PARTS OF SPEECH TAGGER

FOR SIDAMA LANGUAGE USING THE HIDDEN MARKOV MODEL

WITH VITERBI ALGORITHM

M.Sc. THESIS

BELACHEW KEBEDE ESHETU

HAWASSA UNIVERSITY, HAWASSA, ETHIOPIA

DECEMBER, 2022

PARTS OF SPEECH TAGGER

FOR SIDAMA LANGUAGE USING THE HIDDEN MARKOV MODEL

WITH VITERBI ALGORITHM

BELACHEW KEBEDE ESHETU

ADVISOR: TESFAYE BAYOU (Ph.D.)

CO-ADVISOR: ANDARGACHEW MEKONNEN

THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE,

INSTITUTE OF TECHNOLOGY, SCHOOL OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN COMPUTER SCIENCES

(SPECIALIZATION: COMPUTER SCIENCE)

HAWASSA UNIVERSITY, HAWASSA, ETHIOPIA

DECEMBER, 2022

SCHOOL OF GRADUATE STUDIES

HAWASSA UNIVERSITY EXAMINERS' APPROVAL SHEET

We, the undersigned, members of the Board of Examiners of the final open defense by **Belachew Kebede** have read and evaluated his thesis entitled "*Parts Of Speech tagger for Sidama Language using the Hidden Markov Model with Viterbi Algorithm*", and examined the candidate. This is, therefore, to certify that the thesis has been accepted in partial fulfillment of the requirements for the degree.

_____	_____	_____
Name of Major Advisor	Signature	Date
_____	_____	_____
Name of Co-Advisor	Signature	Date
_____	_____	_____
Name of Internal Examiner-I	Signature	Date
_____	_____	_____
Name of Internal Examiner-II	Signature	Date
_____	_____	_____
Name of External examiner	Signature	Date
_____	_____	_____
SGS Approval	Signature	Date

Final approval and acceptance of the thesis is contingent upon the submission of the final copy of the thesis to the School of Graduate Studies (SGS) through the Department/School Graduate Committee (DGC/SGC) of the candidate's department.

Stamp of SGS Date: _____

DEDICATION

This thesis is dedicated to my mother W/ro Gebeyanesh Berhanu Admasu.

DECLARATION

I hereby declare that this M.Sc. thesis is my original work and has not been presented for a degree in any other university, and all sources of material used for this thesis have been fully acknowledged.

Name: Belachew Kebede

Signature: ----------

This MSc thesis has been submitted for examination with our approval as Thesis advisors.

Place: Hawassa University, Ethiopia

Date: ----------

ACKNOWLEDGEMENTS

First and above all, I would like to thank the ALMIGHTY GOD and his mother Virgin Marry who have made it possible for me to undertake this research work and blessed me in every step of my life.

My deepest and heartfelt goes to my thesis advisor, Dr. Tesfaye Bayou, and co-advisor, Mr. Andargachew Mekonnen, for their guidance and helpful comments on my work. I would like to thank Mr. Mathewos W/Giorgis from Hawassa University's Department of Sidama Language and his team members who helped me in manually tagging the Sidaama corpus. I would like also to express my deepest gratitude to Dr. Deribe Workineh from Hawassa University's Department of Psychology, who gave me constructive comments and inspiring suggestions that helped me very much to complete this research work.

I also thank the Office of the Vice President for Research and Technology Transfer, Hawassa University, for funding my thesis as part of the thematic research Project: Integrating ICT for Sustainable Development. I equally thank the SNNPRS Bureau of Finance for all the help with my learning time.

Finally, I would like to extend my acknowledgments to my wife S/r Aster H/Mariam, my sons Darik Belachew and Yosef Belachew who have rendered me all their care, love, and encouragement, friends Denekew Alemayehu, Yirgaye Jeda, Getachew Mekonnen, Misrak Hadis, Suzi W/Mariam and everyone that in one way or another helped me in the completion of this thesis.

TABLE OF CONTENTS

Content **Page**

ACKNOWLEDGEMENTS vi

ACRONYMS AND ABBREVIATIONS x

LIST OF TABLES xi

LIST OF FIGURES xii

ABSTRACT xiii

CHAPTER ONE 1

INTRODUCTION 1

1.1 Background 1

1.2 Statement of the Problem 2

1.3 Research Questions 3

1.4 The objective of the Study 3

1.4.1 General Objective 3

1.4.2 Specific Objectives 3

1.5 Significance of the Study 3

1.6 Scope and limitations of the Study 4

1.7 Organization of the Thesis 4

2. CHAPTER TWO 5

REVIEW OF LITERATURE AND RELATED WORKS 5

2.1. Literature Review 5

2.1.1. Approaches to POS Tagging 6

2.1.2. Rule-based Approach 8

2.1.3. Stochastic Approach 9

2.1.4. Hybrid Approach 10
2.2. Related Work 10
3. CHAPTER THREE 14
SIDAMA LANGUAGE 14
3.1. Introduction 14
3.2. Word Classification of Sidaama 17
3.2.1. Noun (Su'ma) 17
3.2.2. Pronoun (Su'mi-riqiwo) 18
3.2.3. Adjective (Su'mi-ledo) 18
3.2.4. Verb (Gurda) 18
3.2.5. Adverb (Gurdi-ledo) 19
3.2.6. Conjunction (Amandisiisancho) 19
3.2.7. Preposition (Sissa) 20
3.2.8. Interjection (Qaalu darba) 20
4. CHAPTER FOUR 21
MATERIALS AND METHODS 21
4.1. Materials 21
4.2. Methods 21
4.2.1. The Preparation of Sidaama corpus 22
4.2.2. Identification of Sidaama Tag set 25
4.2.3. The n-gram 28
4.2.4. Hidden Markov Model (HMM) 28
4.2.5. Maximum likelihood estimation probabilities (MLE) 35
4.2.6. The Viterbi Algorithm 36

4.2.7. Unknown word handling in HMM 38
4.2.8. Evaluation 39
4.2.9. Cross-validation 43
4.2.10. The Hidden Markov Model for Sidaama 44
4.2.11. The Sidaama HMM POS Tagger 48
4.2.12. The Prototype Implementation 51
5. CHAPTER FIVE 54
RESULTS AND DISCUSSION 54
5.1. The test result of the HMM tagger 54
5.2. Performance Analysis of HMM Tagger 56
6. CHAPTER SIX 61
CONCLUSION AND RECOMMENDATION 61
6.1. Conclusion 61
6.2. Recommendation 63
6. REFERENCES 64
7. APPENDICES 70
7.1. Appendix A: Untagged Sample of the Corpus 70
7.2. Appendix B: Tagged Sample of the corpus 71

ACRONYMS AND ABBREVIATIONS

ANN	Artificial Neural Network
HMM	Hidden Markov Model
IE	Information Extraction
IR	Information Retrieval
MLE	Maximum Likelihood Estimation
NLP	Natural Language Processing
NLTK	Natural Language Processing Tool Kit
OOV	Out of Vocabulary
POS	Parts of Speech
POST	Parts of Speech Tagging
SOV	Subject Object Verb
SPOST	Sidaama Parts of Speech Tagging
TBL	Transformation-Based Learning
VA	Viterbi Algorithm
WSD	Word Sense Disambiguation

LIST OF TABLES

Content **Page**

Table 3.1 The Sidaama Alphabets /Fidalla/ 15

Table 3.2 Phonemic representation of the Sidaama Fidalla 16

Table 3.3 Short and Long Vowels 16

Table 4.1 NLTK Taggers Accuracies of different corpora 244

Table 4.2 The Sidaama tag set 266

Table 4.3 Confusion Matrix for Binary Classification 40

Table 4.4 Multiclass classification problem confusion matrix 43

Table 4.5 Sample Sentences with the corresponding tags 44

Table 4.6 Sample Transition Probabilities 45

Table 4.7 Sample Emission Probabilities 45

Table 5.1 The Results of the Sidaama tagger performance 54

Table 5.2 Ten-Fold Cross-Validation Result 55

Table 5.3 The POS tags Frequency 56

Table 5.4 Confusion matrix of HMM tagger 58

Table 5.5 Precision, Recall, F1-Score and Accuracy 60

LIST OF FIGURES

Content **Page**

Figure 2.1 Classification of POS tagging approaches [15]..... 7

Figure 2.2 Broad Classification of tagging..... 8

Figure 4,1 The corpus preparation process..... 244

Figure 4.2 NLTK Taggers Accuracies of different corpora 255

Figure 4.3 The Architecture of HMM [16]..... 34

Figure 4.4. HMM for Sidaama as Probabilistic Finite State Automata**……...…….……**46

Figure 4.5 The block diagram of the Sidaama Tagger [10]..... 48

Figure 4.6 Prototype of Sidaama POS Tagger..... 51

Figure 4.7 Sample Input and result of the tagger..... 52

Figure 5.1 Performance curve analysis for HMM tagger 55

Figure 5.2 The distribution of tags in Sidaama Corpus 57

ABSTRACT

The Parts of Speech (POS) tagger is an essential low-level tool in many natural language processing (NLP) applications. POS tagging is the process of assigning a corresponding part of a speech tag to a word that describes how it is used in a sentence. There are different approaches to POS tagging. The most common approaches are rule-based, stochastic, and hybrid POS tagging. In this paper, the stochastic approach, particularly the Hidden Markov Model (HMM) approach with the Viterbi algorithm, was applied to develop the part of the speech tagger for Sidaama. The HMM POS tagger tags the words based on the most probable sequence of words. For training and testing the model, 9,660 Sidaama sentences containing 130,847 tokens (words, punctuation, and symbols) were collected, and 4 experts in the language undertook the POS annotation. Thirty-one (31) POS tags were used in the annotation. The source of the corpus is fables, news, reading passages, and some scripts from the Bible. 90% of the corpus is used for training and the remaining 10% is used for testing. The POS tagger was implemented using the Python programming language (python 3.7.0) and the Natural Language Toolkit (NLTK 3.0.0). The performance of the Sidaama POS tagger was tested and validated using a ten-fold cross-validation technique. In the performance analysis experiment, the model achieved an accuracy of 91.25% for HMM model and 98.46% with the Viterbi algorithm.

Keywords: *Hidden Markov model, Natural Language Processing, Part of Speech Tagger, Rule-based Tagger, Stochastic Tagging, Hybrid POS tagger, Viterbi algorithm*

CHAPTER ONE

INTRODUCTION

1.1 Background

Natural language processing (NLP) is a subfield of computer science with strong connections to artificial intelligence (AI). The term natural language processing (NLP) is typically used to describe how computer systems analyze or synthesize spoken or written language. It helps computers understand, interpret, and imitate human language.

One area of NLP is concerned with developing a POS tagger. POS tagging is the act of assigning each word in a sentence a tag that describes how that word is used in the sentence. A POS tagger is a program that reads the text in the given language and assigns tags to each word and other tokens within the text. It can also be considered as a piece of software that reads the text in some language and assigns parts of speech tags to each word in a sentence.

The POS tagger is an important part of many natural language processing applications, such as word sense disambiguation (WSD), parsing, question answering (QA), information retrieval (IR), information extraction (IE), and machine translation (MT).

There are different classifications of POS taggers. POS taggers can be classified into supervised and un-supervised. The pre-tagged corpus is used to train supervised POS tagging models to learn the tag set, word-tag frequency, rule settings, etc.

The performance of this model improves as the corpus grows larger. A pre-tagged corpus is not necessary for the unsupervised models[1].

The POS tagging approaches can be broadly classified as rule-based, stochastic, or hybrid. The rule-based taggers use hand-coded rules to determine the lexical categories of a word

[15]. A statistical approach includes the most frequent tag, the n-gram, and the hidden Markov model (HMM) [10]. The hybrid approaches make use of both manually written rules that have been pre-defined and generated during training [2]. The hybrid approach combines a rule-based method with a probabilistic method to automatically get symbolic rules from a corpus.

This thesis is to apply the supervised HMM of the statistical approach and develop a part-of-speech (POS) tagger for the Sidama language (Sidaama).

Sidaama is the language of the Sidama people who live in the current Sidama regional state, and it is used as the working language of the regional state. It is a Cushitic language closely related to Hadiya, Kembata, Gedeo, and Guji, which together form, the branch known as Highland East Cushitic. It is also sometimes called Sidaamu Afoo. The ethnic name for the language is Sidaama, and 3.7 million people are thought to speak it [22].

1.2 Statement of the Problem

Several POS taggers are available on the web for different languages. There are also POS taggers developed for Ethiopian languages such as Amharic, Afaan Oromo, Tigrigna, Hadiya, and Kafinoono [3][2][4][5][6]. There is also a POS tagger for Sidama language developed by Addisu Bole using Brill's Transformational Error Driven learning approach. On the Other hand a study reported on research gate (abstract only) by Abraham Ayana[7], who has worked on the title "Improving Part of Speech Tagger Performance for the Sidama. Both researchers used the Brill tagger with different data size resulting in the same performance. The goal of this research is to develop a POS tagger for the Sidama language using another method, HMM and the Viterbi algorithm and analyze the performance of the tagger.

1.3 Research Questions

The research will address the following research question:

- How much performance is enhanced when the Viterbi algorithm is applied to an HMM POS tagger for Sidaama?

1.4 The objective of the Study

1.4.1 General Objective

The main goal of this research is to develop a POS tagger for the Sidama language by using HMM and with the Viterbi algorithm and analyze the performance of the tagger.

1.4.2 Specific Objectives

The specific objectives of this research are to:

- identify the Sidama word categories and tag sets,
- develop a Sidama language corpus,
- develop the Sidaama HMM model,
- develop an HMM POS tagger prototype,
- test the performance of the tagger

1.5 Significance of the Study

Developing the Sidaama POS tagger has much significance. Such as

- Researchers who want to use higher-level NLP applications for this language, like parsing, spell checking, grammar checking, speech recognition, question answering, etc., can use it as a preprocessing step.
- It helps people who want to learn Sidaama as a second language by discovering the word categories and grammar construction.

1.6 Scope and limitations of the Study

As far as the knowledge of the researcher concerned, there is no unprocessed or ready-made corpus available for the Sidama language. Thus, the researcher should build the corpus on his own. So, the corpus prepared for this thesis was taken from different areas such as Sidaama Fables, News, and Sidama language students' textbooks, and selected passages from the Bible. The size of the corpus is 9,660 sentences, containing about 130,883 tokens (words and symbols), which is relatively small for using the HMM model. In the HMM model, as the size of the corpus increases, the performance also increases. That means, in this model, the larger the corpus, the better the performance. On the other hand, the tag set doesn't include gender classes, and the number of gender-related classifications needs to be looked at more closely. Since there aren't many Sidama language experts, manual annotation is expensive. Also, not much NLP research has been done on the Sidama language, so it has been hard to use previous works as a guide.

1.7 Organization of the Thesis

The following six chapters comprise this thesis. The first chapter focuses on the introductory part of the thesis. The second chapter presents a review of the literature on concepts related to the thesis and related works to our work done by other researchers in different languages. The third chapter discusses the Sidama language's nature study, word class, sentence structure, and tag set preparation. Chapter four presents the materials and methods used in this work. Chapter five describes the development of the Sidaama POS tagger. Finally, the last chapter focuses on the conclusion and recommendations of the research.

CHAPTER TWO

REVIEW OF LITERATURE AND RELATED WORKS

2.1. Literature Review

Natural language processing (NLP) is a subfield of computer science with strong connections to artificial intelligence (AI). The term natural language processing (NLP) is typically used to describe how computer systems analyze or synthesize spoken or written language[3].

One of the most important activities in natural language processing (NLP) is speech tagging [2][8]. Part-of-speech (POS) tagging is the process of assigning each word in sentences a POS tag that describes how that word is used in the sentences [2]. A part-of-speech tag is a grammatical category of words in a sentence. In the English language, there are eight common parts of speech, namely: verbs, nouns, adjectives, adverbs, pronouns, prepositions, conjunctions, and interjections, and their sub-categories [2].

A part of speech (POS) tagger is a program that reads the text in the given language and assigns tags to each word and other tokens within the text.

In linguistic research for corpora, part-of-speech (POS) taggers play an important role in natural language applications such as text-to-speech (TTS) synthesis, information retrieval (IR), information extraction (IE), automatic written text recognition, grammar checking, and word sense disambiguation (WSD) [9][10][11].

The two main problems in designing an accurate automatic POS tagger are word ambiguity and unknown words [12]. Word ambiguity refers to the different behavior of words in different contexts. As in other languages, many words in the Sidama language can have more than one tag.

For example,

Aana beettu barcimu no. /aana (Adverb)

Aana dikko hadhu. /aana (proper noun)

Mayra woratto qorete aana? /aana (Verb)

To solve this problem, we consider the context instead of a single word. An out-of-vocabulary (OOV) word (also called an unknown word) is a word that is unavailable in the annotated corpus [12]. The percentage of these OOV words that appear in the test set is known as the OOV rate [13].

2.1.1. Approaches to POS Tagging

Several approaches to implementing POS taggers have been proposed in various studies. As noted in [9][14]. POST can be classified into supervised and unsupervised POS tagging. The supervised POS is a rule-based, stochastic, and neural network (NN), while the unsupervised POS is a rule-based, transformation-based, and neural network (NN) [15].

A pre-tagged corpus is utilized for training the supervised POS tagging models to learn the tag set, word-tag frequencies, rule settings, etc. The performance of this model improves as the corpus size grows larger. For the unsupervised POS tagging methods, a pre-tagged corpus is not necessary. Instead, they use advanced computational methods like the Baum-Welch algorithm to automatically induce tag sets [12].

The rule-based POS tagging approach assigns POS tags to words by applying a set of handwritten rules and making use of contextual information.

The stochastic approach, which is also called the statistical approach, finds the word that shows up most often in the annotated training data and uses that information to tag the word in the un-annotated text [2].

Figure 2.1 shows the classification of the parts of speech tagging approach [15].

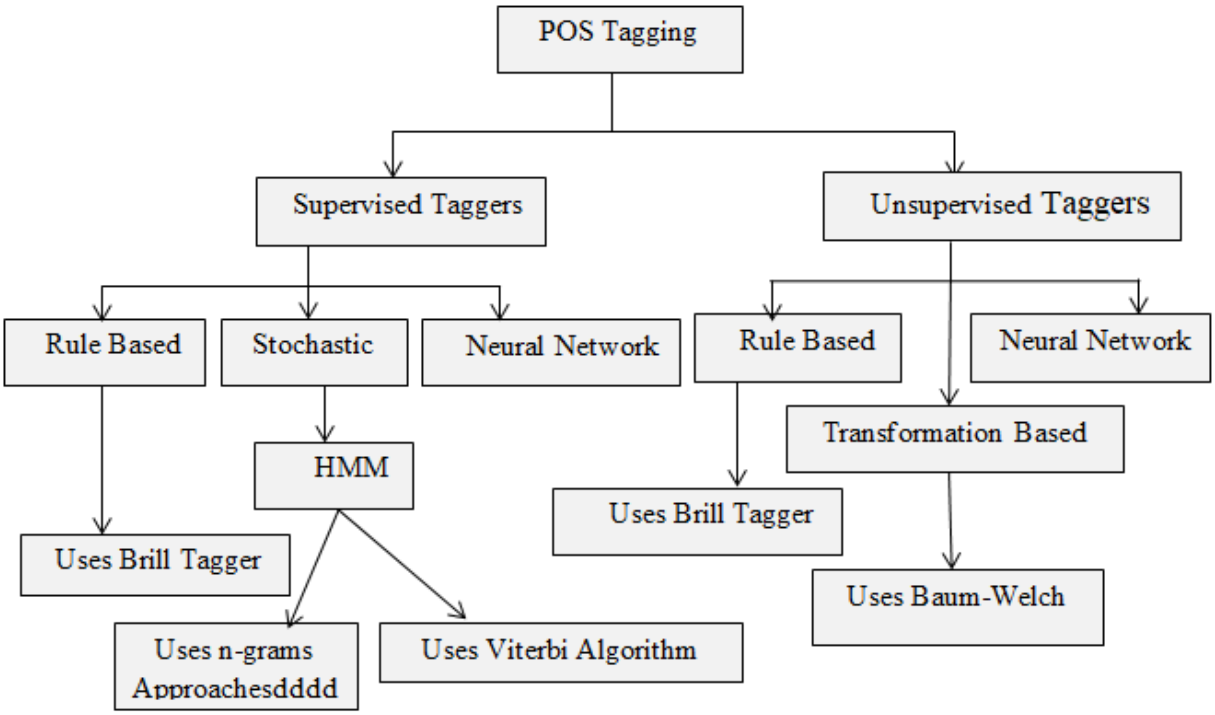


Figure 2.1 Classification of POS tagging approaches [15]

Artificial Neural Network (ANN) models were inspired by biological neural networks that use the many desirable characteristics of the human brain, which embrace distributed illustration and computation, massive parallelism, and the ability to learn and make generalizations. It solves problems that humans or applied mathematics might find impossible or hard to solve. Nowadays, neural networks are used for several business issues, like sales prognostication, client analysis, information validation, and risk management.

Transformation-Based-Learning (TBL) transforms one state into totally different exploitation transformation rules, so you have to go looking for the suitable tag for each word. It extracts linguistic information from corpora. Transformation-based learning (TBL) can be a rule-based algorithmic rule for the automatic tagging of parts of speech to the given text.

These approaches can broadly be classified as rule-based, statistical, and hybrid approaches [16][17][18][19].

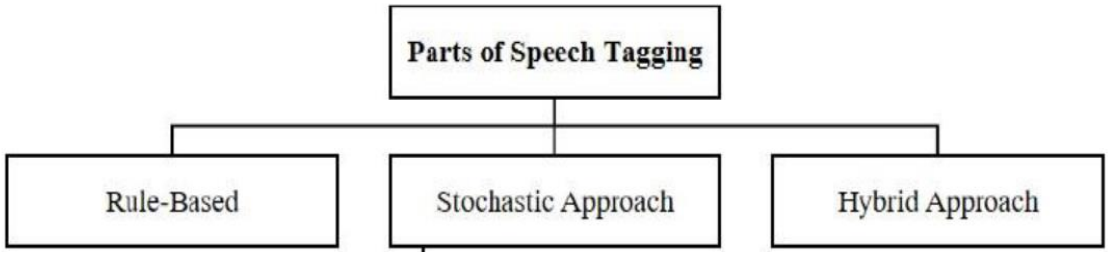


Figure 2.2 Broad Classification of tagging

2.1.2.Rule-based Approach

The rule-based approach uses a set of rules to assign tags to words and sentences. The rule-based POS tagging approach assigns POS tags to words by applying a set of handwritten rules and making use of contextual information. In other words, a rule-based POS tagger applies a POS tag to a word in accordance with a number of linguistic rules that were manually constructed[2]. The main drawbacks of the rule-based approach are the necessity of linguistic background and manually constructing the rules [2]. The disadvantage of this approach is that it doesn't work when the text is not known [14]. A rule-based approach should be used with a full set of hand-coded rules to make this system more efficient and accurate [14].

The part-of-speech tagging system, which used a rule-based approach, is considered the oldest system.

The advantage of the rule-based model

- It needs less stored information (training data).
- It is more portable from one tag set or corpus genre to another.

The disadvantages of the rule-based model

- The development could be very time-consuming.
- It requires a linguistic background[20]
- It is less accurate as compared to stochastic taggers.

2.1.3. Stochastic Approach

The stochastic approach, which is also called the statistical approach, finds the word that shows up most often in the annotated training data and uses that information to tag the word in the un-annotated text [2].

The statistical approach finds the most likely tag for a token based on probability values taken from a manually tagged corpus[19].

The most up-to-date part-of-speech taggers are probabilistic (also called "stochastic"), which prefer to tag a word by figuring out the most likely tag based on the word and the words around it.

Some of the models used in stochastic POS tagging are Hidden Markov Models (HMM), Maximum Likelihood Estimation, N-grams, Support Vector Machines (SVM), Conditional Random Fields (CRF), and Maximum Entropy (ME). This method of POS tagging makes use of statistics, frequency, and probability.

The advantages of the stochastic approach

- It may not need linguistic expert
- It is more accurate as compared to rule-based taggers.

The Disadvantages of the Stochastic Approach

- There are situations when a tag sequence appears that does not follow the grammatical rules of the language [20].
- It requires a vast amount of stored information.

2.1.4. Hybrid Approach

The hybrid approaches employ both manually constructed rules that have been pre-defined and rules that are automatically induced during training [2]. It uses both rule-based and probabilistic methods to automatically get symbolic rules from a corpus.

It may even perform better than statistical or rule-based approaches. In the hybrid approach, statistical methods are used for probabilistic features, and then a set of rules that are specific to each language are applied [14].

The advantage of the hybrid approach

It has higher accuracy than either individual rule-based or statistical approaches.

The disadvantage of the hybrid approach

It does not assign the correct tag to an unknown word.

2.2. Related Work

Previous research work for some foreign and local languages that is related and relevant to this work is discussed in this section.

Archit Yajnik [29] has developed a part-of-speech tagger for Nepali text using a statistical approach. In this research, the annotated corpus training and testing data sets are separated randomly. The researcher employed the Hidden Markov Model and the Viterbi algorithm on the data set and found that the Viterbi algorithm is computationally faster and more accurate as compared to the HMM. An accuracy of 95.43% is achieved using the Viterbi algorithm.

Jyoti Singh et al. [16] have developed part of speech tagger for Marathi text using the trigram method of a statistical approach. They made a test corpus of 2000 sentences (48,635 words) to see how accurate it was, and it was 91.63% accurate.

Kh Raju Singha et al. [14] have developed a POS tagger for the Manipuri language using HMM. In this work, the researchers used the manually annotated test set data with the tag set of 97 morpho-syntactic categories of Manipuri as an evaluation set and got an accuracy of 92%.

Getachew Mamo and Million Meshesha [30] have developed the Afaan Oromo part of the speech tagger using the HMM approach. In this work, the researcher used the HMM approach, which is one of the most common mechanisms under the stochastic approach. The researcher collected and used 159 sentences with a total of 1621 words for both training and testing to make the corpus and used a tag set of 17 tags.

In the tagging process, the tagger assigns word classes to a given Afaan Oromo text in two main phases. In the first phase, the tagger trains on the training data to compute and store both the emission and the transition probability of the training data. In the second phase, the tagger accepts untagged Afaan Oromo text and tokenizes it into words. Then the tagger assigns the correct POS tag to each token.

This is achieved using the unigram and bigram models of the Viterbi algorithm by taking the stored information during the first phase. The author tested the performance of the tagger using a ten-fold cross-validation mechanism. As a result, the accuracy of the unigram model was 87.58% and the accuracy of the bigram model was 91.97%.

Teklay G/Egzabiher [4] has developed a part-of-speech tagger for the Tigrigna language using a hybrid (a combination of rule-based and statistical approaches) model. The researcher used a combination of HMM, which is widely used in the stochastic approach, and the Brill transformation-error-driven learning approach to drive machine-learned rules for designing the rube-based tagger component in this work. The researcher has collected a total of 26,000 words from Tigrigna news broadcasting agencies and annotated them manually with their corresponding word class. In addition to this, he has identified 36 POS tags for the entire tagging process. Among the total words, 75% (20,000) words were used for training purposes, while the remaining 25% (6000) words were used for testing purposes.

The researcher ran tests on the HMM tagger, the rule-based tagger, and the hybrid tagger, and found that they performed at 89.13%, 91.8%, and 95.88%, respectively.

Zelalem Mekuria [30] has developed a part-of-speech tagger for Kafinoonoo/Kaffa Language using HMM, rule-based, and hybrid approaches. In this work, the researcher has collected 354 untagged Kafinoonoo sentences from two genres and annotated them using an incremental corpus preparation approach. Among the total words, 90% of the words were used for training purposes, while the remaining 10% of words were used for testing purposes and used 34 POS tags.

The performances of HMM, rule-based, and hybrid taggers were tested, and obtained accuracies of 77.19%, 61.88%, and 80.47%, respectively.

Addisu Bole(2016) [31] developed parts of a speech tagger for the Sidama language using Brill's Transformational Error Driven learning approach. In his work, he has collected 2,842 sentences, and identified 32 tags. In his study, he used 10-fold cross-validation method for the evaluation. After modification of the templates of the original Brill's tagger with an accuracy of 85.84%, the tagger achieved an accuracy of 93.64%.

Ayana, Abraham (2018)[7] worked on improving the performance of part of the speech tagger for the Sidama language. In his work, he applied an incremental approach to preparing a corpus of 10,274 sentences containing 63,400 words. After modifying the templates of the original tagger with an accuracy of 85.84 percent, the tagger obtained an accuracy of 93.64%.

Wubetu Barud (2019) [25] developed parts of a speech tagger for the Awngi language using the Hidden Markov Model (HMM). In his work, he has collected 94,000 sentences with a total word count of 188,760 for both training and testing sets, and identified 23 tags. His experimental results indicate an accuracy of 93.64% and 94.77% for unigram and bigram taggers, respectively.

CHAPTER THREE

SIDAMA LANGUAGE

3.1. Introduction

The Sidama people live in the Sidama Region. The regional capital is called Hawassa. It is located 175 kilometers south of Addis Ababa, the capital city of Ethiopia. The majority of Sidama people are farmers, producing root crops, like ensete, cereals like wheat, barley, peas, and beans, and cash crops, like coffee. They also rear animals. Fishing is also common in some districts of the region. Coffee is the main cash crop grown in the region, and that engages many traders.

Sidama region shares borders with Arusi Oromo in the north and north east, Guji Oromo in the south, Gedeo in the west, and Wolayta in the North West [32].

The Sidama language/Sidaama/is a Cushitic language closely related to Hadiya, Kembate, Gedeo, and Guji, which together form the Highland East Cushitic branch [33]. It is also sometimes called Sidamigna/Sidaminya (AMH), or Sidamic (in English, using the ic ending of "Amharic"). Sidaama is the ethnic name for the language, while Sidaminya/Sidamegna is its name in Amharic [34]. In this thesis, Sidaama is used to mean "the Sidama language".

The Sidama people are the third-largest Cushitic group with 4 million members in Ethiopia [35]. Sidaama is a morphologically rich language and is one of the major languages of Ethiopia, having an estimated 3.7 million speakers [33]. It is one of the resource-scarce languages.

Sidaama has adopted the Latin script. The Sidaama script is called Fidalla. It has 33 symbols. These are: 25 of them are single-digit letters found in Latin script as found in English, except

v, the 7 two-digit letters; and the symbol (‘). The two-digit letters are ch, dh, ny, ph, sh, ts, and zh [33][36][37].

According to [34], in Fidalla, like in English, both capital letters and small letters are used. The small letters are called Shimmaadda Fidalla and the capital letters are Jajjabaa Fidalla.

The total Sidaama alphabets [Fidalla] are shown in table 3.1.

Table 3.1 The Sidaama Alphabets /Fidalla/

Jajjabbaanna:	A B C CH D DH E F G H I J K L M N NY O P PH Q R S SH T TS U W X Y Z ZH
Shiimmaammaadda:	a b c ch d dh e f g h I j k l m n ny o p ph q r s sh t ts u w x y z zh

Another important thing in the Sidama language is the use of the apostrophe (') and double apostrophe (") when there is a break between vowels or vowels and consonants. According to [38], (‘) is considered one of the alphabets, and the number of alphabets in Sidaama became 33. The apostrophe (') and double apostrophe (") indicate that the vowels are pronounced independently from each other.

For example:

Ce’a /bird

Cee’ma /to be lazy/

Ka”a /She rised/

Ka”u /He rised./

The phonemic representation of the Fidalla symbols is shown table 3.2 in the International Phonetic Alphabet/IPA [33][36].

Table 3.2 Phonemic representation of the Sidaama Fidalla

A a	B b	C c	CH ch	D d	DH dh	E e	F f	G g	H h	I i	J j	K k	L l	M m	N	
[a]	[b]	[tʃ']	[tʃ]	[d]	[ɗ]	[e]	[f]	[g]	[h]	[i]	[dʒ]	[k]	[l]	[m]	[n]	
NY ny	O o	PH ph	Q q	R r	S s	SH sh	T t	TS ts	U u	W w	X x	Y y	Z z			
[n]	[o]	[p']	[k']	[r]	[s]	[ʃ]	[t]	[s']	[u]	[w]	[t']	[j]	[z]			

Sidaama has five short vowel phonemes and five long counterparts [33][37][39]. The vowels and the respective examples are shown in table 3.3.

Table 3.3 Short and Long Vowels

Short Vowel		Example	Long Vowel		Example
Jajjabba Fidalla.	Shimmaadda Fidalla		Jajjabba Fidalla	Shimmaadda Fidalla	
A	a	jawa ‘great’	AA	aa	Jaawa ‘to become thin’
E	e	Tenne ‘at that time’	EE	ee	teenne ‘files’
I	i	dina ‘to limp’	II	ii	diina ‘enemy’
O	o	Ko’la ‘to reply’	OO	oo	k’oola ‘wing’
U	u	Kula ‘to tell’	UU	uu	kuula ‘blackish blue’

In the Sidaama writing system, two vowels in a row show that the vowel is long. For example, gola (to hide) and goola (torch, bright fire) both have long vowels. Consonant germination, which is the doubling of a consonant, is phonemic in Sidaama.

At present, Sidaama is used as the working language of the Sidama Regional Administration and the medium of instruction in elementary schools. It is also taught in secondary and preparatory schools (Grades 7–12).There is also a BA (bachelor of arts) and BEd (bachelor of education)] in Sidama at Hawassa University.

Sidaama structure

Sidaama has the following possible syllable structures: V, VV, VC, VCC, CV, CVV, CVC, CVVC, and CVCC, where C is a consonant and V is a vowel.

The word order of the Sidama language is Subject–Object–Verb (SOV) [30][36][37]. OSV order is possible but is used only when the subject is in focus. For example:

Ise sagale ittu. (She ate food).

Lat'o Dangura batt'anno. /Lat'o loves Dangura/.

In Sidaama, subordinate clauses come before main clauses. Auxiliary verbs come after the main verb. Instead of prepositions, "postpositions" (suffixes and enclitics) are used.

3.2. Word Classification of Sidaama

In this section, we discussed some of the basic Sidaama word classes. These are: noun (Su'ma), pronoun (Su'mi-riqiwo), adjective (Su'mi-ledo), verb (Gurda), adverb (Gurdi-ledo), conjunction (Amandisiisancho), and interjection (Qaalu darba).

3.2.1. Noun (Su'ma)

A noun could be a person, place, concept, or object. Nouns are classified into two categories, namely: common nouns and proper nouns. Common nouns are general names for things, whereas proper nouns are specific names for individual things.

Sidaama nouns mark number, gender, and case grammatically. As in many languages, the number distinction in Sidaama is singular versus plural. Singular nouns are unmarked morphologically. But in most Cushitic languages and some Semitic languages, like Amharic, most singular nouns take a singular marker to refer to a specific noun in a given discourse situation.

The plural form is complex and involves more than ten morphological patterns. Gender is classified as masculine or feminine.

3.2.2. Pronoun (Su'mi-riqiwo)

Pronouns are words that are used in place of or in addition to nouns and noun phrases.

Most of the time, a personal pronoun is either the subject or the object of a sentence.

Personal pronouns can take the place of animate nouns and be used in the same contexts. Sidaama personal pronouns are ani/ane, ati/ate, isi/iso, ninke, kine, and insa.

3.2.3. Adjective (Su'mi-ledo)

An adjective could be a word that describes nouns and pronouns. It specifies which one, how much, what kind, and more. An adjective enables listeners and readers to visualize things more clearly by using their senses.

Adjectives in Sidaama can be simple or derived. The derived ones are mostly derived from verbs by adding the derivational morpheme /a/, /-o/, or /-ado/. According to Anbessa (2000),

Compared to the others, the derivational morpheme /-a/ is more frequent and productive. Both derived and simple adjectives agree with the head noun they modify in terms of number, gender, and case. As a result, adjectives can acquire noun-like inflection to acquire nominal grammatical categories.

3.2.4. Verb (Gurda)

Verbs are action words that describe what happens in a sentence. They can also convey the emotional condition of a sentence subject (is, was). Verbs change forms based on tense and count distinctions.

Sidaama verbs inflect various grammatical categories such as aspect/tense, mood, and agreement. They also take causative and passive morphemes to derive causative and passive verbs, respectively.

The distinction of tense in Sidaama is past versus non-past, where non-past refers either to the present or future depending on the context. These tenses are not morphologically marked by verbs. This does not mean, however, the tense is not indicated in Sidaama.

3.2.5. Adverb (Gurdi-ledo)

Adverbs are words that describe other adverbs, adjectives, and verbs. They detail the circumstances surrounding an event, including its timing, location, mode of action, cause, and frequency.

Adverbs in Sidaama are few. Due to this, adverbial functions are often expressed by ad positional phrases and subordinate clauses. There are, however, simple, derived, and compound adverbs. The latter outnumber the former two. Unlike the other word classes that we saw above, adverbs do not inflect for grammatical categories.

3.2.6. Conjunction (Amandisiisancho)

In a sentence, conjunctions are the words that link individual words, phrases, and clauses.

A coordinating conjunction could be a word that joins two parts of equal grammatical rank and syntactic importance.

Coordinating conjunctions in Sidaama are clitics, which are attached to the second to last constituent of conjoined constituents (or the first constituent, if two constituents are conjoined). Two of the conjunctives, = nna and = nso, begin with consonant clusters. When connecting multiple constituents with the same status, the conjunctive enclitic = nna is used.

A subordinating conjunction is just the word/word that's accustomed to being a part of a subordinator clause to a different clause or sentence. Subordinating conjunctions in Sidaama are words like kayinni ‘but’, ikkollana ‘however’, ikkirono ‘although/even if/’.

3.2.7. Preposition (Sissa)

A preposition is a word or cluster of words used before a noun, pronoun, or phrase to indicate a direction, time, place, location, or spatial relationship, or to introduce an object. It starts a prepositional phrase, which is made up of a preposition and the thing it refers to.

In Sidaama, we tend to use postpositions (suffixes and enclitics) rather than prepositions. In a sentence, a postposition is a word that indicates how a noun or pronoun relates to another word. It works like a preposition, but it comes after the object instead of before it.

3.2.8. Interjection (Qaalu darba)

An interjection is a word, phrase, or sentence that demonstrates an emotion or feeling. These emotional words often proceed with exclamation marks and can stand alone or be placed before or after a sentence.

In Sidaama, discourse particles such as ee ("yes"), deechi ("no"), ballo ("please"), and aassu ("pleasure") can be considered interjections.

CHAPTER FOUR

MATERIALS AND METHODS

4.1. Materials

For this research work, the Python programming language (Python 3.7.0) with the Natural Language Processing Tool Kit (NLTK 3.0.0) was used for implementation. NLTK is an open-source Python code library that is used to build NLP programs in Python. It is a suite of text-processing libraries for sentence and word tokenization, part-of-speech tagging, chunking, named entity recognition, text classification stemming, and parsing [40]. There are also several corpora, such as the Brown corpus and the Tree Bank corpus, which are available in the NLTK library to use for data analysis. For this research, three corpora: sidama (main corpus), sidamatrn (training set), and sidamatst (training set), are prepared and added to the corpora in NLTK.

4.2. Methods

This section includes the Sidaama Corpus preparation, tag set identification, the n-gram, Hidden Markov Model (HMM), maximum likelihood estimation probabilities (MLE), the Viterbi algorithm, unknown word handling in HMM, evaluation, cross-validation, the Hidden Markov Model for Sidaama, the Sidaama HMM POS Tagger, and the prototype implementation.

4.2.1. The Preparation of Sidaama corpus

A corpus, often known as a text corpus, is a language resource used in linguistics that consists of a sizable collection of organized texts. The corpus can be tagged or untagged. Currently, there are different corpora for different languages. For the English language, three main tagged corpora are consistently used for training and testing part-of-speech taggers. These are the Brown, WSJ/Wall Street Journal, and Tyco Brahe corpora, each having one million words, and the Switchboard corpus, which consists of two million words [13][41]. So far, the researcher has not gotten a ready corpus for Sidaama. So there was a need to prepare a corpus for the language.

The corpus used in this research was collected from different sources. Namely, Fables of Sidaama and News of Sidama Language from Sidama Regional State Culture and Tourism Bureau, selected reading passages taken from Grade 3–12 Sidama language students' textbooks containing different topics from different areas, and some passages from the Bible in Sidaama. The texts for the corpus came in various formats. Text from textbooks comes as pdf files, which are converted to word files using pdf to word converter software. Some other texts came in the form of Word documents. These documents are converted to the form of a plain text file and organized to form a corpus. The sentences in the corpus were randomized using Notepad++. It consists of 9,660 sentences and 130.883 tokens/words and symbols used for training and testing.

Before the POS tagging of the raw text file, the POS tags are found, the words are normalized, tokenized, hyphenated words are found and paid attention to, and the Notepad++ is used to mix up the order of the sentences.

Four Sidama language experts participated in the manual POS tagging and verification of the raw text file. Those experts are from Hawassa University's Sidama language department and Hawassa Teachers College.

To tag all the raw text, the incremental tagging approach is implemented. That means 10% of the raw texts are annotated by first-step annotators at the sentence level, checked by a senior linguist, and added to the training set after it is corrected. Similarly, the second 10 % of raw texts are annotated by first-step annotators, checked by a senior linguist, and added to the training set after it is corrected. These steps were repeated until the whole raw text was annotated. These POS-tagged texts are manually verified and validated.

Finally, three custom corpora: sidama (main corpus), sidamatrn (training set), and sidamatst (training set) are prepared and added to the corpora in NLTK. The import method is used to access and use these corpora.

```
>>> from nltk.corpus import sidama
>>> from nltk.corpus import sidamatrn
>>> from nltk.corpus import sidamatst
```

The tagged corpus preparation process is shown in figure 4.1

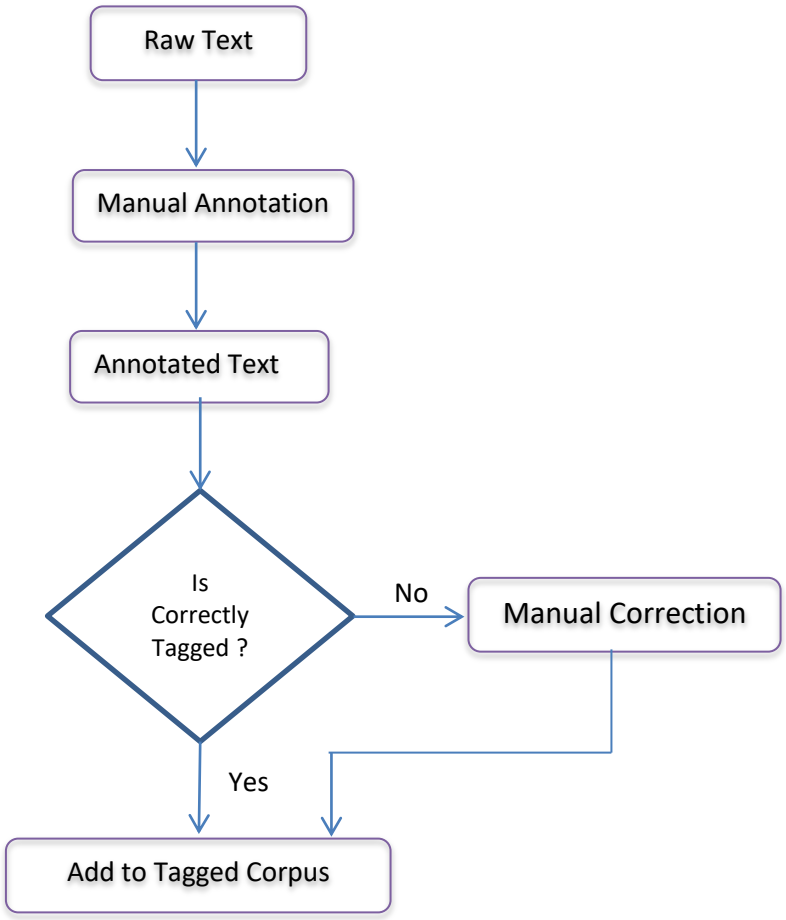


Figure 4.1 The corpus preparation process

Comparison of corpora accuracies

The NLTK accuracy of the prepared Sidaama corpus is calculated and compared to that of the standard Brown and Treebank corpora.

Table 4.1 NLTK Taggers Accuracies of different corpora

Corpus Name	Default (%)	Unigram (%)	Bigram (%)	Trigram (%)	Combination (%)
Brown	13.13	88.49	35.16	20.3	91.3
Sidaama	30.33	89.88	53.87	47.72	93.26
Treebank	13.08	86.28	13.46	7.93	88.99

Table 4.1 shows all taggers perform better on the Sidaama corpus than the rest of the corpora. As in [42]. Unigram Tagger performs better than the other taggers when applied alone on all three corpora.

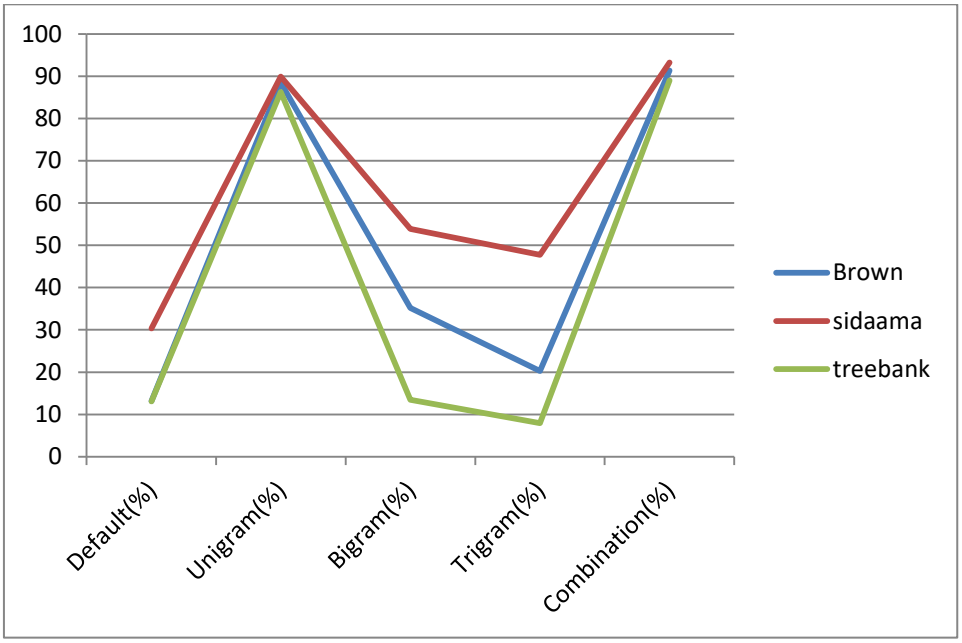


Figure 3.2 NLTK Taggers Accuracies of different corpora

4.2.2. Identification of Sidaama Tag set

A tag set is a set of tags from which the tagger is meant to decide whether to connect to the relevant word [2].

Varieties of POS tag sets are developed by totally different organizations. Due to the differences in language features, objectives, and purposes, the tag set developed for one language can't be used for another.

Since the tag sets for other languages cannot be directly used due to the nature of the language and no tag set is ready-made for Sidaama, the researcher has identified 31 tags. The majority of those tags are from the Penn Treebank tag set [22] [41].Some new tags are added to the tag

set based on the nature of the language. These tags are noun plus auxiliary (NNA), noun plus conjunction (NNC), adjunctive plus auxiliary (JJA), adjective plus conjunction (JJC), adjective plus female (JJF), adjective plus male (JJM), and verb plus auxiliary (VBA). The Sidaama tag set is shown in table 4.2.

Table 4.2 The Sidaama tag set

No.	Tag	Description	Example
1.	CC	Coordinating conjunction	woy 'or', nna 'and'
2.	SC	Subordinate Conjunction	Kayinni 'but', ikkollana 'however'
2.	CD	Cardinal/ordinal number	Mite 'one', Umihu 'first'
3.	EX	Existential	Koo"o 'there', konne 'here'
5	JJ	Adjective	Akkala 'old, aged' xuma 'beautiful'
6	JJA	Adjective + Auxiliary	haranchoho, haranchote "it's/he's/she's short"
7	JJC	Adjective, comparative	labbanno 'looks like', gedee 'as'
8.	JJCC	Adjective + Conjunction	Akkalanna 'old' and
9.	JJS	Adjective, superlative	Roore 'the best'
10	JJF	Adjective + Female	Ballitte 'one who is blind', bukaame 'dusty'
11	JJM	Adjective + Male	Ballichcha 'one who is blind', bukaame 'dusty'
12.	NN	Noun, singular, or mass	Arrishsho 'sun', tima 'bread',
13.	NNA	Noun + Auxiliary	Mineeti 'it's my home, miniraati 'it's for my home
14.	NNC	Noun + Conjunction	Minenna 'house and
15	NNS	Noun, plural	Cea 'birds', timma 'bread'
16	NNP	Proper noun	Latamo, Takilu
17	NNPC	Proper noun + Conjunction	Latamonna, Takilunna
18.	PRP	Personal pronoun singular	Ane 'I', ise 'she' , isi 'he'
19	PRPS	Personal pronoun plural	Insa 'they' ki'ne 'you', ninke 'we'

20	PRP$	Possessive pronoun	Anehu ‘mine’ , isihu ‘his’
21.	RB	Adverb	Soodo ‘morning’, ga’a ‘tomorrow’
22.	RBR	Adverb, comparative	Muddamaanchu/faster/
23	RBS	Adverb, superlative	Addintanni muddamancho /fastest/
24	VB	Verb, the base form	Ita ‘to eat’, aga ‘to drink’
25	VBA	Verb + Auxiliary	Itirooti ‘if he may eat’, iteenaati ‘after he had eaten’
26	VBD	Verb, past participle (all persons)	Itoommo ‘I have eaten’, agoommo ‘I have drunk’ Ittinoonnni ‘you(pl) have eaten’, Itootto or Itootta ‘You(sg) have eaten ’intoommo ‘we have eaten’, Itino ‘He has eaten’, Ittino ‘She has eaten’, Insa itino/ittino ‘They have eaten’
27	VBG	Verb, gerund, or present participle	Itanni… or Itanni noommo ‘I’m eating, Aganni noommo “I’m drinking
28	VBN	Verb, past tense (all persons)	Itummo “I ate’, agummo ‘I drunk’, Iti “He/They ate”, Ittu “She/They ate”, Ittitto “You ate”, Ittini (You(pl) ate)
29	VBZ	Verb, all persons (present or future)	Itanno ‘he eats or will eat’ Agganno ‘she drinks, /they drink/ or will drink’, Iteemmo ‘I eat or will eat Inteemmo ‘we eat or will eat’, Itatto/itatta ‘you (sg) eats or will eat’, Ittinanni ‘you(pl) eat or will eat
30	SYM	Symbol	%, (,)
31	PUN	All punctuations	Taxxeessumalaate (,), UurrishshuMalaate (.), DarbuMalaate (!), MaqishshuMalaate (“ “), HaranchoHawiittoteBicamme (-), IttisuMalaate (;)

4.2.3. The n-gram

The n-gram approach, which calculates the probability of a given sequence of tags, can be used as an alternative to the word frequency approach [14]. The best tag can be determined for a word by finding out the probability that it occurs with the n previous tags, where n is often set to 1, 2, or 3. These models are termed unigram, bigram, and trigram, respectively. The unigram tagger is a simple statistical tagging algorithm. To find the most probable tag for every single word, a unigram POS tagger computes and stores the frequency of tags utilized for every single word established in the tagged training corpus [19][30]. The unigram tagger will give the default tag UNK (Unknown) to any token that wasn't seen in the training data [14]. The bigram model performs POS tagging to determine the most likely tag for a word, given the previous tag. The trigram model performs POS tagging to determine the most likely tag for a word, given the previous two tags. The Hidden Markov Model (HMM) underpins the stochastic approach.

4.2.4. Hidden Markov Model (HMM)

According to [20], HMM is a stochastic model and is an extension of the Markov chain, in which the input symbols are not the same as the states. So, to understand HMM, we need to understand the Markov chain.

Hidden Markov Models (HMM) are well-known generative probabilistic sequences or statistical models that are frequently used for POS tagging [13][18]. The HMM-based POS tagger assigns the best sequence of tags to the entire text of the test set [14].

A hidden Markov model (HMM)-based tagger assigns POS tags by searching for the most likely tag for each word in a sentence [19]. The HMM tagger assigns the most optimal tag sequence given the word sequence [21].

In HMM, if we have a sequence of words, each with one or more potential tags, then we can choose the most likely sequence of tags by calculating the probabilities of all possible sequences of tags, and then selecting the sequence with the highest probability.

The HMM tagger's task is to select the tag sequence that maximizes equation 4.1 [19] [22] [23] for a given input sequence of words.

P(word/tag) * P(tag/previous n tags) …… 4.1

Most frequent tag (likelihood) — N-gram (a prior)

The transition probabilities (P(tag/previous n-tags)) are affected by the states, and thus by pairs of tags [11].

The General Definition of HMM consists of the following five components: [12][21][22]. {Q,O,A,B,π} namely:

$Q = \{q_1, q_2, q_3, \ldots, q_N\}$ a set of N states

$O = o_1, o_2, \ldots, o_M$ a sequence of M observations, each one drawn from a vocabulary $V = v_1, v_2, \ldots, v_N$

$A = \{ a_{ij} \}$ a transition probability matrix A, each a_{ij} representing the probability of moving from state i to state j, such that $\sum_{j-1}^{n} a_{ij} = 1, \quad for\ all\ i$

$1 \le i, j \le N, \quad a_{ij} \ge 0$

$B = b_i(o_t)$ A set of observation likelihoods, also called emission probabilities, each expressing the probability of an observation o_t being generated from state i.

$\pi = \pi_1 \pi_2 \cdots \pi_N$ an initial probability distribution over states. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. And $\sum_{i=1}^{n} \pi_i = 1$

The HMM is completely determined by three probability distributions: the initial probability (π), the state transition probabilities (A), and the emission probability (B), and can be modeled compactly $\lambda = (A, B, \pi)$.

The three problems of HMM

Given an HMM model with N states and V vocabulary, there are three basic HMM problems that must be solved before the model can be used in real-world applications [23][24].

Problem 1: Evaluation Problem

How can the probability of the observation sequence be efficiently computed, given the HMM model $\lambda = (A, B, \pi)$ and the observation sequence $O = \{o_1, o_2, \ldots, o_T\}$ given the HMM model ($P(O \mid \lambda)$)?

This issue can be seen as a scoring exercise for how well a given model fits a set of observations. When choosing between numerous competing models, this is helpful. The solution to the problem allows choosing the model that best matches the observations.

The solution to problem 1 is provided by the forward-backward algorithm. The forward-backward algorithm is an algorithm for computing the probability of a particular observation sequence in the context of hidden Markov models. It also computes a set of backward probabilities that show, given an initial state, how likely it is that the remaining observations will be made.

Problem 2: Decoding Problem

Given the observation sequence $O=\{o_1,o_2,...,o_T\}$ and the model $\lambda=(A, B,\pi)$, how to choose a corresponding state sequence $Q=q_1q_2q_3,...,q_T$ that is the maximum in some meaningful sense?

This is a decoding problem in which we attempt to find the most likely state sequences that lead to the observation sequences.

The solution to problem 2 is provided by the Viterbi algorithm. The Viterbi Algorithm is an inductive procedure in which, at each instant of time, it keeps the best possible state sequence for each of the N states.

Problem 3: Estimation Problem

Given the observation sequence $O=\{o_1,o_2,...,o_T\}$, how to adjust the model parameters $\lambda=(A, B,\pi)$, such that $P(O\mid\lambda)$ are maximized?

Given an observation sequence or a set of observation sequences, O, the estimation problem involves finding the best model parameter values that specify a model most likely to produce the given sequence. The solution to problem 3 is provided by the Baum-Welch algorithm. The Baum-Welch algorithm is a maximization of the solution to the first problem through the

adjustment of the model's parameters. This optimization criterion is called the maximum likelihood criterion [25].

When a word has more than one possible tag, statistical methods enable to determine the optimal sequence of part-of-speech tags T= t_1, t_2, t_3... t_n, given a sequence of words W=w_1, w_2, w_3, ..., w_n where t_i is the tag corresponding to w_i that maximizes the probability: $P(t_1, t_2, \ldots t_n \mid w_1, w_2, \ldots, w_n)$.

P(t1...tn|w1...wn) is transformed into a collection of more manageable probabilities using the Bayes rule.

$$P(x \mid y) = \frac{P(y \mid x)P(x)}{P(y)} \qquad \textit{Bayes Rule}$$

Equivalently:

$$P(t_1, \ldots t_n \mid w_1, \ldots, w_n) = \frac{P(w_1, \ldots, w_n \mid t_1, t_2, \ldots t_n) P(t_1, \ldots t_n)}{P(w_1, w_2, \ldots, w_n)}$$

Then, the solution to problem 2 (the Viterbi algorithm) can be used to solve the equation for the best tag sequence (T*):

$$T^* = \underset{t_1, t_2 \cdots t_n}{\arg\max}\ P(t_1, \ldots t_n \mid w_1, \ldots, w_n) = \underset{t_1, t_2 \cdots t_n}{\arg\max}\ \frac{P(w_1, \ldots, w_n \mid t_1, t_2, \ldots t_n) P(t_1, \ldots t_n)}{P(w_1, w_2, \ldots, w_n)} \quad \ldots 4.2$$

For a given word sequence $w_1, w_2, \ldots, w_n$, $P(w_1, w_2, \ldots, w_n)$ is a constant (is independent of t_i, hence, does not affect argmax.). Therefore do not need to be calculated and equation 4.2 can be rewritten as:

$$T^* = \underset{t_1,t_2\cdots t_n}{\arg\max}\ P(w_1,w_2,\ldots,w_n \mid t_1,t_2\cdots t_n)P(t_1,t_2\cdots t_n) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ 4.3$$

In the first-order hidden Markov model, there are two simplification assumptions. These are

1) The Markov assumption

   The probability of a given state depends only on the previous state.

   $$P(q_i|q_1\ldots q_{i-1}) = P(q_i|q_{i-1})$$

2) Output Independence assumption

   Second, the likelihood of an output observation (oi) depends only on the state that made the observation (qi) and not on any other states or observations:

Output Independence is defined as: $P(q_1\ldots q_i,\ldots,q_T,o_1,\ldots,o_i,\ldots,o_T) = P(o_i|q_i))$

By the independent assumption, the probability of a word depends only on its POS tag. So

$$P(w_1,w_2,\ldots,w_n \mid t_1,t_2\cdots t_n) \approx \prod_{i=1}^{n} P(w_i \mid t_i) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ 4.4$$

By Markov's assumption, the probability of a tag depends only on the previous tag.

$$P(t_1,t_2\cdots t_n) \approx \prod_{i=1}^{n} P(t_i \mid t_{i-1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ 4.5$$

According to [16], in a trigram the probability of the current tag given the previous two tags

$$P(t_1,t_2\cdots t_n) \approx \prod_{i=1}^{n} P(t_i \mid t_{i-2},t_{i-1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ 4.6$$

Combining these equations 4.4 and equation 4.5, the equation of the best tag sequence for the bigram model can be simplified as:

$$T^* \approx \arg\max_{t_1,\ldots t_n} \prod_{i=1}^{n} P(w_i \mid t_i) P(t_i \mid t_{i-1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.7$$

It is now possible to calculate two probabilities: the probability of a word occurring given its tag $P(w_i \mid t_i)$ and the probability of a tag occurring given a previous tag $P(t_i \mid t_{i-1})$. These probabilities are calculated from a tagged corpus.

The model represented by Equation 4.7 is known as a bigram model of HMM [25]. HMM uses Equation 4.7 to compute the best tag sequence for a sentence by applying the Viterbi algorithm. Equation 4.7 can be graphically represented in figure 4.3.

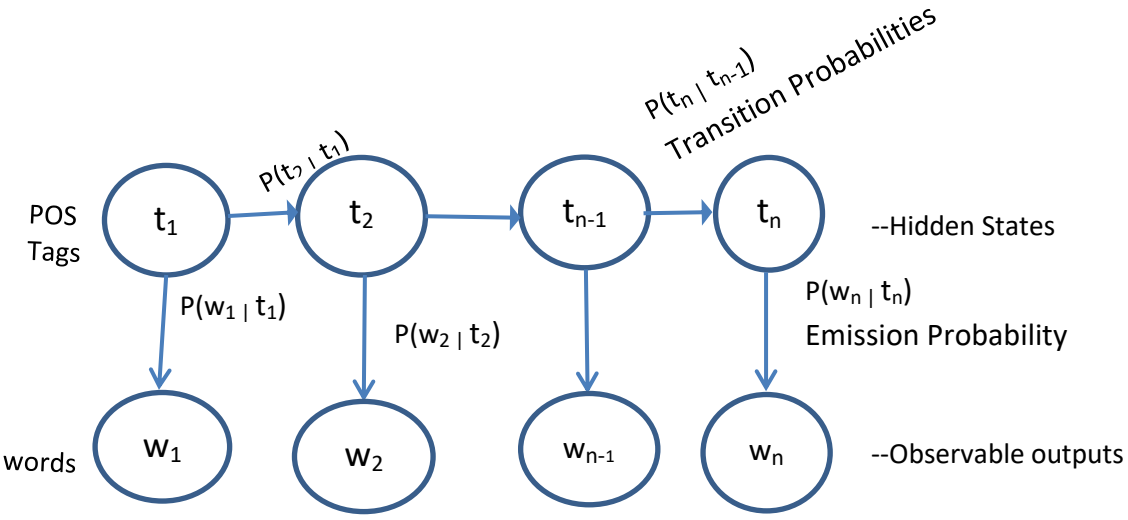


Figure 4.3 The Architecture of HMM [16]

In graphical model notation, circles stand for random variables, and each arrow represents a conditional probability factor in the joint likelihood.

Similarly, by combining equations 4.4 and 4.6, the equation of the best tag sequence for the trigrams model becomes:

$$T^* \approx \arg\max_{t_1,\ldots t_n} \prod_{i=1}^{n} P(w_i \mid t_i) P(t_i \mid t_{i-2}, t_{i-1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.8$$

The probability of the next tag depends only on the previous tag and is called the transition probability. It is the probability of transitioning from one state or tag to another.

The probability of a word depends only on its tag, and $P(w_i | t_i)$ is called the emission probability. This is the probability of emitting a word at a given state.

In transitional probabilities, the information for one part-of-speech category preceded by other categories is developed from the training corpus. The transition probability is the likelihood of a tag given the previous tag(s). Equation 4.5 is used to calculate the bigram transition probability, while equation 4.6 is used to calculate the trigram. For this study, bigram tagging is used.

The emission probabilities are calculated by calculating the relative frequencies of each word in each category in the annotated training corpus. All statistical information is derived automatically from the manually annotated corpus [2].

In the bigram tagger,

In the implementation of $P(t_i|t_{i-1})$ for i = 1 in equations (4.5) and $P(t_i|t_{i-1}, t_{i-2})$ for i = 1 and i = 2 in equation (4.6), a dummy POS tag <START> is added before the first word (two dummies for the trigram case)[27].

As a result, $P(t_i|t_{i-1})$ in equation (4.5) is calculated using $P(t_1 | \text{start})$, and $P(t_i|t_{i-1})$ and $P(t_i|t_{i-2})$ in equation (4.6) are calculated using $P(t_1 | \text{start, start})$ and $P(t_2 | t_1, \text{start})$, respectively.

4.2.5. Maximum likelihood estimation probabilities (MLE)

Utilizing relative frequency estimation, which involves counting the frequencies of word/tag and tag/tag, is the easiest method for calculating the parameters for our HMM. This method is called maximum likelihood estimation. The maximum likelihood estimation (MLE) method is used to estimate the transition and emission probabilities from a tagged corpus [11].

The calculations of these probabilities result in two matrices: the matrix of transition probabilities and the matrix of emission probabilities.

The MLE of transition probability is calculated as:

$$p_{MLE}(t_i \mid t_{i-1}) = \frac{count(t_{i-1}, t_i)}{count(t_{i-1})} \quad \text{.....4.9}$$

The MLE of emission probability is calculated as:

$$p_{MLE}(w_i \mid t_i) = \frac{count(w_i, t_i)}{count(t_i)} \quad \text{.....4.10}$$

Where $count(w_i, t_i)$ *is* number of times word i appears with tag i

$count(w_i)$ *is* number of times word i appears

$count(t_i)$ *is* number of times tag i appears

Likewise, the MLE of the trigram of P(ti|ti-1,ti-2) can be calculated as:

$$P_{MLE}(t_i \mid t_{i-1}, t_{i-2}) = \frac{count(t_{i-2}, t_{i-1}, t_i)}{count(t_{i-2}, t_{i-1})} \quad \text{.....4.11}$$

The sparseness of the data set can make it hard to use the maximum-likelihood estimate, since transitions are given a probability of zero.

4.2.6. The Viterbi Algorithm

The Viterbi algorithm (VA) is a dynamic programming algorithm that applies a table-driven method to solve problems by combining solutions with sub-problems [12]. It was proposed by Andrew J. Viterbi in 1967. The Viterbi algorithm is used to identify the most likely series of hidden states or Viterbi paths that give rise to a series of observed events. It just keeps the

optimal sub-path of each node at each position in the sequence and discards the others. Since the path with the highest probability will be a path that only includes optimal sub-paths, there is no need to keep sub-paths that are not optimal [2]. Thus, it makes tagging much more efficient and reduces the complexity of examining every full path through a trellis by recursively finding partial probabilities for the most likely path from one state to the next in terms of time and memory. A trellis shows the related search space, which is a matrix of all the hidden states and the links (transitions) between them along a sequence of observed states. The Viterbi algorithm sets up a probability matrix with one column for each observation t and one row for each state in the state graph [26]. It requires three steps to search for and identify the one complete and most probable route through the trellis [18]. Those steps are initialization, induction, and termination.

Viterbi Algorithm

Given a sentence $W = \{w_1, w_2, w_3, ,... w_N\}$ and a tag set of length T.

Let us define $\varphi(n,i)$ to be the highest probability of a single path among all the paths ending at word n with tag t_i and a variable $\delta(n,i)$ that allows keeping the track of the best path ending at word n and tag t_i

Step1: Initialization

$$\varphi(1,i) = P(t_i | < s >) P(w_1 | t_i))$$

Where $P(t_i | < s >)$ *is the probability of starting with tag* t_i

$$\delta(1,i) = "",\ i = 1,2,...,T$$

Step2: Iteration or recursion

$$\varphi(n,i) = \max_{1 \le i \le T} \varphi(n-1,i) P(t_j \mid t_i) P(w_n \mid t_i),$$

$$\delta(n,i) = \arg\max_{1 \le i \le T} \varphi(n-1,i) P(t_j \mid t_i);$$

$$2 \le n \le N, 1 \le j \le T$$

Step3: Termination

$$q_1^N = \arg\max_{1 \le i \le T} (\varphi(N,i))$$

Then we backtrack from there to find the highest probability path. That means we find

$$Q = (q_1^*, q_2^*, \ldots, q_n^*) \text{ such that } q_n^* = \delta(n+1, q_{n+1}^*),\ n = N-1, \ldots, 2,1$$

4.2.7. Unknown word handling in HMM

The main problems in POS tagging are word ambiguity and unknown words [12]. These problems affect the accuracy of a POS tagger. In tagging, if a word is unseen in the training text or is an unknown word, then its probability is 0. Such zero probabilities will also result in all possible state sequences for an observation sequence containing an unknown word, causing the probability of the whole sequence to be set to zero when multiplying probabilities. Smoothing techniques are employed to prevent zero probabilities. Smoothing algorithms are primarily used to better estimate probabilities when there is insufficient data to estimate probabilities accurately [22]. There are many different smoothing algorithms in the literature to handle this sparseness problem, all of which consist of decreasing the probability assigned to the known event and distributing it. Some of these techniques are Kneser–Ney /KN/smoothing, Lidstone smoothing, and Laplace (add-one) smoothing.

i. Kneser–Ney /KN/ smoothing

The main purpose of the Kneser-Ney smoothing technique is to determine the probability distribution of n-grams in a document based on their histories.. It goes beyond absolute discounting and comes up with a clever way to build the lower-order (backoff) model.

In Kneser-Ney smoothing, the lower-order model is only important if the count in the higher-order model is small or zero and it should be optimized for that purpose if that is the case.

ii. Laplace smoothing / Add-One smoothing/

Laplace Smoothing is the simple smoothing method for data sparseness. Laplace smoothing is a straight forward smoothing method for sparse data. In this method, one is added for each bigram frequency count seen or unseen in the training data and the size of the vocabulary in the training set.

Using equation 4.10, Laplace's estimation becomes.

$$P_{Lap}(t_i \mid t_{i-1}) = \frac{count(t_{i-1}, t_i) + 1}{count(t_{i-1}) + |V|} \quad \text{.....} \quad 4.12$$

where |V| is the vocabulary size.

The main problem with the Laplace method is that it assigns overvalued probabilities to unseen bigrams. In this research, Laplace smoothing is used when estimating the transition probabilities.

4.2.8. Evaluation

There are many different types of experiments to evaluate the performance of the method, such as precision, recall, F1-score, and accuracy[41].

Accuracy

Accuracy tells you how accurate the model is as a whole, or how many samples out of all of them were correctly categorized by the classifier. The accuracy of the tagger is the number of correctly tagged tokens divided by the total number of tagged tokens in the text file. Its range is between 0 and 100 (in percentage). It is calculated by using the following formula as used in [13].

$$\text{Accuracy}(\%) = \frac{\text{No. of correctly tagged token}}{\text{Total no. of POS tags in the text}} * 100 \quad \text{..... 4.13}$$

The performance of the model can also be tested by introducing the binary classification confusion matrix and using the concepts of the classification measures precision, recall, and F1-score. Based on the predicted label, the matrix elements are put into one of four groups: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) [43][44]. These values are calculated by using confusion matrix for binary classification.

Table 4.3 Confusion Matrix for Binary Classification

		True class: Positive	True class: Negative
Predicted Class	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

True Positive (TP): It refers to the number of predictions where the classifier correctly predicts the positive class as positive. In other words, the model predicted true and it is also true.

True Negative (TN): It refers to the number of predictions where the classifier correctly predicts the negative class as negative. It means, the model predicted false and it is false.

False Positive (FP): It refers to the number of predictions where the classifier incorrectly predicts the negative class as positive. It means, the model predicted true and it is false

False Negative (FN): It refers to the number of predictions where the classifier incorrectly predicts the positive class as negative. In other words, the model predicted false and it is true.

These concepts are used to define and calculate precision, recall, F1-Score and Accuracy of the tagger.

Precision

Precision is the calculation of the number of sentences that are correctly labeled from the entire set of data with which our method has been identified. The higher the precision, the more accurate is the system. Precision can be represented by equation 4.14.

$$\text{Precision} = \frac{TP}{TP + FP} \quad \text{.....4.14}$$

Recall

Recall is a measurement of how many sentences from the whole corpus of input text that the algorithm properly identified. Recall can be represented by equation 4.15

$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{.....4.15}$$

F1- Score

The F1-score is a measure of the accuracy of the system. F1-Score can be represented by equation 4.16

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad \text{.....4.16}$$

The accuracy of the model can also be represented by equation 4.17.

$$Accuray = \frac{TP + TN}{TP + TN + FP + FN} \quad \text{.....4.17}$$

In equation 4.17, the denominator (TP +TN +FP+FN) is equivalent to the total number of tagged tokens.

In this work the confusion matrix is a multi-class confusion matrix. It is not possible to implement directly the confusion matrix for binary classification. In a multi-class confusion matrix, to calculate the precision, recall, F1-scoe and accuracy of a label, we refer the row and column of the label as indicated in table 14. The values on the diagonal of the matrix from left upper corner to the right lower corner (i.e. C11, C22, C33, C44, …, CNN) for the confusion matrix are taken as TP. Unlike a confusion matrix for binary classification, a confusion matrix for multi-class classification requires those values to be calculated for each class.

In table 14, the shaded part corresponding to label C2 shows the TP, FP and FN while the un-shaded region, represent TN. So, it is hard to calculate TN and accuracy manually for multi-class confusion matrix.

Table 4.4 Multiclass classification problem confusion matrix

		Predicted Class					
		C1	C2	C3	C$	…	CN
Actual Class	C1	C11	FP	C13	C14	…	C1N
	C2	FN	TP	FN	FN	FN	FN
	C3	C31	FP	C33	C34	…	C3N
	C4	C41	FP	C43	C44	…	C4N
			FP				
	CN	CN1	FP	CN3	CN4	…	CNN

4.2.9. Cross-validation

Cross-validation is a technique used to prevent over-fitting and gauge the model's performance using fresh data. The cross-validation trials illustrate the findings for sentences with only known words, sentences with unknown words, and all sentences[27].

According to [28], k-fold cross-validation is a validation where a given dataset is split into a k number of folds where each fold is used as a testing set at some point.

In k-fold cross-validation, the value of k shouldn't be too little or too high. Depending on the size of the data, we prefer to choose five to ten. For a larger value of k, the model becomes less biased. For example, the following table shows the average accuracy of folds 5 through 12 for the Sidaama corpus.

The Value of k	5	6	7	8	9	10	11	12
Average Accuracy	90.67	90.61	90.81	90.93	91.12	91.2	91.16	91.18

In the above table, the average value of k-folds varies throughout the table, but it looks like the variation becomes less starting from k = 10 to the end. As a result, k = 10 is an appropriate value for cross-validation.

For 10-fold cross-validation, there are 10 iterations. Here, in the k^{th} iteration, the k^{th}-fold is used to test the model, whereas the rest are used to train the model. For example, for the initial iteration, the first fold is used to test the model, and the rest are used to train the model. In the second iteration, the second fold is used as the testing set, while the rest are used to train the model. Repeat this procedure until all 10 of the folds have been used as the testing set.

4.2.10. The Hidden Markov Model for Sidaama

An HMM is a finite automaton with stochastic transitions and observations. The automation models a probabilistic generator procedure whereby a sequence of observations is produced by beginning in some state, emitting an observation selected by that state, passing to a new state, and emitting another observation. This process continues until the designated state is reached.

To develop an HMM for the Sidama language and calculate how likely it is to change and how likely it is to emit, let's look at the following set of tagged sentences:

Table 4.5 Sample Sentences with the corresponding tags

Owatonna	aana	more	
NNP	VB	VB	
aana	dikko	hadhu	
NNP	NN	VB	
mayra	woratto	qorete	aana
RB	VB	NN	RB
beettu	barcimu	aana	No
NN	NN	RB	VB

Table 4.6 Sample Transition Probabilities

	NNP	NN	RB	VB	</s>
<s>	1/2	1/4	1/4	0	0
NNP	0	1/2	0	1/2	0
NN	0	1/4	1/2	1/4	0
RB	0	0	0	2/3	1/3
VB	0	1/3	0	1/3	1/3

Table 4.7 Sample Emission Probabilities

	NNP	NN	RB	VB
Daraaro	1/3	0	0	0
aana	1/3	0	1/2	1/4
dikko	0	1/4	0	0
hadhu	0	0	0	1/4
mayra	0	0	1/2	0
woratto	0	0	0	1/4
qorete	1/3	1/4	0	0
beettu	0	1/4	0	0
barcimu	0	1/4	0	0
no	0	0	0	1/4

HMM for Sidaama is modeled using table 4.6 Sample Transition Probabilities and table 4.7 Sample Emission Probabilities as Probabilistic Finite State Automata shown in Figure 4.4

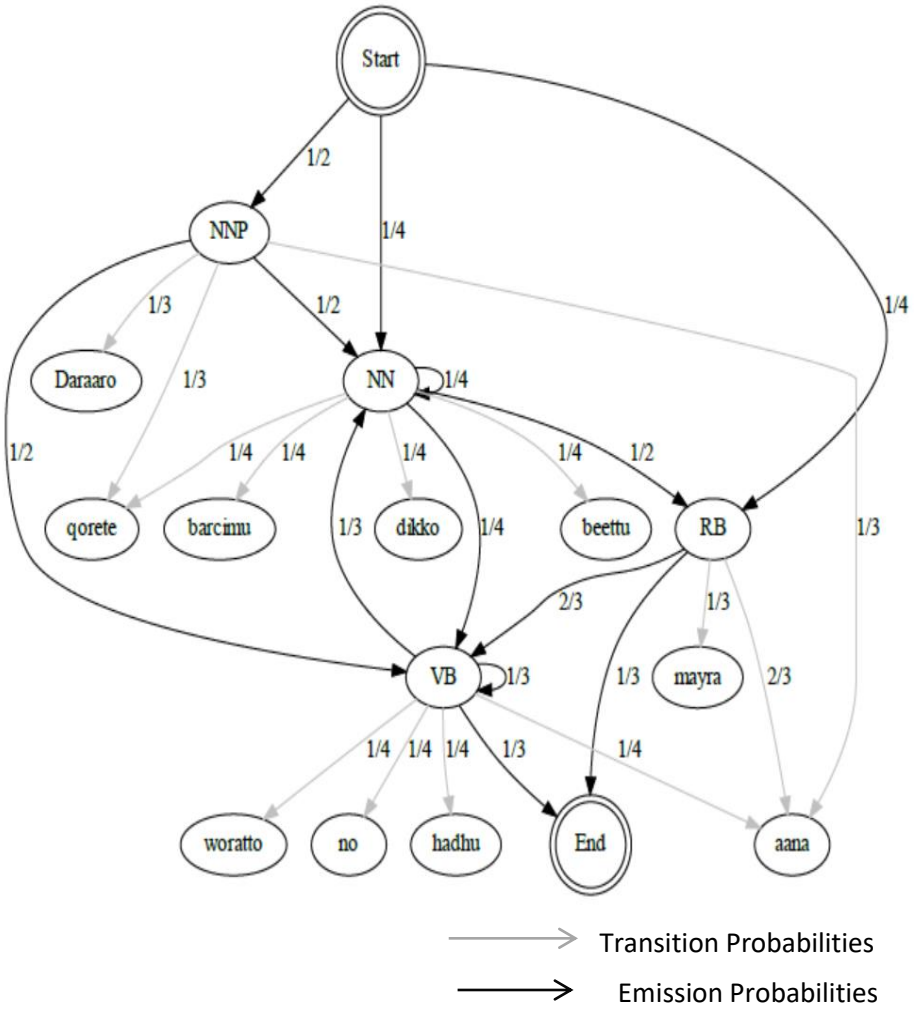
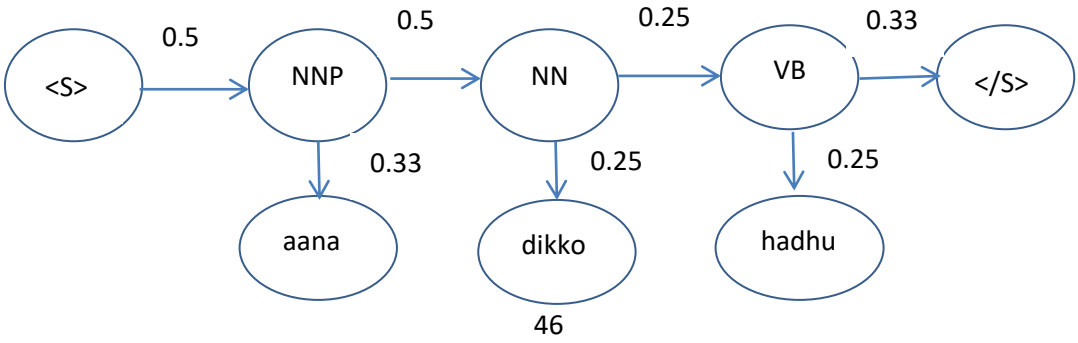
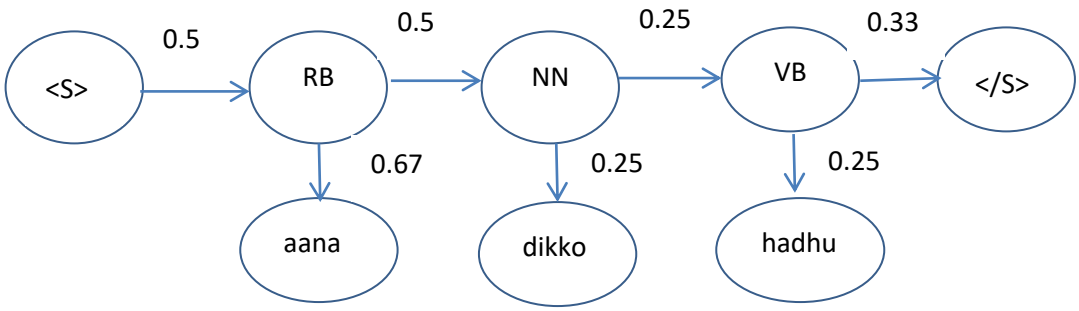


Figure 4.4 HMM for Sidaama as Probabilistic Finite State Automata

To determine the best tag sequence of the sentence "aana dikko hadhu", from the table, one can see that aana has 3 tags namely NNP, RB, and VB.



The product of the probabilities = 0.5×0.33×0.5×0,25×0.25×0.25×0.33 = 0.00042539



The product of the probabilities = 0.5´0.67´0.5´0.25´0.25´0.25´0.33 = 0.000431836

In the third case, P(VB/<S>) = 0. Due to this, the product of the probabilities is 0. This shows that the maximum result is 0.000431836. Therefore, the best tag sequence for the word sequence 'aana dikko hadhu' is RB, NN, and VB. But each word has three possible tags. So, there are $3^3 = 27$ possible comparisons.

If N = the number of different states, and M = the number of observables, then the computational complexity is $O(N^M)$. This implies that as the number of states and observables increases exponentially, the value of N^M becomes very large and computationally expensive. This shows that the hidden Markov model is not very efficient. For the Viterbi algorithm, the computational complexity is $O(M^2N)$. This is less complex than HMM. For example, for a fixed number of states = 4 and the length of observables = 3 through 10, the table shows the values for the Viterbi algorithm are very small as compared to the values for HMM. Therefore, the Viterbi algorithm is preferable to implement.

N	3	4	5	6	7	8	9	10
M	4	4	4	4	4	4	4	4
HMM	81	256	625	1296	2401	4096	6561	10000
Viterbi	36	64	100	144	196	256	324	400

4.2.11. The Sidaama HMM POS Tagger

The Sidaama HMM POS Tagger includes the preprocessing, HMM training, and evaluation steps using the manually tagged corpus and the raw text to be tagged. The tag set of Sidaama is extracted from the manually tagged corpus. The components of the Sidaama HMM POS tagger are depicted in the following block diagram.

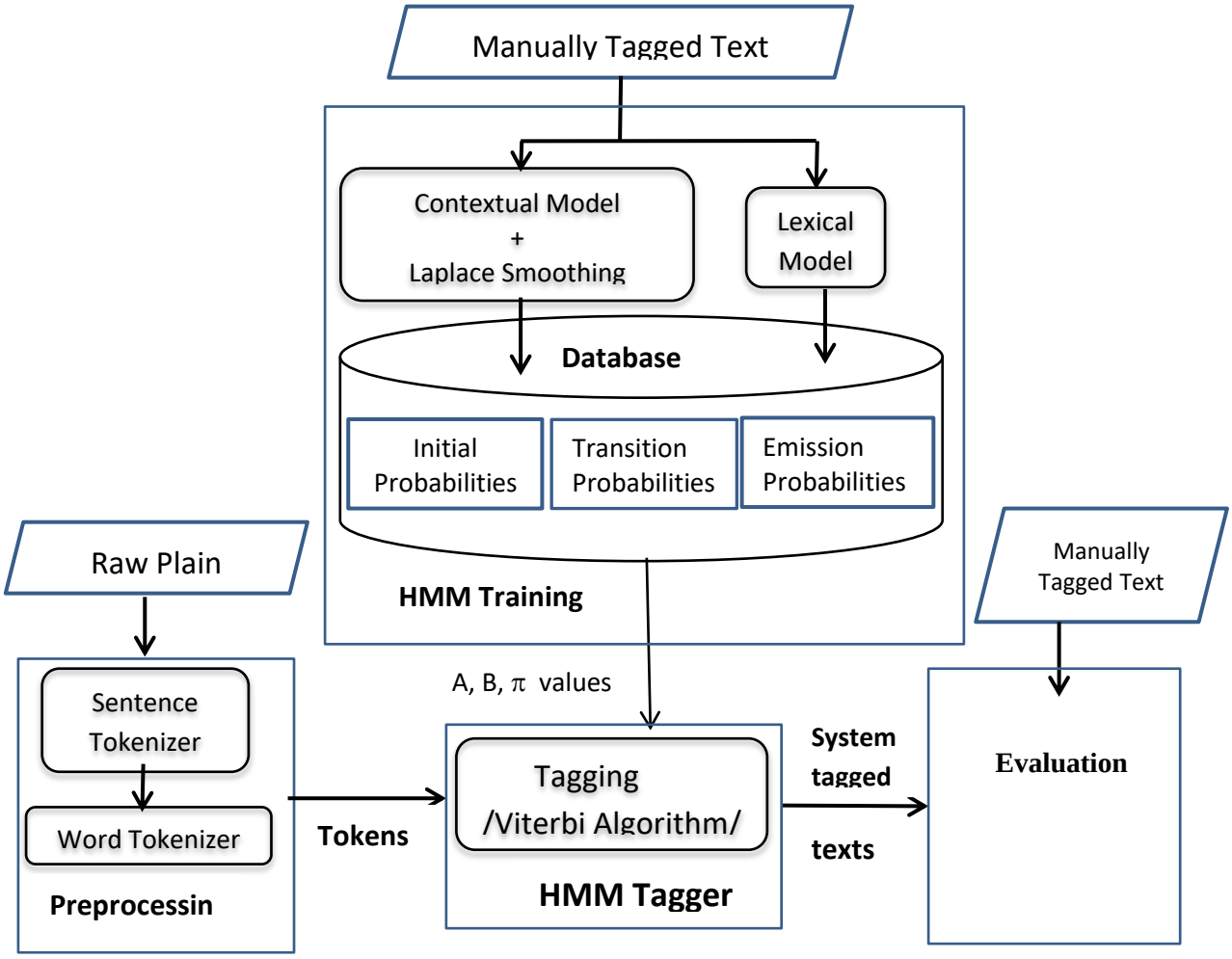


Figure 4.5 The block diagram of the Sidaama Tagger [10]

Description of the block diagram's elements

I. Preprocessing

The preprocessing step is required for the identification of words and other relevant tokens in the text as well as for the markup of sentence boundaries. In this study, the sentence boundaries were based on the assumption that the beginning of a new sentence was indicated by a capital letter and the end of the sentence was preceded by a full stop, exclamation mark, or question mark [10].

Some of the pre-processing steps in this work are:

- Change the non-starting letter of a sentence and the first letter and non-proper noun words into lower case to avoid different tags for the same word.
- Sentence tokenization (splitting the input text into sentences)
- Tokenization (breaking the input text stream or test set into tokens, i.e., words, phrases, symbols, punctuation, and other meaningful elements by keeping whitespace between them) These tokens are taken as input for the tagger.

II. Manually Tagged Corpus

Tagged texts are usually stored as a sequence of tokens separated by white space. The manually tagged corpus is required to perform part of speech tagging by applying the stochastic technique. The manually tagged text for this tagger is the result of section 4.2.1 corpus preparation. This corpus is divided into two parts: 90% for training and 10% for testing. The training set is used to train the HMM tagger..

III. The HMM Training

The HMM training is based on a supervised learning approach. It accepts manually tagged text discussed in Section 4.2.1 and estimates the transition (contextual) probabilities of the occurrence of each tag-tag pair and the emission (lexical) probabilities of tag-word pairs in a given context by using maximum likelihood estimation (MLE) equations 2.9 and using maximum likelihood estimation (MLE) equations 2.9 and 2.10, respectively. These values are stored and sent to the Viterbi algorithm.

IV. The HMM tagger

The HMM tagger assigns the most probable POS tag for each word by calculating the emission probability and transition probability. The tagger accepts values for the HMM model parameters. The tagger learns from the training module's emission probability and transition probability how to find the best tag sequence and label each word in a sentence with the right tag.

Automated taggers are trained on an extract of an annotated corpus, and therefore the assessed performance is on a separate extract of a corpus, stripped of annotations (the testing set).

V. Evaluation

The evaluation extracts the unigram, bigram, and HMM probabilities from the tagged corpus. The frequency of each word and pair of words in the corpus is used to figure out the n-gram probabilities.

4.2.12. The Prototype Implementation

The Sidaama POS tagger is an application developed to tag plain texts in the Sidama language, extract words for a specific POS, and provide other related information about the system. The end-users of this system are expected to have different levels of experience with computerized systems. Due to this, the Sidaama POS tagger user interface is designed as simply as possible so that it is easy for the end-users to understand and use. The user interface is designed by applying Python 3.7.0 with Tkinter.

The window of the tagger directly shows the main interface, which contains a text input field to take the input sentence, the main control button to run the system process, the tokenized output field, the tagged output field, the best tag sequence output field, and the probability of the input sentence.

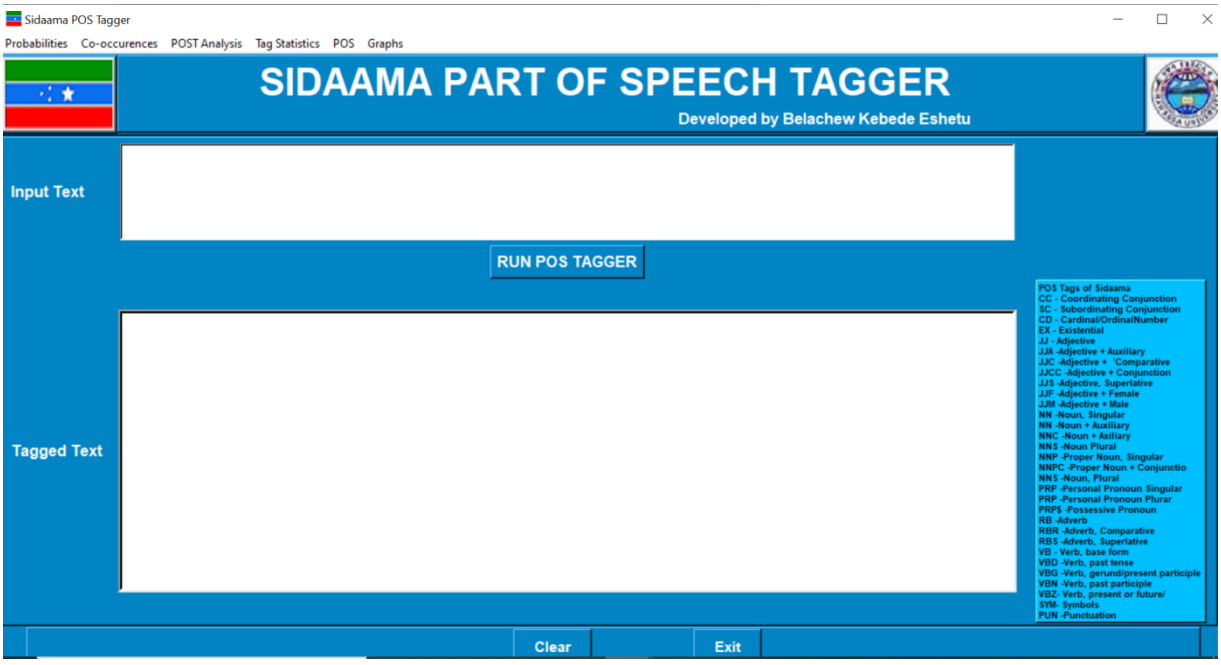


Figure 4.6 Prototype of Sidaama POS Tagger

Two sentences are taken from the corpus and tagged by the Sidaama POS tagger. The output field of Figure 4.6 displays the sample outcome.

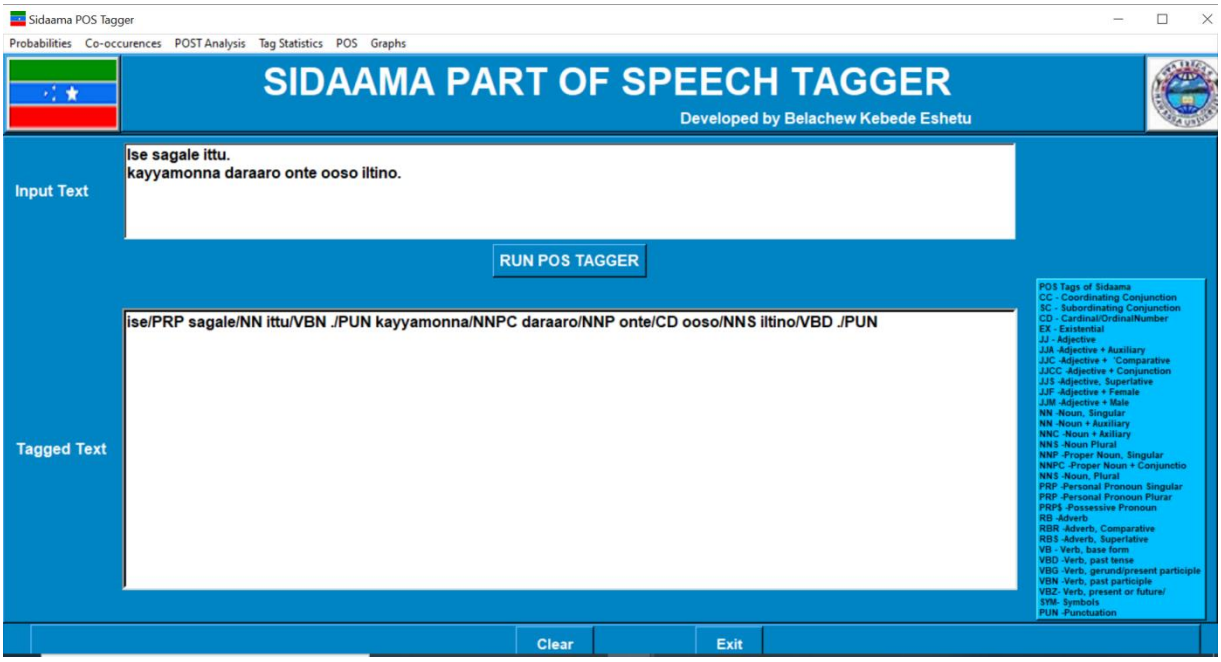


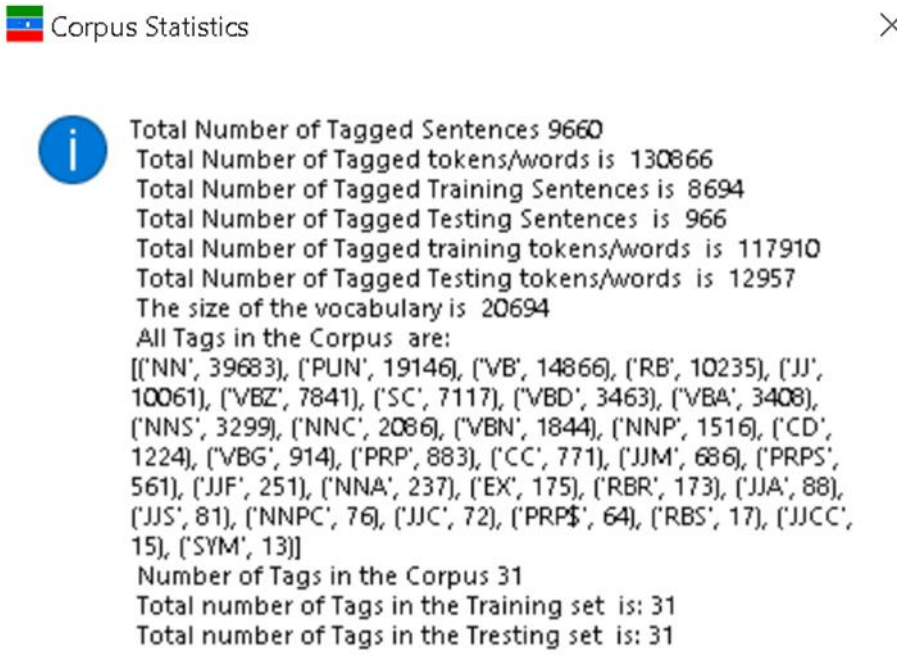
Figure 4.7 Sample Input and result of the tagger

Additional features of the Sidaama POS tagger

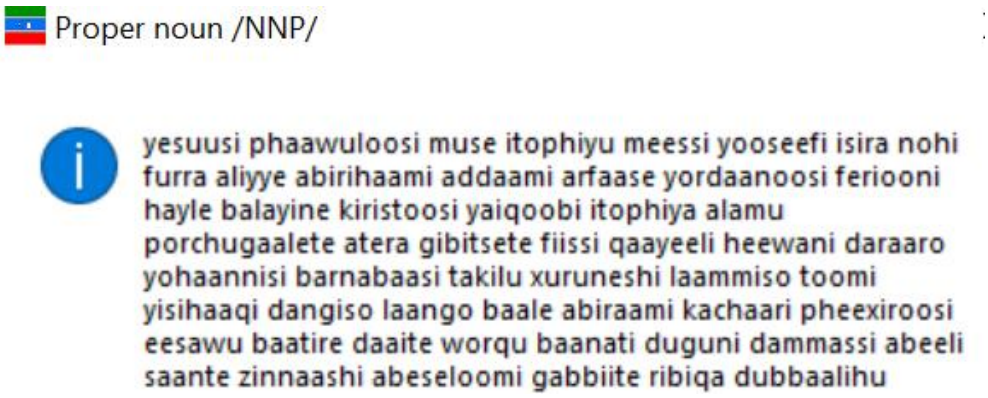
The Sidaama POS tagger also helps to get all the statistics of the corpus, to search for words with a specific tag and tags for a word from the corpus, and to put a word next to a given word or two consecutive words. Moreover, the tagger gives all the graphs used in the thesis.

A. **Extraction of corpus statistics**

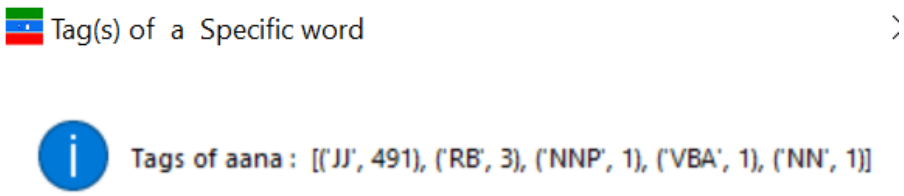
From the entire corpus, training set, and testing set, this function extracts the following data.



B. Searching for words of specific tag e.g. Proper noun



C. Searching for tags of specific words e.g. tags of the word aana



CHAPTER FIVE

RESULTS AND DISCUSSION

In this chapter, we present the results and discuss the parts of the speech tagger of the Sidama language. Different experiments were done to obtain the results that helped analyze the performance of the tagger.

5.1. The test result of the HMM tagger

To conduct the experiments, the researchers divided the entire training set into ten equal parts and performed ten experiments. Firstly, the researcher trained the system using 10% of the training corpus and measured the performance of the system using the test corpus. Then, using an incremental approach, the researcher added 10%, conducted an experiment on 20%, and kept the result. This process is repeated until the entire training set has been completed. The desired performance of the tagger is considered to be the performance measured from the total training set (100%) applied).

Table 5.1 shows the results of the performance of the Sidaama tagger for different experiments conducted on different portions of the training set and their corresponding performances.

Table 5.1 The Results of the Sidaama tagger performance

Training Set	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Performance (%)	75.45	80.62	83.33	85.00	86.48	88.42	88.47	90.25	90.45	91.25

From table 5.1, the performance of the tagger for 100 % of the training corpus is 91.25%.

From the table, one can see that as the size of the corpus increase the accuracy of the HMM tagger also increases.

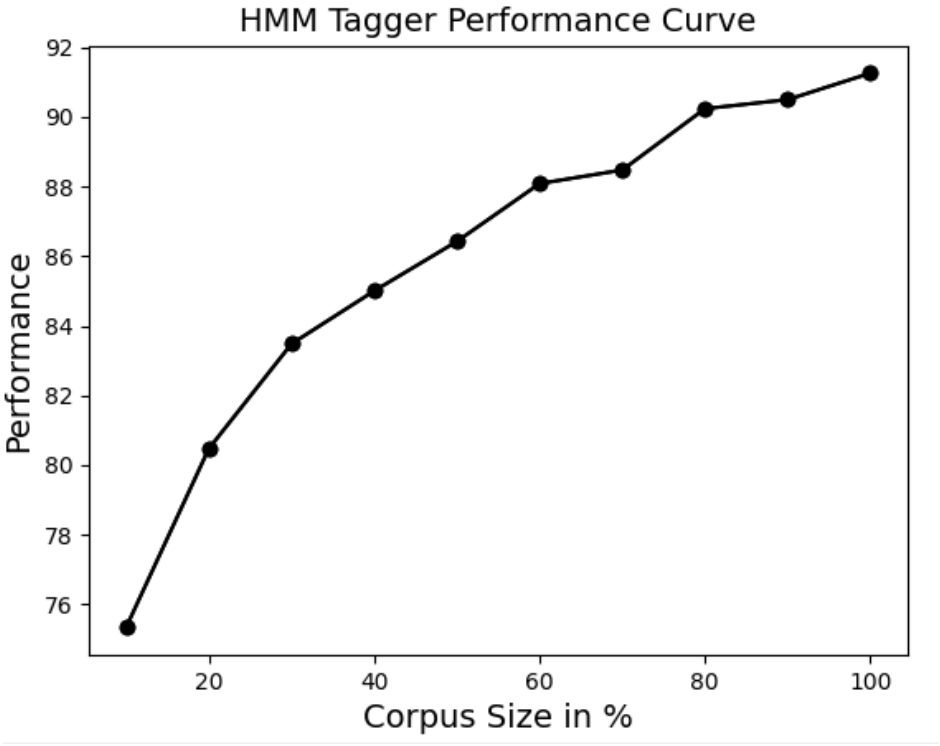


Figure 5.1 Performance curve analysis for HMM tagger

Validation of the Sidaama tagger Performance

This experiment is done to validate the accuracy of the POS taggers described above.

Table 5.2 Ten-Fold Cross-Validation Result

10-Folds	Test Sentences		Accuracy
	From	to	
Fold 0	0	966	91.62
Fold 1	967	1932	90.71
Fold 2	1933	2898	91.77
Fold 3	2899	3864	90.96
Fold 4	3865	4830	90.86
Fold 5	4831	5796	91.32
Fold 6	5797	6762	90.73
Fold 7	6763	7728	91.30
Fold 8	7729	8694	91.30
Fold 9	8695	9660	91.22
Average Accuracy			91.19

The models' performance is calculated as the average of all 10 trials. In table 5.2, the average accuracy of the folds is 91.19%, while the accuracy of the HMM POS tagger is 91.25% with a deviation of 0.06%. The variation between tagger accuracy and average accuracy in cross-validation is very small. So, there is very little over fitting in the training corpus and the testing corpus..

5.2. Performance Analysis of HMM Tagger

To analyze the performance of the Sidaama part of the speech tagger, the frequency of tags within the entire corpus, training set, and testing set are considered. In this work, a tag set of 31 tags was identified and used to train the model. Table 5.3 shows the frequency of POS tags in the corpus. In the table, NN (noun) is the most frequent tag, whereas SYM (symbol) is the least frequent tag.

Table 5.3 The POS tags Frequency

	Tag	Frequency		Tag	Frequency
1	NN	39691	17	CC	772
2	UN	19140	18	JJM	688
3	VB	14870	19	RPS	527
4	RB	10236	20	JJF	251
5	JJ	10063	21	NA	237
6	BZ	7841	22	EX	175
7	SC	7118	23	RBR	173
8	BD	3462	24	JJA	88
9	BA	3409	25	JJS	81
10	NNS	3299	26	NPC	76
11	NC	2086	27	JJC	72
12	VBN	1843	28	RP$	64
13	NP	1514	29	RBS	17
14	CD	1225	30	JJCC	15
15	VBG	914	31	SYM	13
16	PRP	884			

The graph of the frequencies of the POS tags for the entire corpus is depicted in figure 5.2.

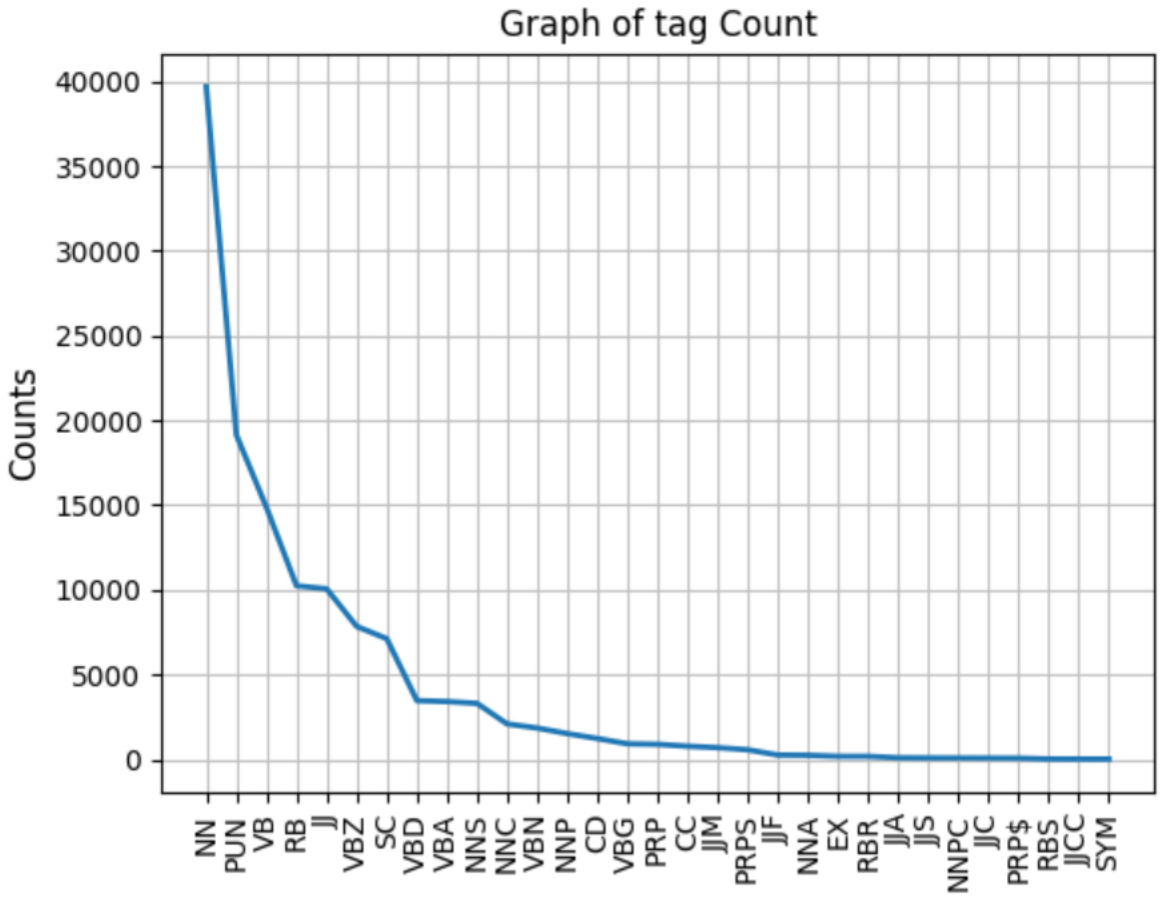


Figure 5.2 The distribution of tags in Sidaama Corpus

Figure 5.2 shows that the most common tag in Sidaama is NN while the least one is SYM. The accuracy of the tagger can be calculated using the classification concept of confusion matrices.

Confusion matrix of the Sidaama HMM POS tagger

A confusion matrix is a table that is used in classification problems to assess where errors in the model were made. The rows correspond to the actual courses for which the results were intended. While the columns represent the predictions we have made. This table makes it simple to identify whose predictions were incorrect.

The classification model's confusion is displayed in the confusion matrix as it generates predictions[43]. It can show a classifier's errors and, even more important, the kinds of mistakes that come from those errors.

The confusion matrix for the tagger is shown in table 5.4. In the table, the most frequent 15 tags from the tag set are used to show the partial table columns and rows, and the rest of the columns and rows are considered others, which include PRP, JJM, CC, PRPS, EX, NNA, JJF, PRP$, RBR, JJC, NNPC, JJA, JJS, JJCC, RBS, and SYM.

Table 5.4 Confusion matrix of HMM tagger

		Test Tags (Predicted Tags)																
		NN	PUN	VB	RB	JJ	VBZ	SC	VBD	NNS	VBA	NNC	VBN	NNP	CD	VBG	others	SUM
Reference Tags (Desired Tags)	NN	**3504**	89	95	31	31	33	12	8	3	49	11	22	8	5	1	3	3905
	PUN	0	**1874**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1874
	VB	45	16	**1303**	19	15	28	5	10	1	18	1	11	6	2	2	0	1482
	RB	16	19	13	**927**	10	17	6	2	2	8	1	3	1	0	1	0	1026
	JJ	7	6	11	7	**941**	8	0	0	3	8	2	1	2	0	1	1	998
	VBZ	4	4	4	9	6	**722**	2	9	0	30	1	7	0	1	0	0	799
	SC	0	0	1	0	1	2	**679**	0	0	0	0	3	1	0	0	0	687
	VBD	4	6	1	0	4	7	1	**307**	0	13	2	2	0	0	1	0	348
	NNS	4	1	3	1	4	2	0	0	**306**	5	1	0	0	0	0	0	327
	VBA	5	3	4	0	2	17	1	4	0	**265**	0	10	0	0	0	0	311
	NNC	18	10	4	2	4	4	1	1	0	3	**170**	2	0	1	0	1	221
	VBN	7	4	3	3	0	12	3	0	0	7	0	**149**	0	0	0	0	188
	NNP	6	4	2	4	1	0	1	0	0	1	0	0	**126**	0	0	0	145
	CD	2	7	9	0	1	2	0	2	0	4	3	0	0	**103**	0	0	133
	VBG	2	3	6	4	2	3	0	0	0	3	1	0	0	1	**82**	**0**	107
		8	5	12	1	1	3	2	0	0	4	2	0	0	0	**0**	**365**	406
	Total tagged																	12957
	Total Correctly Tagged																	11823

In table 5.4, the value of correctly classified tags was displayed in the table's diagonal. The experimental analysis of HMM indicates that from 3,905 NN in the test set, 3,504 were correctly tagged as NN and incorrectly tagged: 89 as PUN, 95 as VB, 31 as RB, 31 as JJ, 33 as VBZ, 12 as SC, 8 as VBD, 3 as NNS, 96 as VBA, 11 as NNC, 22 as VBN, 8 as NNP1 as

VBG and 3 as other tags. Moreover, the above confusion matrix shows that the tagger tagged 11,823 tokens correctly and 1,131 tokens incorrectly. That means the tagger has confused 1,131 tags with different parts of speech tags.

From table 5.4, the number of correctly tagged tokens = 11,823 and

The total tagged tokens in the corpus = 12,957, using equation 5.1

$$\text{Accuracy}(\%) = \frac{\text{No. of correctly tagged token}}{\text{Total no. of POS tags in the text}} * 100$$

$$= \frac{11,823}{12,957} * 100$$

$$= 91.248\%$$

This is approximately 91.25 in rounding to two decimal places.

On the other hand, to calculate precision, recall, F1-score and accuracy of the tagger, we should find TP, FP and FN for each tag and use the corresponding formula. For example, to calculate the precision, recall, F1-scoe and accuracy of the tagger for VB tag in table 5.4. Then, TP = 1,303

FP = 95+13+11+4+1+1+3+4+4+3+2+9+6+12

= 168

FN = 45 + 16 +19+15+28+5+10+1+18+1+11+6+2+2

= 179

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{1303}{1471} = 0.88579$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{1303}{1482} = 0.8792$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recal}}{\text{Precision} + \text{Recall}} = 2 \times \frac{0.88579 * 0.8792}{0.88579 + 0.8792}$$

$$= 0.88086$$

These calculations demonstrate how time-consuming it is to manually calculate the tagger's precision, recall, F1-score, and accuracy. Instead of this manual work, these values can be calculated by using a Python program with the scikit-learn package. The values shown in table 5.7 are calculated using a Python program with scikit-learn.

Table 5 Precision, Recall, F1-Score and Accuracy

Tag	Precision	Recall	F1-score	Support	Tag	Precision	Recall	F1-score	Support
CC	1.00	1.00	1.00	67	PRP	0.98	0.98	0.98	100
CD	0.91	0.77	0.84	133	PRPS	1.00	0.64	0.78	14
EX	1.00	0.96	0.98	23	PRP$	0.93	98.00	0.95	43
JJ	0.92	0.94	0.93	998	PUN	0.91	1.00	0.95	1874
JJA	1.00	0.38	0.55	8	RB	0.92	0.90	0.91	1026
JJC	0.50	0.10	0.17	10	RBR	0.93	1.00	0.97	14
JJCC	1.00	0.50	0.67	2	RBS	1.00	1.00	1.00	1
JJF	1.00	0.78	0.88	18	SC	0.95	0.99	0.97	687
JJM	1.00	0.97	0.99	70	SYM	1.00	1.00	1.00	1
JJS	1.00	0.60	0.75	5	VB	0.89	0.88	0.88	1483
NN	0.97	0.90	0.93	3905	VBA	0.63	0.85	0.73	311
NNA	1.00	0.70	0.82	20	VBD	0.90	0.88	0.89	348
NNC	0.87	0.77	0.82	221	VBG	0.93	0.77	0.85	106
NNP	0.86	87.00	0.87	145	VBN	0.71	0.79	0.75	188
NNPC	1.00	0.70	0.82	10	VBZ	0.84	0.90	0.87	799
NNS	0.97	0.94	0.95	327	Item accuracy			0.91248	12957

Support column indicates the size of the data.

The last experiment is to test the performance of the HMM tagger with the Viterbi algorithm. The performance obtained in this experiment was 98.46%.

CHAPTER SIX

CONCLUSION AND RECOMMENDATION

6.1. Conclusion

Part of the speech tagger is one of the natural language processing (NLP) applications. Part of speech tagging is the process of classifying words into their parts of speech and labeling them accordingly. Parts of speech taggers can be put into three main groups: rule-based, Stochastic, and hybrid.

For this thesis, the HMM approach with the Viterbi algorithm was implemented to develop the part-of-speech tagger for the Sidama language. The two basic and important things required in POS tagging are the corpus and the set of parts of speech tags. So a corpus of 9,660 Sidaama sentences containing 130.847 tokens was collected from different sources and randomized. It is annotated by Sidama language experts using the identified 31 part-of-speech tags.

The corpus is divided into two sets (a training set and a testing set) for testing and training purposes. 90% of the corpus, or 117,932 tokens, are in the training set, and 10%, or 12,952 tokens, are in the testing set.

The Python programming language (Python 3.7.0) with the Natural Language Toolkit (NLTK 3.0.0)) was used to create the custom corpora and develop the tagger for the Sidaama language. Notepad++ is used to randomize the collected corpus. The corpus' NLTK accuracy is calculated and contrasted with that of the common Brown and Treebank corpora. As a result, all taggers perform better on the Sidaama corpus than the rest of the corpora.

The HMM model for the Sidama language is made by looking at a small number of tagged sentences and figuring out how likely each transition is and how likely each emission is.

The Sidaama HMM POS Tagger has been made. It includes the processes of preprocessing, HMM training, tagging, and assessing.

The Sidaama POS tagger is a program created to assign the appropriate tags to words in the Sidama language. It also helps get all of the statistics about the corpus, look for words with a certain tag and tags of a word from the corpus, as well as words next to given phrases or words that come after each other, and other system-related information.

Experiments on a different portion of the training set were conducted to evaluate the performance of the Sidaama HMM tagger, and a performance of 91.25% was obtained. This result was validated using a tenfold cross-validation mechanism with a variation of 0.05%.
On the final experiment, the performance of the tagger with the Viterbi algorithm is computed and obtained 98.46%. This demonstrates that the performance is enhanced by 7.21% via the Viterbi algorithm. This illustrates that the Viterbi algorithm outperforms the HMM in terms of computational speed and accuracy.
In the related work referenced in this thesis, the performance of the taggers using HMM was 92% for Manipuri [13], 89.13% for Tigrigna [3], and 77.19% for Kaffinaanoo [4]. On the other hand, a part-of-speech tagger for Nepali text using HMM and the Viterbi algorithm achieved an accuracy of 95.43% for the Viterbi algorithm[29].

6.2. Recommendation

As described in section 1.6, the corpus for this thesis is taken from the limited sources indicated in section 4.2.2 and is relatively small. On the other hand, in this work, the tag set does not cover gender classes and number classifications related to all parts of speech. In order to convey a token's precise meaning and information, a wider tag set is preferable [12]. Moreover, the Sidama language is morphologically rich. So, knowledge of Sidaama morphology plays an essential role in the improvement of the tagger. As a result, the researcher proposes the following points for future work to improve this work:

- Using wide coverage of domain areas, which is not included in this study, and increasing the size of the training corpus collected from different domain categories
- Extending the tag set that can fully identify gender, number classifications, and tenses
- Combining the HMM-based tagger and the rule-base tagger with the morphology of the language

REFERENCES

[1] S. Teferra, "Addis Ababa University College of Natural and Computational Sciences School of Information Science Development of Part of Speech Tagger Using Hybrid Approuch By: Getachew Emiru a Thesis Submitted in Partial Fulfillment of the Re-Quirement for the Degree o," 2016.

[2] G. Mamo and M. Meshesha, "Parts of Speech Tagging for Afaan Oromo," *Int. J. Adv. Comput. Sci. Appl.*, vol. 1, no. 3, pp. 1–5, 2013, doi: 10.14569/specialissue.2011.010301.

[3] B. Gambäck, F. Olsson, A. A. Argaw, and L. Asker, "Methods for Amharic part-of-speech tagging," no. March, p. 104, 2010, doi: 10.3115/1564508.1564527.

[4] Teklay G/Egziabher, "School of Graduate Studies Part of Speech Tagger for Tigrigna Language Part of Speech Tagger for Tigrigna," 2010.

[5] Z. Mekuria, "School of Graduate Studies College of Natural Sciences Department of Computer Science Design and Development of Part-of-speech Tagger for Kafi-noonoo Language Addis Ababa University School of Graduate Studies College of Natural Sciences Department of Comp," 2013.

[6] K. Desta, "Part of Speech Tagger for Hadiyyisa Language," 2018.

[7] A. Ayana, "Improving Part of Speech Tagger Performance for Sidama Language Research · February 2018 with 51 Reads Abraham Ayana Harbin Institute of Technology," no. February, p. 2018, 2018.

[8] A. F. Wicaksono and P. Ayu, “HMM Based Part-of-Speech Tagger for Bahasa Indonesia,” in *Prooceedings of 4th International MALINDO (Malay and Indonesian Language) Workshop*, 2010, no. June 2014, pp. 1–7.

[9] F. M. Hasan, “COMPARISON OF DIFFERENT POS TAGGING TECHNIQUES FOR SOME SOUTH ASIAN LANGUAGES,” no. December, 2006.

[10] V. G. Ut, “Designing HMM-Based Part-of-Speech Tagger for Lithuanian Language . Designing HMM-Based Part-of-Speech Tagger for Lithuanian Language,” no. January 2004, 2014.

[11] M. Hadni, S. A. Ouatik, A. Lachkar, and M. Meknassi, “Improving Rule-Based Method for Arabic POS Tagging Using HMM Technique,” no. May 2016, pp. 257–269, 2013, doi: 10.5121/csit.2013.3821.

[12] A. F. Wicaksono, P. Ayu, C. Paper, and I. Meical, “HMM Based Part-of-Speech Tagger for Bahasa Indonesia,” *Prooceedings 4th Int. MALINDO (Malay Indones. Lang. Work.*, no. January, pp. 1–7, 2010.

[13] J.-H. M. D. Jurafsky, “Speech and Language Processing. Computer Science, Stanford University,” 2018.

[14] K. R. Singha, B. S. Purkayastha, and K. D. Singha, “Part of Speech Tagging in Manipuri with Hidden Markov Model,” vol. 9, no. 6, pp. 146–149, 2012.

[15] S. Tyagi, G. S. Mishra, and G. Noida, “Statistical Analysis of Part of Speech (Pos) Tagging Algorithm for English Corpus,” vol. 2, no. 3, [Online]. Available:

www.Ijariit.com.

[16] J. Singh, N. Joshi, and I. Mathur, “Part of Speech Tagging of Marathi Text Using Trigram Method,” *Int. J. Adv. Inf. Technol.*, vol. 3, no. 2, pp. 35–41, 2013, doi: 10.5121/ijait.2013.3203.

[17] D. Kumawat, “POS Tagging Approaches : A Comparison,” vol. 118, no. 6, pp. 32–38, 2015.

[18] M. Okhovvat and B. M. Bidgoli, “A hidden Markov model for Persian part-of-speech tagging,” *Procedia Comput. Sci.*, vol. 3, pp. 977–981, 2011, doi: 10.1016/j.procs.2010.12.160.

[19] S. Dutta and B. Arora, “Preprocessing for parts of speech (POS) tagging in dogri language,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 8 Special Issue 3, pp. 114–120, 2019.

[20] R. Forsati and M. Shamsfard, “Hybrid PoS-tagging: A cooperation of evolutionary and statistical approaches,” *Appl. Math. Model.*, vol. 38, no. 13, pp. 3193–3211, 2014, doi: 10.1016/j.apm.2013.11.047.

[21] B. Arslan, “DEVELOPING METHODS FOR PART OF SPEECH TAGGING IN TURKISH,” no. June, pp. 1–45, 2009.

[22] M. Haulrich, “Different Approaches to Unknown Words in a Hidden Markov Model Part-of-Speech Tagger Tagging with HMMs,” pp. 1–8, 2009.

[23] K. Tumilaar, Y. Langi, and A. Rindengan, “Hidden Markov Model,” *d’CARTESIAN*,

vol. 4, no. 1, p. 86, 2015, doi: 10.35799/dc.4.1.2015.8104.

[24] H. Samira, B. Fateh, M. Smaine, and B. Mohamed, “Contributions to HMM-based Speech Recognition Systems,” vol. 4, no. 1, pp. 38–47, 1955.

[25] D. Jurafsky and J. H. Martin, “Speech and language processing: An introduction to speech recognition,” *Comput. Linguist. Nat. Lang. Process. Edn., Prentice Hall, ISBN*, vol. 10, no. 0131873210, pp. 794–800, 2018.

[26] K. Sonai, M. Anbananthen, and J. K. Krishnan, “Comparison of Stochastic and Rule-Based POS Tagging on Malay Online Text,” 2017, doi: 10.3844/ajassp.2017.843.851.

[27] W. B. Demilie, “Parts of Speech Tagger for Awngi Language,” vol. 9, no. 9, 2019.

[28] S. .M, “Cross Validation,” vol. 1, no. Cv, p. 3, 2018.

[29] Archit Yajnik and A. Yajnik, “Part of Speech Tagging Using Statistical Approach for Nepali Text,” *Int. J. Cogn. Lang. Sci.*, vol. 11, no. 1, pp. 76–79, 2017.

[30] S. Dutta *et al.*, “DEVELOPING METHODS FOR PART OF SPEECH TAGGING IN TURKISH,” *Procedia Comput. Sci.*, vol. 3, no. June, pp. 977–981, 2018, doi: 10.1016/j.procs.2010.12.160.

[31] A. B. Tunsisa, “Information Science Development of Part of Speech Tagger for Sidaama Language Information Science Development of Part of,” no. November, 2020, doi: 10.13140/RG.2.2.25853.79845.

[32] M. G. Fekede, “The sociolinguistics and pragmatics of greetings in Sidama,” *J. Lang. Cult.*, vol. 7, no. 3, pp. 28–36, 2016, doi: 10.5897/jlc2015.0353.

[33] Sidama Information and Culture Department, “01 Sidaama-Amharic-English-dictionary.pdf.” Master Printing Pres, Addis Ababa, 2007.

[34] “The History and Culture of Sidama Nation12.pdf.”

[35] W. L. Kumo, *The Sidama Nation: History, Culture and Political Economy*, First. Addis Ababa: Bole Printing Enterprise, 2018.

[36] A. K. Mohammed and A. Pandey, “Sphinx based speech recognition application for Sidama language,” *Proc. 2nd Int. Conf. Inven. Syst. Control. ICISC 2018*, no. Icisc, pp. 197–202, 2018, doi: 10.1109/ICISC.2018.8399063.

[37] K. Kawachi, “A GRAMMAR OF SIDAAMA (SIDAMO), A CUSHITIC LANGUAGE OF ETHIOPIA by Kazuhiro Kawachi,” *ProQuest Inf. Learn. Co.*, 2007.

[38] M. W/Giorgis, *Sidaamu Afiinni: Reading and writing in sidaamu Afoo*. Addis Abbaba: Mega Publishing and Distribution P.L.C., 2019.

[39] A. Kemal, “Speaker Dependent Speech Recognition for Sidaama Language,” 2010.

[40] S. Bird, “NLTK 3.2.5 Documentation,” 2017, [Online]. Available: https://nltk.readthedocs.io/en/latest/.

[41] D. Garrette and J. Baldridge, “Type-Supervised Hidden Markov Models for Part-of-Speech Tagging with Incomplete Tag Dictionaries,” in *Emnlp*, 2012, no. July, pp. 821–831.

[42] S. Yumusak, E. Dogdu, and H. Kodaz, “Tagging Accuracy Analysis on Part-of-Speech Taggers,” *J. Comput. Commun.*, vol. 02, no. 04, pp. 157–162, 2014, doi:

10.4236/jcc.2014.24021.

[43] I. Markoulidakis, G. Kopsiaftis, I. Rallis, and I. Georgoulas, “Multi-Class Confusion Matrix Reduction method and its application on Net Promoter Score classification problem,” *ACM Int. Conf. Proceeding Ser.*, pp. 412–419, 2021, doi: 10.1145/3453892.3461323.

[44] A. Singh and N. Joshi, “Part of speech tagging of Hindi using Markov model,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 6, pp. 1723–1726, 2019.

[45] M. Grandini, E. Bagli, and G. Visani, “Metrics for Multi-Class Classification: an Overview,” pp. 1–17, 2020, [Online]. Available: http://arxiv.org/abs/2008.05756.

APPENDICES

7.1. Appendix A: Untagged Sample of the Corpus

Maganu saadate goga seekke addaaminna heewani uddisiisinsa.

Hakkuno, hakkonni qachira heedhannoti mitte jawaatanna worba mancho daraarooti.

Togo assanke muxxe jajja yaano yihowa ledo muli jaalooma kalaqi'neemmo gede assitannonke.

Ilamoommahu 1966 M.D. anni'yara ilantino ooso giddonni ani leekkite.

Hatteentenni yihowa shaqqillunninna cincatenni dhukasi afinohanna umosi heeshshi assanno soqqamaanchosi jawaachishinosi.

Tini ikkito qaaqquulleho tiiu lophonna fayyimmansa aana gawajjo abbitanno.

Shoole ooso noosihu maarki yinanni rodii togo yiino: sheexaanu maate gawajje songuwa wolqa ba'anno gede assate wo'naalanni no.

Mittu manchi saate me''ete yee xa'mihero mayite qolatto saate kullanniti babbaxxitino doogga no.

Cee'maaleessu allaalaanchi fafu bayichinni qolanno.

Furra gashshooti yannara cimeeyyenna geerru songo ofolte ma'litannoha ikkirono, furra geerru songote ofolle amaalamannokki gede gadadissanni sa'ino yinanni.

1985 M.D. kayise rosunnihanna loosunniha ikke addi addi borreessamme loosira hosa hanafinkunni lamu tonni aieenni kiirsiisi'ranni no.

Hatte balla verziliyo ferguseni yinannihu biraazilete hee'rannohu qullaawu maxaafi rosaanchi maganu mangiste looso kaa'late porchugaale ha'rate hedino.

Laammiso lame ooso ilino.

7.2. Appendix B: Tagged Sample of the corpus

Maganu/NN saadate/NNS goga/NN seekke/VB addaaminna/NNPC heewani/NNP uddisiisinsa/VBN ./PUN

Hakkuno/SC ,/PUN hakkonni/JJ qachira/NN heedhannoti/NN mitte/CD jawaatanna/JJC worba/JJ mancho/NN daraarooti/VB ./PUN

Togo/RB assanke/NN muxxe/RB jajja/NN yaano/VB yihowa/NN ledo/RB muli/JJ jaalooma/RB kalaqi'neemmo/VBZ gede/SC assitannonke/VBA ./PUN

Ilamoommahu/VB 1966/CD M.D./NN anni'yara/PRP$ ilantino/VBD ooso/NNS giddonni/RB ani/PRP leekkite/VB ./PUN

Hatteentenni/RB yihowa/NN shaqqillunninna/NN cincatenni/RB dhukasi/NN afinohanna/NNC umosi/NN heeshshi/NN assanno/VBZ soqqamaanchosi/NN jawaachishinosi/VBA ./PUN

Tini/JJ ikkito/NN qaaqquulleho/NNS tiiu/NN lophonna/NNC fayyimmansa/NN aana/JJ gawajjo/NN abbitanno/VBZ ./PUN

Shoole/CD ooso/NNS noosihu/NN maarki/NN yinanni/VBZ rodii/NN togo/RB yiino/VBD :/PUN sheexaanu/NN maate/NN gawajje/JJC songuwa/NN wolqa/NN ba'anno/VBZ gede/SC assate/VB wo'naalanni/RB no/VB ./PUN

Mittu/JJM manchi/NN saate/VB me/NN "/PUN ete/NN yee/VB xa'mihero/NN mayite/NN qolatto/VB ?/PUN saate/VB kullanniti/JJ babbaxxitino/VBD doogga/NNS no/VB ./PUN

Cee'maaleessu/NN allaalaanchi/NN fafu/NN bayichinni/NN qolanno/VBZ ./PUN

Furra/NNP gashshooti/VBA yannara/SC cimeeyyenna/NNC geerru/NNS songo/NN ofolte/VB ma'litannoha/VB ikkirono/SC ,/PUN furra/NNP geerru/NNS songote/NN ofolle/VB amaalamannokki/VB gede/SC gadadissanni/VB sa'ino/VBD yinanni/VBZ ./PUN

1985/CD M.D./NN kayise/VB rosunnihanna/NNC loosunniha/NN ikke/VB addi/JJ addi/JJ borreessamme/NN loosira/NN hosa/NN hanafinkunni/JJ lamu/JJ tonni/VBZ aieenni/RB kiirsiisi'ranni/VB no/VB ./PUN

Laammiso/NNP lame/CD ooso/NNS ilino/VBD ./PUN