiarity  
print(model['ደቃላ'])
```

```
[ 9.00633109e-04 -1.20685168e-03 -2.14350308e-04  6.99211669e-04  
-1.50629564e-03  7.01396959e-04  8.71173324e-05 -1.23240182e-03  
 3.59278230e-04 -1.13206590e-03 -1.58821791e-03  8.92636308e-04  
 1.11859394e-04 -7.76274654e-04 -6.53568830e-04 -1.06429914e-03  
 1.19110628e-03 -3.91720794e-04  7.44176737e-04 -9.90561792e-04  
-3.25107016e-04  1.01939491e-04 -5.88552153e-04 -8.92309530e-04  
-2.09346428e-04  9.43119521e-04 -2.17071516e-04 -1.10940670e-03  
-1.41153438e-03  1.54491409e-03 -6.39836770e-04  8.31987127e-04  
 4.45218902e-04 -1.25477603e-03 -1.57096877e-03 -8.60122498e-04  
-1.20575144e-03  6.48803602e-04  7.38905859e-04  1.27815432e-03  
-1.28849235e-03  9.46188884e-05 -8.72653211e-04 -4.28703439e-04  
 4.92547173e-04 -6.15808647e-04  5.02140902e-04 -1.29500136e-03  
 3.45771201e-04  9.08639922e-04 -7.87019206e-04 -5.29677782e-04  
-4.63606528e-04 -1.94282897e-04 -1.22985430e-03  1.15901011e-03  
 4.32737637e-04 -1.21440948e-03  6.55834214e-04 -5.47145319e-04  
-1.53538393e-04  1.37280824e-03 -2.52507598e-04 -7.86929682e-04  
-1.16135448e-03 -1.01561414e-03  1.48001791e-03  2.38911016e-04  
 2.00139519e-04 -8.97965510e-04 -3.46884452e-04  1.66371209e-03  
 1.33637059e-03 -1.06390617e-04 -6.76170224e-04 -6.11707801e-04  
 6.84414917e-05  5.96258207e-04 -4.66898258e-04 -1.23678718e-03  
 3.57251614e-04 -1.36626069e-03 -2.27456403e-05 -1.19959016e-03  
 7.48237653e-05  1.25699327e-03 -6.87123793e-06  4.95835964e-04  
 6.24445907e-04  1.10870902e-03 -6.34738361e-04  1.45973288e-03  
-3.36139638e-04 -6.61085302e-04  1.01266988e-03  4.93890293e-05  
 2.52624275e-04  2.48228986e-04  8.53655918e-04 -6.73130096e-04  
 1.41940138e-03  1.08179147e-03 -1.86073725e-04 -1.20404596e-03  
 9.87492152e-04 -6.60160367e-05 -1.38607793e-04 -9.20404855e-04  
-1.87336933e-04  1.39647524e-03 -6.35598321e-04 -7.88402758e-05  
-3.96891410e-04 -5.16837579e-04 -6.91899157e-04 -7.64912518e-04  
-3.83234030e-04 -1.17681222e-03  2.96371640e-04 -5.57849533e-04  
 1.25871401e-03 -1.44539063e-03  8.04983662e-04  1.54587440e-03]
```

APPENDIX H: AFAAN OROMO LANGUAGE WORD SIMILIARITY EXAMPLE IN THE SKIP-GRAM MODEL

```
# Print word similiarity
print(model['tokkummaa'])
```

```
[ 2.64306590e-02 -3.47278446e-01 -1.09621964e-01 -1.76423058e-01
-9.18513313e-02  8.59758481e-02  1.74550131e-01 -7.88534582e-02
 2.27978975e-01 -1.71712697e-01  5.60939424e-02 -1.42438635e-01
 1.58370603e-02  1.18905470e-01 -2.12481007e-01  6.53469443e-01
-4.72968459e-01  9.48255882e-02  4.40082818e-01 -3.05710942e-01
-3.93923633e-02  1.73297413e-02 -1.57127962e-01 -2.45028242e-01
 2.25409821e-01  1.66842878e-01 -2.04959556e-01 -6.33525670e-01
 1.34741878e-02  1.92124650e-01 -8.68488401e-02  5.37986636e-01
 6.40078634e-02 -2.03394145e-01 -1.63328081e-01 -4.53233272e-01
-5.31449676e-01 -1.38226837e-01  2.35845923e-01 -3.05935174e-01
-2.24829540e-02 -3.75580788e-01 -3.74983877e-01  4.46199656e-01
 3.18062723e-01  9.06747207e-02 -4.09028143e-01 -3.44939232e-01
-2.52354503e-01  2.45971814e-01  1.64180323e-01 -1.37874633e-01
 3.18169326e-01  1.58342812e-02 -1.18553147e-01  2.04154924e-01
 9.37148482e-02  9.07609761e-02  4.51529950e-01 -1.37589658e-02
-2.58832633e-01 -6.92160577e-02  1.11999877e-01  2.39077166e-01
-1.50334716e-01  3.64084184e-01  1.06073916e-01 -4.16428447e-01
 2.29963914e-01 -9.63083655e-02  1.98398724e-01 -5.59512060e-03
-2.53954113e-01 -1.32950380e-01 -3.82595435e-02 -5.97590864e-01
-3.41619998e-01  9.40075070e-02 -3.39168049e-02  3.72476637e-01
-2.97539115e-01  8.31718594e-02 -1.05931535e-01 -5.32525718e-01
-4.10213023e-02  2.96134621e-01  2.36837178e-01  2.44795203e-01
 1.50094146e-03 -6.32453039e-02  3.65390000e-03 -3.26587945e-01
 1.37869194e-01  4.76450734e-02  1.39202386e-01 -8.17149580e-02
-3.06301296e-01  7.89650679e-02 -2.55038530e-01  3.78647144e-03
-1.28691345e-01 -6.44994602e-02 -6.61741912e-01  1.23637371e-01
 2.03203615e-02  1.55935079e-01 -3.42846900e-01  6.60257190e-02
-6.16673753e-02  2.21848026e-01  1.61938161e-01 -3.47527027e-01
-7.82035664e-03  1.67397544e-01  5.31292893e-02 -3.17388475e-01
 6.90946504e-02  4.21612620e-01  1.77965689e-04 -1.95791245e-01
 1.04482837e-01 -4.11800183e-02  4.40644264e-01 -5.08491807e-02
 7.04079688e-01 -1.57837216e-02 -2.30647400e-01  8.61688331e-02
 1.51654720e-01  1.19320780e-01  4.18401718e-01 -4.09354717e-01
 2.11479336e-01  1.32294223e-01 -1.06352739e-01 -2.37461776e-01
 1.41606359e-02  4.26004648e-01  4.72982712e-02 -6.00568891e-01
-9.49680284e-02  4.16677177e-01  2.33441040e-01 -4.05517630e-02
-5.04219115e-01  4.53805208e-01 -2.62461692e-01 -1.63002983e-01
 1.60981119e-01  6.24103725e-01 -4.36733276e-01 -9.43293869e-02]
```