



**DEVELOPING PART-OF-SPEECH TAGGER MODEL FOR
AFAAN OROMO LANGUAGE**

A THESIS PRESENTED

BY

ABEL MEKURIA TEKLA

TO

THE FACULTY OF INFORMATICS

OF

ST. MARY'S UNIVERSITY

**IN PARTIAL FULFILLMENT OF THE REQUIRMENTS FOR
THE DEGREE OF MASTER OF SCIENCE**

IN

COMPUTER SCIENCE

JUNE, 2023

ADDIS ABABA, ETHIOPIA

ACCEPTANCE

**DEVELOPING PART-OF-SPEECH TAGGER MODEL FOR AFAAN
OROMOO LANGUAGE**

BY

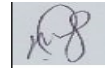
ABEL MEKURIA

**A THESIS ACCEPTED BY THE FACULTY OF INFORMATICS, ST.
MARY'S UNIVERSITY IN PARTIAL FULFILLMENT OF THE
REQUIRMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN
COMPUTER SCIENCE**

Thesis Examination Committee:

Internal Examiner

Mulugeta Adibaru (Assoc.Proff)



05-Jul-2023

External Examiner

Mesfin Abebe (PHD)



10-Jul-2023

Dean, Faculty of Informatics

DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been duly acknowledged.

Full Name of Student

ABEL MEKURIA TEKLA

Signature

Addis Ababa

Ethiopia

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

Full Name of Advisor

ALEMBANTE MULU KUMLIGN (PHD)

Signature

Addis Ababa

Ethiopia June 2023

Acknowledgement

First and foremost, I thank God who helped me through my ups and downs, I became who I'm today it is because of his mercy and protection. I would like to express my deepest gratitude to my thesis advisor Dr. Alembante Mulu for his willingness, enthusiasm to assist in any way throughout the research encouragement, limitless support, assistance, and guidance during the running of this thesis.

I would also like to thank the linguists Mr. Birhanu Tafa and Mr. Lamii Garedo for their valuable support especially on word class identification and tagging the corpus.

Finally, I must express my very profound gratitude to my parents, whose love and guidance are with me in whatever I pursue. They are the ultimate role models. Most importantly, I wish to thank my loving and supportive father and Mother, Mr. Mekuria Tekla and Mrs. Mulunesh Hundie, and my brothers Dr. Achalu Mekuria and Dr. Demmessa Mekuria for their endless love, support and encouragement. This accomplishment would not have been possible without them. Thank you.

List of Acronyms

ANN	Artificial Neural Network
CRF	Conditional Random Field
DT	Decision Tree
HMM	Hidden Markov Model
IE	Information Extraction
IR	Information Retrieval
MBT	Memory Based Tagger
MEMM	Maximum Entropy Markov Model
MT	Machine Translation
NER	Named Entity Recognition
NEWS	North East West South
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
POS	Part-of-Speech
SVM	Support Vector Machine
TEL	Transformation-Based Error-Driven Learning
TnT	Trigram Tagger
WSD	Word sense Disambiguation

Contents

Acknowledgement	iii
List of Acronyms	iv
List of Tables	viii
List of Figures	ix
Abstract	x
CHAPTER ONE	1
1. INTRODUCTION	1
1.1 Background of the Study	1
1.2 Statement of the Problem.....	2
1.3 Motivation	3
1.4 Research Question	4
1.5 Objective	4
1.5.1 General Objective	4
1.5.2 Specific Objective	4
1.6 Significance of the Study	4
1.7 Research Methodology	5
1.7.1 Tag set Preparation and Corpus Collection	5
1.7.2 Review Literature.....	5
1.7.3 Tool Selection	5
1.7.4 Design and implementation of the tagger Model.....	6
1.7.5 Test and Performance Evaluation	6
1.8 Scope and Limitation.....	6
1.9 Organization of the Thesis	6
CHAPTER TWO	8
2 REVIEW OF RELATED LITERATURE.....	8
2.1 Introduction.....	8
2.1.1 Unsupervised Part-Of-Speech Tagging	9
2.1.2 Supervised Part-Of-Speech Tagging.....	9
2.2 Stochastic POS Taggers	10

2.2.1	Word Frequency Approach.....	11
2.2.2	Tag Sequence Probabilities.....	11
2.2.3	Hidden Markov Model.....	11
2.2.4	Conditional Random Field.....	13
2.2.5	Decision Trees	14
2.2.6	Support Vector Machine (SVM).....	15
2.2.7	Neural Network.....	16
2.2.8	Maximum Entropy Markov Model.....	17
2.3	Rule-Based Approach	18
2.4	Hybrid Taggers.....	19
2.5	Related Works	24
2.5.1	Review of Concept Literature.....	24
2.5.2	Review of Related Research Works	24
2.6	Summary	28
CHAPTER THREE		35
3	THE GRAMMAR OF AFAAN OROMO AND TAGSET.....	35
3.1	Introduction.....	35
3.2	Afaan Oromo Writing system.....	35
3.3	Afaan Oromo Morphology	35
3.3.1	Word order in Afaan Oromo.....	36
3.4	Word Categories.....	37
3.4.1	Punctuation Marks in Afaan Oromo	41
3.5	Afaan Oromo Tagsets	42
CHAPTER FOUR.....		46
4	DESIGN OF AFAAN OROMOO PART-OF-SPEECH TAGGER.....	46
4.1	Introduction.....	46
4.2	Corpus Preparation	46
4.3	Corpus Split Methods.....	47
4.4	Tools Used	48
4.5	Designing Hidden Markov Model Tagger.....	49

4.5.1	Lexical Model	50
4.5.2	Contextual Model.....	50
4.6	System Architecture	53
4.7	Designing Rule-Based Tagger	54
4.7.1	Transformation-Based Error-Driven Learning	54
4.7.2	Brill Tagger Architecture	59
4.8	Summary	61
CHAPTER FIVE		62
5	EXPERIMENTS AND PERFORMANCE ANALYSIS.....	62
5.1	Introduction.....	62
5.2	Performance Evaluation Metrics.....	62
5.3	Test Result of HMM Tagger.....	62
5.4	Test Result of Brill Tagger.....	63
5.5	Experimental Analysis	63
5.5.1	Experimental Analysis of HMM Taggers.....	64
1.1.1.	Experimental Analysis of Rule-based Taggers.....	69
5.6	Discussions	74
CHAPTER SIX.....		75
6	CONCLUSION AND RECOMMENDATION	75
6.1	Conclusion	75
6.2	Recommendation.....	76
References.....		i
Appendixes		viii

List of Tables

Table 2-1. Difference between supervised and unsupervised POS tagging	10
Table 2-2. Summary of literature adopted from	20
Table 2-3. Summary of Review of Related Research Works	29
Table 3-1. Afaan Oromo tagsets	43
Table 4-1. Pros and cons of train/test split and cross-validation	48
Table 5-1. Test result of HMM taggers	63
Table 5-2. Test Result of Rule-based taggers	63
Table 5-3. Experimental analysis of HMM unigram tagger	66
Table 5-4. Experimental analysis of HMM bigram tagger	67
Table 5-5. Experimental analysis of HMM trigram tagger.....	68
Table 5-6. Experimental analysis of Rule-based unigram tagger	70
Table 5-7. Experimental analysis of Rule-based bigram tagger	71
Table 5-8. Experimental analysis of Rule-based trigram tagger.....	73

List of Figures

Figure 2-1. Classification of POS tagging adopted from	9
Figure 4-1. HMM tagger training model	51
Figure 4-2. HMM tagging process	52
Figure 4-3. HMM performance analyzer	53
Figure 4-4. Hidden Markov Model architecture	54
Figure 4-5. Rule based tagger training model.....	56
Figure 4-6. Rule based tagger model	60
Figure 4-7. Rule based performance analyzer	61

Abstract

To manipulate, analyze and process human language in a computer, it must be organized and structured in a way it understands. Part-of-Speech (POS) tagging is one of the Natural Language Processing (NLP) applications. It is a task of labeling words with their appropriate Part-of-Speech tags. Different studies have conducted on Part-of-Speech tagging for Afaan Oromo but none of the studies have conducted a comparative study which best suited for Afaan Oromo. In this study, a Part-of-Speech tagger for Afaan Oromo language has been developed using a Hidden Markov Model and rule-based approach. The Viterbi algorithm for Hidden Markov Model and brill transformation-based error-driven learning for the rule-based approach was used with slight modifications in their modules based on the nature of the language. Natural Language Toolkit version 3.4.5 and Python 2.7 were used to implement the tagger model and conduct experimental analysis. Discussion with linguists and review on different works of literature were made to understand the morphological and grammatical structure of the language and to identify possible tagsets for the study. As a result, 27 tagsets were identified. 1196 sentences which are composed of 30, 165 words with 8366 unique words are collected from BBC Afaan Oromo, VOA Afaan Oromo and Afaan Oromo bible. The collected corpus has been split into training and testing corpus. Hence 80% of the corpus is used to train the tagger model and the remaining 20% is to test the performance of the tagger model. Both the Hidden Markov Model and rule-based taggers were trained and tested on the same data. As a result, Hidden Markov Model taggers: unigram, bigram, and trigram taggers achieved an accuracy of 87.3%, 88.4%, and 89.3% respectively and the rule-based taggers which use unigram, bigram, and trigram taggers as initial stage taggers achieved an accuracy of 88.6%, 89.3%, and 89.9% respectively. As shown in the performance analysis result that the rule-based taggers outperform the Hidden Markov Model taggers. To improve the performance of the taggers pre-prepared standard balanced corpus and standard tagsets were recommended for future work.

Keywords: Rule-based taggers, Hidden Markov Model taggers, Afaan Oromo, unigram, bigram, trigram.

CHAPTER ONE

1. INTRODUCTION

1.1 Background of the Study

Natural Language is a means of communication that human beings use to share information distinct from artificial commands or programming language with which one usually talks to a computer. Natural Language is the most important means of communication between human beings; it is also used to preserve cultural achievement and transformation through historical periods to transfer from generation to generation [1].

The semantic representation of knowledge plays a central part in natural language formulation because it connects all components of natural language processing systems. The intuitive understanding of the natural language or the rational reasoning of knowledge bases, the generation of natural language expressions from formal representations [1].

Language experts group words of a language into word classes (sets) based on the syntactic or grammatical behavior of words. These word classes are called syntactic or grammatical categories, but more commonly by the traditional name parts-of-speech (POS) [2].

During the last few decades, the overflow of digitized information has been growing tremendously. This tendency will continue with the globalization of information societies and the growing importance of national and international computer networks. Digitized data is one reason why the theoretical understanding and the automated treatment of communication processes based on natural language have such a decisive social and economic impact [1].

Natural language processing (NLP) is a field of study that deals with computing natural language with a machine and provides a potential means of gaining access to the information inherent in a large amount of text available through intranet and the Internet [3].

NLP is a way for computers to analyze, understand, and derive meaning from human language in a smart and useful way. Developers can organize and structure knowledge to perform automatic text summarization, named entity recognition, word sense disambiguation, grammar checker, sentiment analysis, part-of-speech tagging, relationship extraction, question answering, Topic extraction, Stemming, and more. NLP is commonly

used for text mining, part-of-speech tagging, machine translation, and automated question answering [4].

One area of NLP is Part-of-Speech (POS) tagging which is the task of labeling or tagging each word in a sentence with its appropriate word classes to get the correct sense of the word from in the context [5].

POS tagging is a precondition or subcomponent for most natural language processing tasks rather than being an application to stand by itself. It is used in machine translation, syntactic parsing, information extraction, named entity recognition, information retrieval, spell checker, and stemmer [5].

POS tagger approaches are built based on the models to train data sets using different algorithms and apply them to unseen language instances. The corpus-based methods that are used to tag the text in a sentence include Hidden Markov Model (HMM), Artificial Neural Networks (ANN), Memory Based Taggers (MBT), and Decision tree (DT) taggers [5].

Hidden Markov Models (HMM) have been the mainstay of the statistical modeling used in modern speech recognition systems. Despite their limitations, variants of HMM models are still the most widely used technique in that domain and are generally regarded as the most successful [2].

The Viterbi algorithm only involves multiplications working with logarithms and choosing the largest element. This not only solves the problem with floating-point underflow, but it also speeds up the computation since additions are much quicker than multiplications [2].

1.2 Statement of the Problem

The desire for this study comes from that many NLP applications require Part-Of-Speech tagging as their basic component. Lexical attributes, like syntactic and semantic attributes are in most cases ambiguous in every language, particularly in Afaan Oromo. This ambiguity of the languages can be solved using automated NLP applications like POS tagger and word sense disambiguation. Currently there is natural language processing system developed to process Part of Speech tagging for Afaan Oromo which are conducted using Hidden Markov Model approaches. In this comparative study the Afaan Oromo text is tagged by using both HMM approach and Rule based approach and finally the two models are compared.

Afaan Oromo is one of the major languages that is widely spoken and used in Ethiopia. Currently it is an official language of Oromia state. It is used by Oromo people, who are the largest ethnic group in Ethiopia, which amounts to 34.5% of the total population according to the 2008 census. With regard to the writing system, since 1991 Qubee (Latin-based alphabet) has been adopted and become the official script of Afaan Oromo. Currently, Afaan Oromo is widely used as both written and spoken language in Ethiopia. Besides being an official working language of Oromia State, Afaan Oromo is the instructional medium for primary and junior secondary schools throughout the region and its administrative zones. It is also given as the department in five universities in Ethiopia. Thus, the language has well established and standardized writing and spoken system.

1.3 Motivation

Part of speech tagging is a crucial task in natural language processing that involves assigning a grammatical category to each word in a sentence. By accurately identifying the parts of speech in a sentence, we can better understand the meaning and context of the words being used. Part of speech tagging has been extensively studied for many languages, but there has been relatively little research on part of speech tagging for Afaan Oromo language. Afaan Oromo is one of the languages with the largest number of speakers in Africa after Arabic and Hausa. Despite its importance, there is a lack of computational resources and tools for Afaan Oromo language processing. Therefore, developing an accurate and efficient part of speech tagger for Afaan Oromo language can significantly improve the quality of various NLP applications such as machine translation, sentiment analysis, and information retrieval.

Therefore, this research can contribute to the development of more effective natural language processing tools, which can ultimately benefit both individuals and organizations by improving communication, increasing efficiency, and reducing errors. Moreover, it can contribute to the preservation and promotion of the language and culture of the Oromo people.

1.4 Research Question

This study will try to answer the following questions:-

- How can we develop an accurate and efficient part of speech tagger for Afaan Oromo language?
- What are the most effective methods and techniques for improving the performance of Afaan Oromo part of speech tagger?
- What is the accuracy and effectiveness of different part of speech tagging techniques for Afaan Oromo language?
- How Afaan Oromo POS tagger can be improved to better support other natural language processing tasks?

1.5 Objective

1.5.1 General Objective

The main goal of this paper is to develop Part of speech tagger system using HMM and rule-based approach and make a comparison between the approaches.

1.5.2 Specific Objective

The specific objectives of this study are

- To identify and review word categories, morphological properties and sentence structures to derive the POS tags and the tag set for the Afaan Oromo language.
- To study the type of word classes and the type of lexicon used in Afaan Oromo POS tagging and design suitable word classes.
- To prepare POS tag corpus for Afaan Oromo Language and design the appropriate lexicon
- To select the best performing algorithm for both rule-based and Hidden Markov Model
- To design and develop POS tagger models for Afaan Oromo language.
- To evaluate the performance of the models and make comparison.

1.6 Significance of the Study

The major importance of POS tagger for Afaan Oromo language is that it serves as a milestone for further NLP researches in the language. This is because POS tagger is the

necessary requirement for advanced NLP areas such as machine translation, spell checker, text summarization, information extraction, and information retrieval. Additionally, Linguists and students in the higher education can also apply the output of the study to identify word classes in a sentence automatically for language education.

The other significance of this study is that, it can help peoples who are new and want to learn Afaan Oromo language so that they can easily and quickly identify the word classes, how words are used in a sentence, and their morphological structure.

Additionally, the development of POS tagging plays a vital role in increasing the computational usability of Afaan Oromo language. This is due to the fact that POS tagger would minimize the workload of researchers who are using it as a fundamental Part of any other NLP application and motivate researchers to conduct further research on Afaan Oromo language.

1.7 Research Methodology

To develop a POS tagger for the Afaan Oromo language different methods are used in this study. Tag set preparation and corpus collection, Review Literature, tool selection, design and implementation of the tagger model and finally testing and performance evaluation are used.

1.7.1 Tag set Preparation and Corpus Collection

The word class's identification and tag set preparation are conducted by discussing with linguists and experts about the grammatical and morphological structure of words in a sentence.

The corpus that is used in this study is collected from BBC Afaan Oromo, VOA Afaan Oromo and Afaan Oromo bible.

1.7.2 Review Literature

The literature focuses on the grammatical structure of the Afaan Oromo language and the works of literature conducted on POS tagging for other languages are reviewed.

1.7.3 Tool Selection

Different software and designing tools are selected using the most-fit strategy using the parametric analysis to conduct the study. Python 2.7, which is open-source and user-friendly software, and NLTK is an open-source tool that contains open-source Python

modules, linguistic data, and documentation for research and development in natural language is used to design and implement the automatic POS tagger model.

1.7.4 Design and implementation of the tagger Model

The tagger model has been designed by using both rule-based and Hidden Markov approaches. Rule-based tagger implementation involves large databases of automatically driven rules with a transformation-based error-driven Learning approach. It uses Unigram, Bigram, and trigram taggers as initial stage taggers to tag the words. The HMM implementation is based on the Viterbi algorithm. It uses Unigram, Bigram, and trigram taggers to tag the words.

1.7.5 Test and Performance Evaluation

The sample data collected for the proposed research study is used for training and testing the tagger model. From the collected data, 80% is used to train the tagger (including training and validation datasets) and 20% is used to test the POS tagger model's performance.

1.8 Scope and Limitation

The word class's identification and tag set preparation are conducted by discussing with linguists and experts about the grammatical and morphological structure of words in a sentence.

The corpus that is used in this study is collected from BBC Afaan Oromo, VOA Afaan Oromo and Afaan Oromo bible.

The study was conducted by using the HMM approach implemented with the Viterbi algorithm and the Rule-based approach with the Brill Transformation-Based Error-driven Learning approach.

The study was conducted on the word classes or tag sets that are categorized by the researcher with the help of a linguistic expert.

There may be some possible limitations in this study. The first is the Unavailability of readily available corpus, readily available word classes and shortage of linguistics in the area of Afaan Oromo language are the challenges faced during the study.

1.9 Organization of the Thesis

The thesis is organized into six chapters. The second chapter discusses the methods used for POS tagging learned from the comprehensive literature review and the related studies

conducted on POS tagging using HMM and Rule-based approaches. The third chapter focuses on Afaan Oromo's sentence structure, word classes, and tag set preparation. The fourth chapter focuses on corpus preparation, tool selection, and design of the HMM model and Rule-based model. The fifth chapter discusses the implementation of tagger models and experimental analysis of the taggers. The last chapter focuses on the conclusion of the study, discussions about the developed model, and recommendations for future works.

CHAPTER TWO

2 REVIEW OF RELATED LITERATURE

2.1 Introduction

Part of Speech (POS) tagging is the undertaking of naming or labeling each word in a sentence with its proper word classes to get the word's right feeling from the context. POS labeling is a precondition or subcomponent for most normal language processing such as Word Sense Disambiguation (WSD), Machine Translation (MT), Information Retrieval (IR), Named Entity Recognition (NER), Information Extraction (IE), Spell Checker, and Stemmer instead of being an application to remain without anyone else [5].

POS tagging is a classification task [6]. The most common problems in POS tagging are lexical ambiguity and unknown words [7]. The lexical ambiguity of a word can appear with more than one tag. It can be solved based on the context rather than emphasizing a single word.

The POS labeling task models compute training data sets using different algorithms to identify the words with their given tag and later use these computational results on untrained datasets to label the words with their suitable word classes [5]. Tagging approaches rely on the pre-tagged corpus and untagged corpus for training based on the algorithm they use.

Part of speech (POS) labeling is basic in natural language processing. Numerous statistical POS taggers use text data that are manually annotated with POS labels as training data to acquire factual data or rules to perform POS labeling [8]. In any case, in lexical category labeling, we frequently experience words that don't exist in training data. Such obscure words are normally taken care of by an exceptional process due to the factual data or rules for those words that are obscure [3].

The base of POS labeling is that being most words ambiguous concerning their POS, they can be disambiguated considering an adequate text from the training corpus.

Corpus-based techniques mostly produce more outstanding results while compared to techniques that do not use training data. The languages' complexity often challenges these techniques and these complexities are observed indirectly through the result they produce [9].

On the other hand, POS tagging systems collect statistics from existing corpora, either tagged (supervised training) or untagged (unsupervised training). Part-of-Speech tagger system is software that is developed for specialized purposes that can manipulate the tagging task. It can understand a text in some language and assign a suitable word class based on the information provided to it. Labeling word classes can be categorized into three categories: Rule-based tagging, Statistical tagging, and hybrid tagging [10].

2.1.1 Unsupervised Part-Of-Speech Tagging

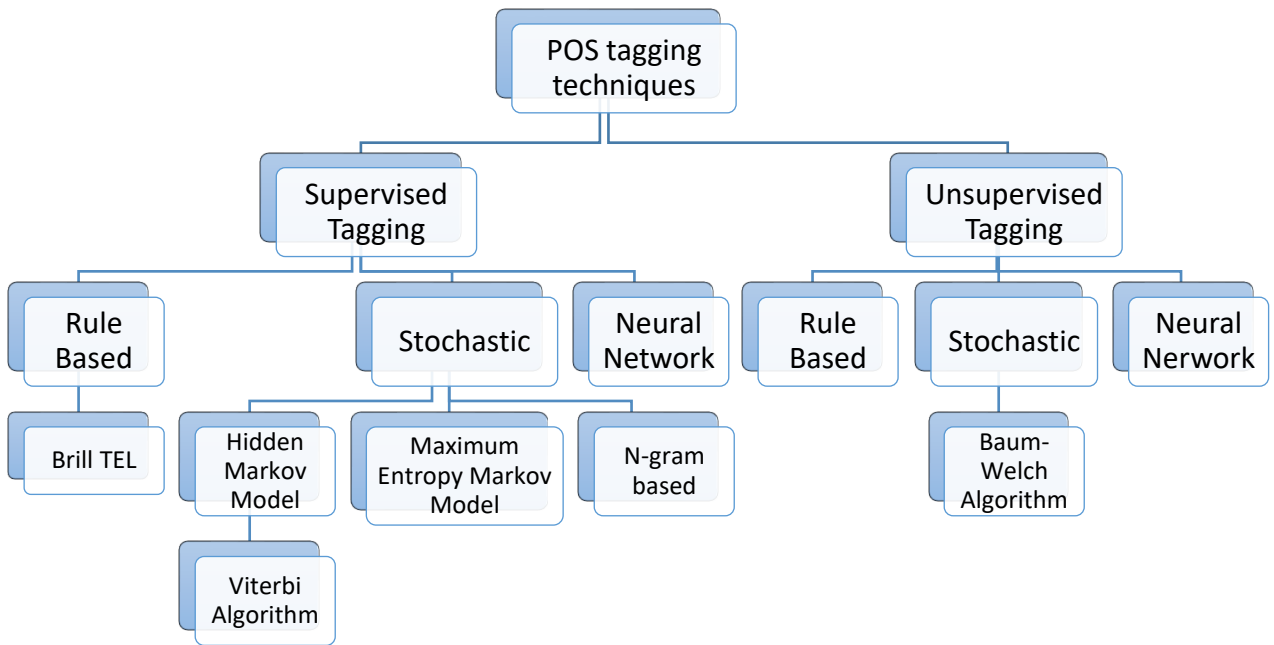


Figure 2-1. Classification of POS tagging adopted from [80]

Unsupervised Part-Of-Speech labeling is characterized as, it does not require a pre-labeled corpus; these models, instead utilize computational systems that are complex and create tag sets automatically for each text [11]. These programmed label sets are used to either evaluate the probabilistic data required by stochastic techniques or to develop explicit rules required by the rule-based systems.

2.1.2 Supervised Part-Of-Speech Tagging

Supervised Part-Of-Speech labeling is characterized as a pre-labeled corpus is created and it is depended on by supervised taggers to work as an establishment for making tools that can be used in the complete labeling process [11]. It can be represented as the tagger word reference since it has the quantity of word use (frequencies) and a tag, just as the

probabilities of tag sequences and a set of rules. The differences between supervised and unsupervised POS tagging are listed in table 1 that is adopted from [11].

Table 2-1. Difference between supervised and unsupervised POS tagging

Supervised	Unsupervised
Trains from a pre-tagged corpus.	A pre-tagged corpus is not required for training.
The algorithm is dependent on the pre-tagged corpus.	The algorithm is independent of the pre-tagged corpus, but it creates its pattern features on its own.
The accuracy of the tagger depends on the pre-tagged corpus.	The accuracy of the tagger depends on the algorithm.
Mostly need manual intervention.	Less accurate when compared to supervised learning.
Greater accuracy.	Disambiguation using statistical, rule-based, or hybrid approaches.
Training and final output (testing) take place at different times.	Training and the final output (testing) occur in real-time.

2.2 Stochastic POS Taggers

Most current POS taggers are stochastic. The stochastic model uses frequency, probability, or statistics to assign the most appropriate tag for a given word by calculating the most suitable tag based on the word and its preceding tag [12]. Since word tag is not only dependent on its tag but also the previous tag. It uses the most frequently used tag for specific words in the training corpus and uses this information in the untagged or testing corpus [10].

Statistical tagging algorithms rely on finding the most probable tag sequence for a sentence instead of individually disambiguating the morphological class of each token [13], [14], [6]. Statistical tagging models trained on the specific corpus or trained on a small corpus usually perform worse on the words from a different domain.

For languages with a lack of annotated corpora, it is difficult to disambiguate [13]. For those languages to create an annotated corpus, a repetitive approach can be used. Initially, a small set of the corpus is disambiguated manually, and the tagger is trained on this set, then another set of the corpus is tagged automatically and corrected manually [13]. This

could yield a new, bigger training corpus and this process is repeated until the tagger is trained with enough words to tag accurately to some extent. Maximum tagger accuracy minimizes the human time spent to annotate the results and fewer annotation errors in the result [15].

This approach's main drawback is that it tags words in the sentence without considering the language's grammatical rules [15]. The Stochastic method uses the following techniques to POS tagging [16], [17]:

2.2.1 Word Frequency Approach

This approach checks the training corpus for a particular word with its label. If the word was labeled with that specific label frequently, then that word's ambiguous instance will be labeled with that specific tag. Sometimes this approach can label the word with a valid or an invalid label.

2.2.2 Tag Sequence Probabilities

In this technique, the tagger computes the probability of a given sequence of tags occurring. In this technique, the best tag for a given word is defined by the probability at which it occurs with the n previous tags. N-gram is a kind of this technique. The Hidden Markov Model (HMM) combines both tag sequence probabilities and word frequency measurements.

The statistical model is based on many models such as - Hidden Markov Model (HMM), the Maximum Entropy Markov Model, Decision Trees, Neural Network, Support Vector Machine, and Conditional Random Field [18].

2.2.3 Hidden Markov Model

HMMs have become the most popular model because of their robustness and accuracy which is a generative model and it gives an output straightforwardly by modeling transition matrix depending on the training data [19]. The outcomes can be improved by providing more data points; however, there is no immediate authority over the output labels. HMM learns the change probabilities on its own, depending on the given training data [3]. Thus, if we give more data points, at that point, we can improve the model to incorporate a more extensive assortment.

Hidden Markov Model-based taggers incredibly beat other tagger algorithms in training time like maximum entropy and conditional random fields, setting aside relatively longer time to train [13].

A Hidden Markov Model (HMM) is a powerful statistical tool for displaying generative sequences that can be portrayed by a fundamental process of creating an observable sequence [20]. An HMM can be utilized to explain classification issues that have an inherent state sequence portrayal. The model can be imagined as an interlocking arrangement of states. These states are associated with a set of transition probabilities, which demonstrate the likelihood of change between two given states [13]. A process starts in some state, at that point at discrete time intervals, the process moves to a new state as directed by the transition probabilities.

In an HMM, the precise sequence of states that the cycle creates is obscure (hidden). As the process enters each state, one of a lot of output symbols is discharged by the process. Precisely which symbol is discharged is controlled by a likelihood dispersion that is specific to each state. The output of the HMM is a sequence of output symbols [21].

In HMMs, all calculation relies upon the two probabilities $P(\text{Tag} | \text{Tag})$, and $P(\text{Word} | \text{Tag})$, if we have to incorporate some source of information into the tagging process, we have to figure out how to encode the information into one of these two [22].

In Markov Model-based taggers, the lexicon consists of probabilities relative to some degrees of values other than the expectation still proper form $P(\text{Word} | \text{Tag})$. In contrast, in transformation-based taggers, the lexicon is simply a sequence of tags assigned for a word in the training corpus with a single tag named as most probable [9].

There are five components expected to characterize an HMM [21] [23], [22]:

The number of different states denoted by S , in part-of-speech tagging systems, the states are tags that are hidden. The system uses each state to relate a single state of the Hidden Markov Model. Each state and states can be defined as $S = \{S_1, S_2, \dots, S_m\}$ and the state at time t as q_t respectively.

The number of different observation symbols denoted by O , in a part-of-speech tagging system, the observation symbols are the set of words in the system's dictionary. Each symbol can be represented as $O = \{O_1, O_2, \dots, O_m\}$.

The state transition probability distribution $M = m_{ij}$. The probability m_{ij} is the change of state from one state to the other state in a single transition. In POS tagging, as states represent the tags, the state transition probability is the probability that the model will change from tag t_i to tag t_j . The probability can be calculated using the information from the training corpus.

The observation symbol probability distribution $B = \{b_j(k)\}$. The probability $b_j(k)$ shows the k th output symbol's production when the model is in state j . In part-of-speech tagging, the observation symbol probability is the probability that the word will be generated when the model is at a tag t_j . The probability can be calculated using the information from the training corpus.

The initial state distribution $\pi = \{\pi_i\}$, is the probability that marks the beginning of the model in state i . In POS tagging it is the probability that the sentence will begin with a tag t_i .

When utilizing an HMM to perform part-of-speech tagging, the objective is to decide the most likely sequence of tags that produces the words in the sequence of output symbols (sentence) [21]. As such, given a sentence S , calculate the sequence T of labels that maximize $P(S|T)$. The Viterbi algorithm is a widely used method for calculating the most probable label sequence when utilizing an HMM [21], [23].

2.2.4 Conditional Random Field

CRF is a probabilistic framework for segmenting and labeling sequence of data which has the advantages of both HMM and MEMM and solves label bias problems in a principled way [24], [25], [26].

CRF is an undirected graph model in which tag sequences for specific observation sequences are defined by a single exponential distribution [24].

It is a discriminative probabilistic classifier that models conditional probability distributions $P(Y | X)$ of label sequences of given input sequences rather than a joint distribution over both label and observation sequences in which X is always known or observed and Y is a label of sequences [26], [24].

In CRF non-independent, the capability to use rich features of the input is combined with the graphical model's flexibility [27]. The CRF is trained using features over labeled and unlabeled data. These features describe the different sides of data and may provide a

flexible way of describing the task. A CRF, when equipped with useful features, gives accuracy much better than other models [28].

CRF combines the best of generative and classification models [27]. As generative models, they balance decisions at different levels to obtain globally acceptable labeling, whereas as classification models they train the models independently classifying each model with a distinction and have many statistically dependent input features [27], [24].

When applying CRFs to the part-of-speech tagging problem, an observation sequence and state sequence probabilities are defined. An observation sequence is a word in the corpus and the state sequence is its corresponding label sequence [29].

2.2.5 Decision Trees

Decision trees inspect contextual information to label words with POS tags or guess particular POS tags [5]. Decision trees are n-ary branching trees built top-down and may have zero or more (non-terminal) nodes and one or more leaf (terminal) nodes.

Decision tree induction is based on a greedy algorithm that selects test features to build decision trees in the top-down recursive divide-and-conquer method [30], [31].

A decision tree has three nodes: the root node, internal node, and leaf node [5]. The root node has zero or more outgoing edges and doesn't have an incoming edge. Internal nodes have two or more outgoing edges and have a single incoming edge. Leaf nodes have no outgoing edges and have a single incoming edge. In POS tagging the internal nodes represent tests and leaf nodes represent conditional probability distribution [30].

In a decision tree, the tree's first growing leaves are pruned to fit with the other leaves in the tree. Pruning is reducing the growth of apical leaves to increase the growth of the lowest leaves to fit it into the training data. When classifiers make decisions on random attributes of training data it can cause over-fitting, this will lead to errors on newly arriving data [31], [32].

Whenever the tree-based model has been ready we decide how to use it for tagging sequences of ambiguous words. The most common way is to let the trees decide the particular part-of-speech tag for each in a single pass, for instance from left to right, throughout the sequence [33].

When an unknown word appears in the corpus, its POS tag will be guessed by traversing decision trees for unknown words by identifying contextual features, the suffix, and capitalization of the word then tags with one of the open-class words [34].

The use of a decision tree for POS tagging is two-fold; the first is that it minimizes the costs spent to develop classification rules as it learns the rule directly from large annotated corpora and the second is, the rules generated by the system are easy to understand and are simple to reconstruct [5].

2.2.6 Support Vector Machine (SVM)

Vapnik developed an SVM model in the year 1995, which is a very suitable object for theoretical analysis, and it unifies the Structural Risk Minimization model, Data Compression model, a universal model for constructing complex features, and a model of real-life data. The unification of models reflects the essential properties of the learning mechanism [35].

SVM is a supervised machine learning model based on statistical learning theory which uses a classification algorithm for a two-class classification problem. Hence it is called a binary classifier and used for classification and regression [36]. SVMs are mostly known for their good performance on generalization and are also used for pattern recognition.

The SVM maps the input space into some high-dimensional feature space and constructs a linear decision surface with unique properties that ensure the network's maximum generalization ability by separating one class of data points from another class. The linear decision surface is known as a hyper-plane [35]. The support vectors are points that are nearest to separating hyper-planes.

While comparing SVM with other statistical learning algorithms, SVM has high generalization performance due to its independence of computational complexity and dimensionality of the space where the input feature is mapped [37]. Other statistical learning algorithms such as Hidden Markov Model, maximum entropy, and decision tree select features and optimize through training to avoid over-fitting [38].

SVM learning is conducted with a combination of selected features with optimal computational complexity where other algorithms fail to manipulate these combinations efficiently [38].

2.2.7 Neural Network

Artificial Neural Networks are Computer programs that are intended to simulate the natural human cerebrum data preparation. A neural system's intensity is controlled by its segment preparing components (neurons) that have weighted input, transfer function, and output [39]. The conduct of a neural system, then again, is dictated by the transfer function of its component neurons, the learning rule, and the design of the network itself. The actuation signs of the neurons in the system are the weighted entirety of inputs, which went through the transfer function to deliver output for every neuron [40].

Artificial neural networks comprise an enormous number of precise handling units. These units are profoundly interconnected by coordinated weighted connections [41]. Related to every unit is actuation esteem through the associations, this initiation is spread to different units.

ANNs learn from the fact of examples and connections got in the input data and not from programming. Henceforth, it is conceivable to call the ANN learning process versatile learning since it can discover properties from the introduced input data. The system doesn't require to be advised on how to respond to every data input independently like regular programming [8].

Artificial neural systems have their assortments, including the input layer's neural systems, hidden layer, and output layer, and ANNs also incorporate unique profound neural systems having more than one hidden layer [41]. A deep neural network is connected to learning various degrees of depiction and abstractions that help comprehend information, such as pictures, sound, and text.

Neural Network (NN) system taggers have properties that give advantages over HMM taggers, in NNs uncertain labeling can be taken care of effectively without extra calculation and training can be made with modest quantities of data even if they are suitable for dialects with lesser training data [41].

To use artificial neural systems for Parts-of-speech labeling, it needs to preprocess the input data. The preprocessing may incorporate tokenizing, feature generation, feature extraction, and other actions to prepare the corpus for tagging using NN systems by such preprocessing errands, the data is set up in an arrangement required by the neural network and afterward given to it as input [42].

2.2.8 *Maximum Entropy Markov Model*

A Maximum Entropy Markov Model is a method for automatic training that combines some of the useful properties of both transformation-based tagging and the Hidden Markov Model [43]. The maximum entropy model is an obsolete name for logistic regression.

Maximum Entropy is a versatile and adaptable modeling system that determines the probabilities depending on constraints [6]. Upon the utilization of constraints, the most likely sequence of labels is produced. These constraints are resolved from the training data, keeping up the association between the history and likelihood results. The results are the arrangements of permissible labels.

It is a linear classification technique [44]. In its essential manifestation, linear classification joins, by expansion, the pre-decided weights utilized for representing the significance of each element to a given class [44].

The ME model produces a grouped model for specialized features [43]. The expectation is to boost the entropy of the model. An attempt is made to lessen the measure of data that the model does. Each time we add a feature we need to do a lot of molding, which gets increasingly hard as we have an ever-increasing number of such features. The MEMM based part-of-speech tagger conditions on observing a word itself, neighboring words, previous tags, and different combinations, utilizing feature templates [22].

Training a Maximum Entropy classifier includes fitting loads of each feature value for a specific class to the available training data. The best match of the weights to the data is found by choosing weights to maximize the learned classification model's probability [44]. The maximum entropy approach to deal with POS labeling is one group of AI or corpus-based techniques to deal with classification in which numerous features are computed for the word to be labeled, and all the features are consolidated in a model dependent on multinomial logistic regression [22]. The posterior $P(T | W)$ is computed straightforwardly, training it to discriminate among the suitable label sequences. The MEMM computes each state's posterior adapted on the previous state and current observation [22].

The ME tagger system derives a probability model consistent with the training data whose primary goal is to find a model with maximum entropy that meets the preconditions provided from the training corpus [45].

This ME framework's main advantage is that it causes or empowers us to represent to issue explicit knowledge as features [46].

2.3 Rule-Based Approach

The Rule-based approach is one of the earliest POS tagging systems used to tag words with their word classes or lexical categories [7]. The rule-based POS taggers are highly dependent on the linguistic feature of specific languages such as morphological, dictionary, or lexicon and syntactical information to assign possible tags to each word [8].

The earliest rule-based POS tagging systems are systems where a set of hand-written rules constructed by the researcher with the help of linguists or machine language were applied to the text and for the words having ambiguous meaning, contextual information was used to assign POS tags to the words [25], [7].

The earliest rule-based system rules are usually difficult to construct and are not very robust [47]. The rules applied are often known as context-free rules. Disambiguation is also done by analyzing linguistic features of the word along with preceding and the following word. In the earliest algorithms of rule-based systems, two-stage architectures were used for automatically assigning POS tags [47], [48], [22].

The initial stage uses a dictionary to tag each word by assigning its most appropriate tag, estimated by examining the large tagged corpus without concern to the context. It was usually done by using the Unigram tagging model.

The initial tagger has two methods to improve the performance of the tagger: in the first method the words that were not in the training corpus would be capitalized and tagged to proper nouns and the second method is attempting to tag words that are not found in the training corpus would be assigned the tag most commonly tagged for the words ending with the same three ending letters.

In the second stage, it uses a large set of manually-constructed disambiguation rules to minimize or that would result in the greatest error reduction to a single part of speech for each word. A rule-based POS tagger works by automatically recognizing and correcting its defects step by step to improve its performance [47].

In the recent Rule-based approach, the rules are generated using transformation-based error-driven learning. Brill presented TEL in the mid-90s as another way to deal with corpus-based natural language learning [9]. The learning algorithm is an error-driven

greedy methodology that creates a set of rules. It works iteratively by including a show step the standard that best fixes the current errors. Rules are obtained by instantiation of a predefined set of template rules.

The computational time of TEL is high when compared to simple Rule-based algorithms [41]. This is due to its nature of creating and evaluating rule templates while dealing with a large corpus and large set of rule templates it gets difficult to manage [41]. This problem can be solved by restricting the number of rule templates that are reserved for the wrong tagging [41].

The framework named randomized TEL which is created by [49], depends on this thought: at every cycle, it analyzes each mistakenly labeled word, yet just some predefined constant of all conceivable rule templates that would address the tag are randomly chosen. By using this method, the training time gets free of the number of rules [41].

Transformation based learning for POS tagger system works in the following manner [9]:

1. According to the training corpus, the tagger system labels the most probable tags for each word in the corpus.
2. The tagger system learns a set of transformation rules and applies them in an annotated corpus. These transformation rules can change the label of the word according to the condition of learned transformation rules. The word's label can be changed either the word is unknown (not found in the training corpus) or the word has been found tagged with that specific label once in the training data.
3. After all possible transformations or rules applied to the corpus, the system evaluates and picks the transformations with fewer errors.
4. Finally, the learning halts when there is no transformation with fewer error results are found.

2.4 Hybrid Taggers

Hybrid taggers are a combination of rule-based taggers with stochastic taggers used to assign the best tag for each input text's words. Hybrid taggers combine both taggers' best features and produce a better output of tagged text than each tagger produces individually.

The advantages and disadvantages of models are summarized in the table 2-2 below.

Table 2-2. Summary of literature adopted from [50] [8] [51]

No	Approach	Pros	Cons
1	Stochastic Method	<ul style="list-style-type: none"> • Requires minimal specialists or expertise effort • Can be created for any language pair with enough training data • Can prototype a new system quickly at a very low cost • Resolves linguistics uncertainty problems by a solid mathematical basis. • Extract knowledge from corpus • Coverage depends on the training data 	<ul style="list-style-type: none"> • Sometimes produce an unacceptable sequence of labels according to the grammatical rules of the language. • In supervised statistical learning, some changes may result in the re-annotation of the whole training corpus.
2	Rule-based Method (TEL)	<ul style="list-style-type: none"> • Less stored information. • The system is cost-efficient and accurate in terms of its result • Do not require a high level of linguistic knowledge • A small set of simple rules are used. • Ease of finding and fixing errors of a tagger • Reduce the risk in terms of system accuracy. 	<ul style="list-style-type: none"> • It does not improve with data its answer is always fixed. • It fails when the text is unknown.

		<ul style="list-style-type: none"> • The output that the system has generated is dependent on, so the output responses are stable. 	
3	Hidden Markov Model (HMM)	<ul style="list-style-type: none"> • Strong statistical foundation. • Efficient learning algorithms. • Can handle inputs of variable length most flexible generation of sequence profiles. • Can be combined into libraries • Can be applied to a wide variety of NLP applications. • Allow consistent treatment of insertion and deletion penalties in the form of locally learnable. • Easy and reliable and uses a memory-efficient algorithm. 	<ul style="list-style-type: none"> • A large number of unstructured parameters. • They cannot express dependencies between hidden states.
4	Maximum Entropy Markov Model (MEMM)	<ul style="list-style-type: none"> • Increased freedom in choosing features to represent observations. • It can also naturally solve the problem of parameter smoothing in statistical models. 	<ul style="list-style-type: none"> • suffer from the "label bias problem," • Due to the slow convergence speed of the algorithm, the maximum entropy statistical model has a large computational cost and large space-time overhead;

5	Conditional Random Field (CRF)	<ul style="list-style-type: none"> • It is possible to reach high-quality labeling if you choose the right features • CRF is flexible enough in terms of feature selection. In addition, features don't need to be conditionally independent 	<ul style="list-style-type: none"> • CRF is highly computationally complex at the training stage of the algorithm. It makes it very difficult to re-train the model when newer data becomes available.
6	Decision Tree	<ul style="list-style-type: none"> • Compared to other algorithms decision trees requires less effort for data preparation during pre-processing. • A decision tree does not require the normalization of data. • A decision tree does not require the scaling of data as well. • Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent. • A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders. 	<ul style="list-style-type: none"> • A small change in the data can cause a large change in the structure of the decision tree causing instability. • For a Decision tree sometimes calculation can go far more complex compared to other algorithms. • Decision tree often involves higher time to train the model. • Decision tree training is relatively expensive as the complexity and time have taken are more. • The Decision Tree algorithm is inadequate for applying regression and predicting continuous values.
7	Support Vector Machine (SVM)	<ul style="list-style-type: none"> • SVM works relatively well when there is a clear margin of separation between classes. 	<ul style="list-style-type: none"> • SVM algorithm is not suitable for large data sets.

		<ul style="list-style-type: none"> • SVM is more effective in high-dimensional spaces. • SVM is effective in cases where the number of dimensions is greater than the number of samples. • SVM is relatively memory efficient 	<ul style="list-style-type: none"> • SVM does not perform very well when the data set has more noise i.e. target classes are overlapping. • In cases where the number of features for each data point exceeds the number of training data samples, the SVM will underperform. • As the support vector classifier works by putting data points, above and below the classifying hyper-plane there is no probabilistic explanation for the classification.
8	Artificial Neural Networks (ANN)	<ul style="list-style-type: none"> • Storing information on the entire network • The ability to work with inadequate knowledge • It has a high fault tolerance • Having a distributed memory • Gradual corruption • Parallel processing ability 	<ul style="list-style-type: none"> • Hardware dependence • Unexplained functioning of the network • Assurance of proper network structure • The difficulty of showing the problem to the network • The duration of the network is unknown

2.5 Related Works

2.5.1 Review of Concept Literature

To Support and achieve this study's aim, several previously done related research literature and Salient concepts (books, articles, and journals) are reviewed. To understand the research matter, the researcher reviewed research and journals conducted on POS tagging for Afaan Oromo, other Ethiopian languages and languages of other countries. The researcher systematically reviewed related material thus to understand and analyze the scientific concepts related to this research and critically relate, and evaluate the studies to get an idea of:

- What is the objective of the study?
- What are the methodologies used?
- What are the significant findings?

2.5.2 Review of Related Research Works

The related research and review papers are listed according to the locality. Review results of research conducted in Ethiopian languages are prioritized first, reviewed documents, journals, articles of other countries are prioritized second and the literature reviewed on Afaan Oromo language are prioritized last.

The study conducted by Mequanent Argaw focuses on developing the Amharic part of speech tagger using neural word embedding as a feature. Neural word embedding is a vector representation of words that capture syntactic and semantic information about words. Long Short-Term Memory recurrent neural networks and their bidirectional versions are used to develop tagging models based on deep learning algorithms. Evaluating the performance of the model bidirectional Long Short-Term memory obtained 93.67% result [52].

The study conducted by Gebeyehu Kebede focuses on analyzing and assessing decision trees' application for POS tagging for the Amharic language. The tagger was developed using the J48 decision tree classifier, the Weka implementation of the C4.5 algorithm. The data was comprised of 1065 sentences from the NEWS, including 210,000 words. According to the performance test that is conducted by 10-fold cross-validation the result shows that the tagger is 84.9% accurate. Based on the test result he concluded that only the

frequency of the target word with its neighboring context is not sufficient for POS tagging [5].

The study conducted by Teklay Gebregziabher focuses on developing a POS tagger model for the Tigrigna language using a hybrid approach (HMM and Rule-based approach). The study was conducted on 1000 sentences which are composed of 26,000 words and 36 tag sets are prepared and used to tag the words. According to the study, different experiments are conducted for HMM and Rule-based independently. The result shows that HMM tagger performance is 89.13% and Rule-based tagger performance is 91.8%. By combining the two approaches the accuracy is improved to 95.88% [8].

The study conducted by Getachew Mamo focuses on developing a POS tagger prototype for the Afaan Oromo language. The tagger prototype was developed based on the HMM approach with unigram and bigram models implementing the Viterbi algorithm. Java programming language is used to develop the prototype. The performance of the prototype was tested using 10-fold cross-validation. The study used 159 sentences which are composed of 1621 words and 17 tag sets are prepared and used to tag words. According to the test, both Unigram and Bigram models yield 87.58% and 91.97% accuracy [3].

The study conducted by Sisay Fissaha Adafre tries to develop word segmenting and POS tagger for Amharic using Conditional Random Fields. According to the study morphological and lexical features significantly improve the result of POS tagging. The study was conducted with 1000 manually annotated corpus and a large number of corpus for testing. The developed system performs with an accuracy of 84% in word segmenting and 74% in POS tagging for Amharic [53].

The study conducted by Getachew Emiru focuses on developing a POS tagger for Afaan Oromo using a hybrid approach. The tagger model is developed by combining an HMM model and a Transformation or Rule-based approach. For the Hidden Markov Model, a Viterbi algorithm identifies a tag sequence for a given word sequence. In a rule-based model transformation-based error-driven learning approach is used to learn rules. The tagger was implemented using a Natural Language Toolkit (NLTK) and a python programming language. To conduct the learning and testing 1517 sentences are used and HMM, Rule-based, and hybrid tagger were tested with the same training and testing data and achieved an accuracy of 91.9%, 96.4%, and 98.3% respectively [54].

The study conducted by Berhanu Herano tries to develop a part-of-speech tagger for Wolayta Language. The tagger model was developed for a Hidden Markov Model and Conditional Random Field. TnT which is based on HMM and CRF++ which uses C++ is used as an implementation tool. The study was conducted on 200 sentences and from these sentences, 90% are used for training and 10% are used for testing the tagger model. The taggers achieved an accuracy of 83.58% and 74.63% using reduced tagset for supervised Hidden Markov Model (HMM) and Conditional Random Fields (CRF) based taggers respectively. According to his study, The HMM model can yield better accuracy in tagging unknown words [55].

The study conducted by Zelalem Mekuria focuses on developing a POS tagger for Kafi-Noonoo language using a hybrid approach. The tagger model is developed by combining an HMM model and a rule-based approach. For the Hidden Markov Model, a Viterbi algorithm is used to identify a tag sequence for a given word sequence. In a Rule-based model transformation-based error-driven learning approach is used to learn rules. The tagger was implemented using a Natural Language Toolkit (NLTK) and a python programming language. To conduct the learning and testing 354 sentences and tagsets are identified and prepared. An HMM, Rule-based and hybrid taggers were tested with the same training and testing data and achieved an accuracy of 77.19%, 61.88%, and 80.47% respectively [12].

The study conducted by Patil Nita tries to develop a POS tagger for the Marathi language using Hidden Markov Model. The study used supervised learning methods that use Hidden Markov Model to Marathi text with POS tags. Around 12,000 Marathi sentences that are collected from newspapers are used for training and testing the tagger. According to the study, the POS tagger system achieved an accuracy of 86.61% in predicting correct POS tags for the words [56].

The study conducted by Yajnik and Archit, focuses on developing a POS tagger using a statistical approach to tag sentences for Nepal text. This study is based on HMM. Datasets are selected randomly for training and testing. According to the study, the study provides an accuracy of 95.43% [57].

The study conducted by Anisha Aziz and Sunitha tries to develop a POS tagger system for the Malayalam language. The study was based on the Hybrid models that are traditional

rules and N-grams. To reduce the word's ambiguity, the researcher used a bigram dictionary; the dictionary was composed of 100,000 Malayalam corpora. According to the study rule-based approach is the heart of the tagger which includes 267 manually annotated rules. The result shows that the tagger accuracy is 90.5% [58].

The study conducted Vishal by Goyal, Suman Preet, and Navneet Garg, is conducted on POS tagger for Hindi corpus using a rule-based approach. In this study, the researchers have used a rule-based tagger for the Hindi corpus. The tagger system is evaluated with 26,149 words with 30 POS tags. The evaluation is conducted in different domains like news, stories, and essay it achieved an accuracy of 87.55% [59].

The study conducted by Mr. Vipul Gamit, Rutva Joshi, and Ekta Patel, is conducted on revising POS tagging for the Gujarati language. This study focuses on reviewing the POS tagging methods for the Gujarati language. According to their study, the hybrid approach provides higher accuracy when compared with other methods [51].

The study conducted by Kevin Gimpel, is conducted on annotation, features, and experiments for POS tagging for Twitter. They have developed tagsets, annotated data, developed features, and achieved 90% accuracy [60].

The study conducted by Jyoti Singh, Nisheeth Joshi, and Iti Mathur presents, a POS tagging system for Marathi text using the trigram method. The study used a statistical approach with trigram implementation. According to the evaluation result, the system achieved 91.63% of accuracy [10].

The study Nidhi Mishra and Amit Mishra focuses on developing POS tagging tools for Hindi corpus. The developed tagger tool can read and tokenize sentences and words. In the system parameters like user-friendliness and throughput are used to evaluate the performance [61].

The study conducted by Yuan Lichi focuses on the improvement of automatic POS tagging based on HMM. The study focuses on the Markov family model that is the first introduced kind of statistical model. According to the study, Markov family Model greatly improved the precision of tagging when compared to conventional POS tagging methods under the same testing conditions [62].

2.6 Summary

Different POS tagger approaches use their methods and implementation tools to tag words with their appropriate POS tag. The stochastic model uses frequency, probability, or statistics to assign the most appropriate tag for a given word by calculating the most suitable tag based on the word and its preceding tag. It can work based on supervised and unsupervised learning methods. The HMM, conditional random field, decision trees, neural network, support vector machine, maximum entropy markov models are some of the models discussed in this study that are stochastic and use their algorithm to identify tag sequences for specific word sequences.

The Rule-based approach depends on rules or transformations which are learned by transformation-based error-driven learning to tag words with their appropriate POS tag. Rule-based taggers are the first-ever developed type of POS taggers. The literature reviewed about the syntactic and semantic structure of the Afaan Oromo language is presented in chapter three.

Table 2-3. Summary of Review of Related Research Works

S. No	Author(s)	The objective of the study	The Methodology of the study	Key findings of the study	Remarks
1	[52]	To develop POS tagger for Amharic language.	<ul style="list-style-type: none"> • Long Short-Term Memory recurrent neural networks 	Evaluating the performance of the model bidirectional Long Short-Term memory obtained an accuracy of 93.67%.	The accuracy improvement of the POS tagger is obtained from the increased total number of instances and decreased a number of tags due to segmentation.
2	[5].	To analyze and assess decision trees' application for POS tagging for the Amharic language.	<ul style="list-style-type: none"> • J48 decision tree classifier • Weka C4.5 algorithm • 1065 sentences from the NEWS, including 210,000 words • 10-fold cross-validation 	Evaluating the performance of the model obtained an accuracy of 84.9%. frequency of the target word with its neighboring context is not sufficient for POS tagging	Even though, the accuracy of this study is encouraging further study to improve the accuracy to reach the implementation level is recommended.
3	[8].	To develop a POS tagger model for the Tigrigna language using a hybrid approach.	<ul style="list-style-type: none"> • Hidden Markov Model • Rule-based approach • 1000 sentences • 26,000 words • 36 tagsets 	<ul style="list-style-type: none"> • The result shows that HMM tagger performance is 89.13%. • Rule-based tagger performance is 91.8%. 	Therefore, to assist researchers, it will be of great paramount if a standard corpus for Tigrigna language is developed that will be available for NLP researchers in Tigrigna language.

				<ul style="list-style-type: none"> • By combining the two approaches (hybrid) the accuracy is improved to 95.88% 	
4	[3].	To develop a POS tagger prototype for the Afaan Oromo language.	<ul style="list-style-type: none"> • HMM approach with unigram and bigram models implementing the Viterbi algorithm. • Java programming language • 10-fold cross-validation. • 159 sentences which are composed of 1621 words • 17 tagsets 	<ul style="list-style-type: none"> • Unigram and Bigram models yield 87.58% and 91.97% accuracy 	The accuracy and effective processing of natural language processing applications that need annotated data sets were dependent upon standardized and sufficient amounts of the corpus
5	[53].	To show the applicability of Conditional Random Fields in POS tagging for a morphologically complex language	<ul style="list-style-type: none"> • Conditional Random Fields. • 1000 manually annotated token • 10 POS tagsets 	An accuracy of 74% is obtained for POS tagging and 84% for Amharic word segmentation.	The word division and POS labeling were completed generally freely because of scant assets.

6	[54]	To develop a POS tagger for Afaan Oromo using a hybrid approach.	<ul style="list-style-type: none"> • HMM model • Transformation or Rule-based approach. • Natural Language Toolkit (NLTK) and a python programming language. • 1517 sentences • 35 tag sets 	HMM, Rule-based, and hybrid tagger were tested with the same training and testing data and achieved an accuracy of 91.9%, 96.4%, and 98.3% respectively.	To increase the performance of the tagger wide coverage/domain area of training data and morphologically segmented words were recommended.
7	[55]	To develop a part-of-speech tagger for Wolayta Language.	<ul style="list-style-type: none"> • Hidden Markov Model • Conditional Random Field. • C++ is used as an implementation tool • 200 sentences • 22 tag sets 	The taggers achieved an accuracy of 83.58% and 74.63% using reduced tag set for supervised Hidden Markov Model and Conditional Random fields based taggers respectively.	Most of the Ethiopian languages are under-resourced and do not have large size POS tagger annotated corpus, they can benefit from a semi-supervised approach.
8	[12]	To develop a POS tagger for Afaan Oromo using a hybrid approach.	<ul style="list-style-type: none"> • Hidden Markov Model, rule-based, and hybrid • 354 sentences 	Accuracy of 77.19%, 61.88%, and 80.47% for Hidden Markov Model, rule-based and hybrid taggers respectively	Arrangement of a reasonable corpus that contains messages which speak to various types like papers, fiction, course books, parliamentary reports,

					and so on. Would have a great role in the performances.
9	[56]	To develop a POS tagger for the Marathi language.	<ul style="list-style-type: none"> • Hidden Markov Model • 12,000 Marathi sentences 	The POS tagger system achieved an accuracy of 86.61% in predicting correct POS tags for the words.	This POS tagging system will be integrated with a named entity recognition system to discover the effectiveness of POS tagging for named entity recognition for Marathi language.
10	[57]	To develop a POS tagger using a statistical approach to tag sentences for Nepal text.	<ul style="list-style-type: none"> • Hidden Markov Model • 45,000 Nepali words • 41 tagsets 	The POS tagger system achieved an accuracy of 95.43%	Errors can be minimized by applying proper grammar rules and that is the future task to work on. Another approach to improve accuracy is to apply Conditional Random Fields (CRF) with a morphological analyzer.
11	[58]	To develop a POS tagger system for the Malayalam language.	<ul style="list-style-type: none"> • Hybrid models that are traditional rules and N-grams. • 100,000 Malayalam tokens 	The tagger system achieved an accuracy of 90.5%.	For the unidentified words, it can be caused by either the root word may not be in corpus or bigram, or the absence of rule. So adding the word, bigram or

			<ul style="list-style-type: none"> • 267 manually annotated rules. 		rule, we can improve the result and enhance the work.
12	[59]	To develop a POS tagger for Hindi corpus using a rule-based approach.	<ul style="list-style-type: none"> • Rule-based tagger • 30 POS tagsets. • 26,149 words 	The tagger system achieved an accuracy of 87.55%.	By increasing the size of the database accuracy of the part-of-speech tagger can be increased. A Hybrid based system can be developed to increase the accuracy of the system.
13	[60]	To annotate, features, and experiments for POS tagging for Twitter.	<ul style="list-style-type: none"> • 25 tags. • manually tagged 1,827 tweets, 1.9 million tokens from 134,000 unlabeled tweets 	The tagger achieved an accuracy of 90%.	We also believe that the annotated data can be useful for research into domain adaptation and semi-supervised learning.
14	[62]	To improve an automatic POS tagging based on HMM.	<ul style="list-style-type: none"> • Hidden Markov Model 	Experimental results show that this part-of-speech tagging method based on Markov Family Model has greatly improved the precision comparing the conventional POS tagging method based on Hidden Markov Model under the same testing conditions.	The Markov Family Model is also very useful in other natural language processing technologies such as word segmentation, statistical parsing, text-to-speech, optical character recognition, etc.

15	[10]	To develop a POS tagging system for Marathi text using the trigram method.	<ul style="list-style-type: none"> • Trigram method • Unigram, Bigram, Trigram, and HMM Methods. • Test corpus of 1000 sentences (25744 words). 	According to the evaluation result, the system achieved 91.63% of accuracy	The performance of the current system is good and the results achieved by methods are excellent. We believe that future enhancements of this work would be to improve the tagging accuracy by increasing the size of the tagged corpus.
----	------	----------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

CHAPTER THREE

3 THE GRAMMAR OF AFAAN OROMO AND TAGSET

3.1 Introduction

The Oromo community is the largest ethnic group in Ethiopia. They have their own language called Oromo language, also known as Afaan Oromo. Afaan Oromo is a language from the Cushitic language family and is spoken by the largest ethnic group in Ethiopia. It is one of the languages with the largest number of speakers in Africa next to Arabic and Hausa [63]. It is spoken in three countries in the horn of Africa including Ethiopia, Kenya, and Somalia [64]. The language is also called Oromiffa. By the year 1993, the charter of the transitional government of Ethiopia granted the language to become regional official language [65]. Since then, the language has been used as a working language of the Oromia regional state and as a medium of instruction for primary schools throughout the region [65]. Now a day's huge amount of literature works such as textbooks, fiction, and newspapers are being written and published in Afaan Oromo.

According to [66], the Afaan Oromo language is said to have six major dialects, Northern Afaan Oromo (Baate and Raayyaa), Western Afaan Oromo (Macca), Highland Shawan Afaan Oromo (Tuulama), Eastern Afaan Oromo (Hararge), Central Afaan Oromo (Gujii and Arsii), and Southern Afaan Oromo (Boorana). The language does not have a standard form, therefore all of the dialectic variations are being used in written materials and the mass media.

3.2 Afaan Oromo Writing system

Afaan Oromo adopted Latin script for writing and it is called *Qubee*, it has 33 characters representing unique sounds with 26 of them are the same as English language and additional 7 compound characters called "Qubee Dachaa" (*ch, dh, ny, sh, ph, ts, and zy*) [64]. The language has a set of five short (a, e, i, o, u,) and five long (*aa, ee, ii, oo, uu*) vowels. The variation in the length of the vowels resulted in the change of meaning.

3.3 Afaan Oromo Morphology

Morphology is a branch of linguistics that studies the structure of words or components of a word. Morphology is categorized into two: inflection and derivational. Inflectional morphology is concerned with inflectional changes in words where stems are combined with grammatical markers that does not result in a change in the parts of speech. On the

other hand, Derivational morphology deals with those changes that result in changing classes of words (parts of speech). The smallest unit to which words can be broken down into is called morphemes. A morpheme either possess meaning or conveys grammatical function [67]. Morphemes are of two types: free and bound morphemes. Free morpheme can stand alone with a specific meaning whereas, bound morpheme cannot stand alone with meaning. According to [67] morphemes can be divided into root (stem) and affixes. The root (stem) is the main morpheme of the word which carries the main meaning, whereas affixes add additional meanings of various kinds.

In Afaan Oromo stems are bound morpheme they cannot stand alone with meaning for instance the root “*bar-*”, does not have meaning when it stands alone. Roots are pronounceable only when affixes are added to them. Likewise, affix is also bound morpheme it cannot occur independently. These affixes are of three types: prefix, suffix and infix. Prefix and suffix occur at the beginning and at the end of a root respectively. For instance, in a word, *barumsa* (education), *-umsa* is a suffix and *bar-* is a stem. On the other hand, infix is a morpheme that is inserted within another morpheme. Afaan Oromo does not have infixes [68]. Afaan Oromo is a language that has very complex and rich morphology. It is a language that involves very extensive inflectional and derivational morphological processes [69]. Most of the grammatical information in the language is conveyed through affixation. In Afaan Oromo, words can be formed through inflection, derivation, compounding, and reduplication [68].

3.3.1 Word order in Afaan Oromo

For sentence construction, Afaan Oromo language uses *subject-object-verb (SOV)* structure [70]. In subject-object-verb sentence structure, the subject comes first, followed by the verb and followed by object.

Example (1), in Afaan Oromo sentence:

“caalaan ciree nyaate” to mean, “chala ate breakfast”.

In this sentence, “caalaa” is a subject, “ciree” is an object and “nyaate” is a verb. In addition to this, in Afaan Oromo the adjectives follow a noun or pronoun that they modify.

Example (2), in the Afaan Oromo sentence

“caaltuun bareeddu dha”, which means “Caaltuu is beautiful.”,
“Caaltuu” is a noun and “bareeddu” is an adjective that follows noun.

3.4 Word Categories

The Afaan Oromo language words can be categorized into nouns, verb, adverb, adjective, pronoun and prepositions [71] [68]. In this section those categories of words in Afaan Oromo language is discussed in detail.

1. Nouns

A noun is a word that names something, such as a person, place, thing, or idea. Nouns in Afaan Oromo are inflected for gender, definiteness and number [72].

a. Gender

Afaan Oromo has a two gender system (feminine and masculine). Frequent gender markers in Afaan Oromo include *-eessa/-eettii*, *-a/-ttii* or *-aa/tuu*.

<i>Afaan Oromo</i>	<i>Construction</i>	<i>Gender</i>	<i>English</i>
Obboleessa	obol + eessa	male	brother
Obboleettii	obol + eettii	female	sister
Jabaa	jab + aa	male	strong
Jabduu	jab + duu	female	strong

b. Number

Afaan Oromo has different suffixes to form the plural of a noun. The use of different suffixes differs from dialects to dialects. There are more than ten major and very common plural markers in Afaan Oromo including: *-oota*, *-oolii*, *-wwan*, *-lee*, *-an*, *een*, *-eeyyii*, *-oo*, etc.).

For example (4)

<i>Singular</i>	<i>Plural</i>	<i>English translation</i>
Hiriyaa	hiriyoota	Friend/ Friends
Gaangee	gaangolii	Mule/Mules
Laga	Lageen	River/rivers

c. Definiteness

In Afaan Oromo demonstrative pronouns like *kun* (this), *sun* (that) are used to express definiteness. In some Afan Oromo dialects the suffix *-icha* for male and *-ittii(n)* for female and for undermining usually has a singularize function is used where other languages would use a definite article.

For example (5)

<i>namicha this/ the man (Subject)</i>	<i>kitaabni sun that/ the book</i>
<i>nitittii this/ the women (Object)</i>	<i>namtichi kun this / the man</i>

2. Verbs

A word that is usually one of the main parts of a sentence and that expresses an action, an occurrence, or a state of being. Afaan Oromo has base (stem) verbs and four derived verbs from the stem. Moreover, verbs in Afaan Oromo are inflected for gender, person, number and tenses [68]. There are four derived stems, the formation of which is still productive, Auto-benefactive, Passive, Causative and Intensive.

a. Auto-benefactive

The Afan Oromo auto-benefactive (or "middle" or "reflexive-middle") is formed by adding -(a) adh, -(a) ach or -(a) at or sometimes -edh, - ech or – et to the verb root. This stem has the function to express an action done for the benefit of the agent himself.

Example (6)

Word (the verb)	Root/stem	meaning
Bitachuu	bit	to buy for oneself

b. Passive

The Oromo passive corresponds closely to the English passive in function. It is formed by adding -am to the verb root. The resulting stem is conjugated regularly.

Example (7)

(a) Beek- *know*, Beekam- *be known*

c. Causative

The Afaan Oromo causative of a verb corresponds to English expressions such as ‘cause’, ‘make’, ‘let’. With intransitive verbs, it has a transiting function. It is formed by adding -s, -sis, or -siis to the verb root.

Example: (8)

(a) deemuu - to go, deemsisuu - to cause to go

d. Intensive

It is formed by duplication of the initial consonant and the following vowel, geminating the consonant.

Example (9)

(a) Waamuu - to call, invite wawwaamuu - to call intensively

3. *Adjectives*

An adjective is a word that describes a noun or a pronoun. Adjectives in a sentence are used to modify nouns to show the quality of things.

Examples (10)

(a) ‘boontun *bareeddu* dha’ (bontu is beautiful.)

(b) ‘caalaan *dheeraa* dha’ (chala is tall.)

In the above examples the word ‘bareeddu’ (beautiful) and ‘dheeraa’ (tall) are adjectives. Afaan Oromo adjectives can be formed from compound words [72]. For instance, ‘humna dhabeessaa’ (weak), ‘simbo qabeessa’ (handsome) are some of adjectives constructed from compound words. Adjectives inflect for number and gender in Afaan Oromo language.

Singular	Plural	masculine	Feminine
Guddaa	Gudguddaa	guddaa	guddoo
jabaa	jaboota	jabaa	Jabduu
Ko’eessa	Ko’eeyyii	Ko’eessa	Ko’eettii
cimaa	ciccimoota	cimaa	Cimtuu

4. *Adverbs*

Adverbs are words that are used to modify verbs. In Afaan Oromo adverbs come before the verb they modify. Afaan Oromo adverbs are categorized as adverbs of time, place, and manner [68]. Adverbs of time show the time when the action takes place. Mostly adverbs of time answer the question of when the action takes place. Some of the words that can be used as adverbs of time in Afaan Oromo language includes: ‘amma’ (now), ‘boru’ (tomorrow), ‘kaleessa’ (yesterday), ‘yoom’ (when), ‘har’a’ (today), ‘galgala’ (tonight) etc.

Example (11)

a) ‘salamoon *kalessa* dhufe.’ (solomon came yesterday)

b) ‘Adaama boru ni deemna.’ (We will go to Adama tomorrow.)

In these examples the word ‘kaleessa’ (yesterday) and ‘boru’ (tomorrow) are adverbs of time.

Adverbs of place show the place where the action takes place. words that can be used as adverb of place in Afaan Oromo includes: ‘as’(here), ‘achi’(there), ‘gadi’(below), ‘gubbaa’(above), ‘jidduu’(middle), ‘irra’(on),etc.

Example (12)

- a) ‘Tolaan *mana* jira.’ (Tola is at home.)
- b) ‘Inni *konkolaataa irra* jira’ (he is on the car.)

Adverb of manner show how the action of the sentence is done. The following are Afaan Oromo words that can be used as adverb of manner ‘ariitin’(quickly), ‘suuta’ (slowly), ‘akka gaarii’ (well) etc.

Example (13)

- a) ‘Isheen ariitin figdi.’ (She is running quickly).
- b) ‘Calaan baay’ee cimaa dha.’ (Chala is very clever).

In the above two sentences the word ‘ariitin’ (means quickly) and ‘baay’ee’ (mean very) are adverbs of manner.

5. Pre, para, and post positions

Afaan Oromo languages use prepositions, postpositions and Para positions [70].

i. *postpositions*

a. *suffixed post positions*

<i>post positions</i>	<i>English equivalent</i>
-tii	in, at, to
-rra, irra	on
-rraa, irraa	out of, from

Example (14)

- a) hojitti yoom deebina? Which means (When shall we get back to work?)

b. *Postpositions as independent words*

Some of the independent word postpositions are listed in the following table.

<i>Postpositions as independent words</i>	<i>English Equivalent</i>
Ala	outside
Bira	beside
Booda	after, behind
Duuba	behind
Wajjin	with, together with

Example (15)

- (a) Qarshii nu bira jiru fudhadha. Which means (Take the money with us.)

ii. *Prepositions*

<i>Prepositions</i>	<i>English Equivalent</i>
Akka	like, according to
Gara	to, in the direction of
Hanga, hamma	to, in the direction of
Karaa	along, the way of, through

Example (16)

(a) Hanga deebi`e dhufutii na eegi. Which mean (Wait me until I come back.)

iii. Para positions

<i>Para-positions</i>	<i>English Equivalent</i>
Gara...tti	To
Gara...tiin	from, from the direction of
Hanga...tti	up to,until

Example (17)

(a) booru gara mana keenyatti deebina. Which means (we will get back to our home tomorrow.)

3.4.1 Punctuation Marks in Afaan Oromo

Punctuation marks are placed in a text to make the reading easier and to make the meaning clear. In the Afaan Oromo language, the same punctuation pattern is used just like that of other languages that use the Latin writing system [68]. The most commonly used punctuation marks in Afaan Oromo are:

- a) **Tuqaa**, *Full stop* (.): is a statement terminator, it is used at the end of a sentence. It is also used in abbreviations.
- b) **Mallattoo Gaafii**, *Question mark* (?): is used at the end of questions or interrogative.
- c) **Rajeffannoo**, *Exclamation mark* (!): is used at the end of command and exclamatory sentences.
- d) **Qooduu** *Comma* (,): it is used to separate listing in a sentence or to separate the elements in a series.
- e) **Tuqlamee** *colon* (:): is used to separate and introduce lists, clauses, and quotations, along with several conventional uses, and etc.

- f) **Hudha apostrophe** (‘): Even though apostrophe is as a punctuation mark in the English language, it is not a punctuation mark in Afaan Oromo it is part of words. For instance, apostrophe in the word *re`ee* (which means goat in English) is part of the word.

3.5 Afaan Oromo Tagsets

Different researcher tried to identify tagset for Afaan Oromoo language with the help of language experts. Different researchers are developed Afaan Oromoo tagsets for their study.

In this study 27 tagsets were identified with the help of linguistic experts and used to tag words to their appropriate word class in order to compare the tagger models. The tagsets used in this study are summarized in the table below with description and examples.

Table 3-1. Afaan Oromo tagsets

NO	Basic class	Tag No	Tags	Description	Example
1.	NOUNS	1.	/NN	This tag is used for Common nouns and proper nouns which are singular and plural.	Dhagaa ”Stone”, Mana ”House”, Gamoo ”Building”, Bosona ”Jungle”, Hiriya Friend , Seena ”History”, Qubeelaa “ring”, Aduu ”Sun” Gaadisa ”Shadow”, Bishaan ”Water” ,Gaara ”Mountain”
		2.	/NP	This tag is for nouns + Preposition	Dhagaadhan “with stone”, Bishaanin “with water”, Firaafan ”For relative”,Mukaraa ”From stick”,Biyyaraa “From Country”, Harakaraa “From a hand”
		3.	/NC	This tag is for nouns that are suffixed with conjunctions.	Bishaan fi ”Water and, Dhoqee fi “Mud and”, Qamalee fi “ monkey and”,
		4	/NPO	This tag is used to tag nouns that show possessiveness.	Kan abbaa ”his father”, kan Jaarsaa “ the elder”, kan biyyaa ”the countries”
		5.	/NV	This tag is used to indicate the nouns which are formed from verbs.	Taphataa ”Player”, Oggessa ””, Barsisa ”Teacher”, Barataa ”Student” , Daldalaa “Merchant”
		6.	/NZ	This tag is used for all other nouns. E.g. abstract noun, collective noun, etc.	Du’a “Death”, Biyya “Country”, Jiba “Hate”, Jaalala “Love”

2.	VERBS	7.	/VV	This tag is used to tag all other types of verbs.	Deemii “Go”,Taa’i “Sit”, Kottu “Come”,Nyaadhu “Eat”,seenii “enter”, Baresii “Write”, Fiidii “Bring”, Galchii “let in” deebisii “return”
		8.	/VP	This tag is used for verbs that are suffixed with pre,para and post positions	Kaachuuf ”to run”, waamuuf ”to call”, Taphachuuf ”to play”, deemuuf ”to go”, Baresuuf ”to write”, Dubisuuf “To read”
		9.	/VC	This tag is used for verbs that are suffixed with conjunctions.	Galchuufi, “to let in”,deemuufi ”to go”,Nyaachuufi ”to eat”
		10.	/VNE	This tag is used for verbs that are prefixed with negation	Hin nyaatin “don’t eat” ,hin deemin ”don’t go”,Hin taa’in don’t sit
3.	ADVERBS	11.	/AD	This tag is used to tag all adverbs.	Kalessa “Yesterday”,”yoom” “when” ,Boruu ”Tomorrow”
4.	ADJECTIVES	12.	/AJ	This tag is used to tag adjectives.	Goota ”Hero”,Jabaa ”Strong” Gamna ””,Dheeraa ”long”, Gabaabaa ”Short”, Furdaa ”Fat”
		13.	/AJP	This tag is used to tag adjectives that are suffixed with prepositions	
		14.	/AJC	This tag is used to tag adjectives that are suffixed with conjunctions.	Dheeraaf ”for long”, Furdaaf “for fat”
5.	PREPOSITIONS	15.	/PR	This tag is used to tag prepositions.	Ala “outside”,bira ”beside” booda ”behind”
6.	PRONOUNS	16.	/PP	This tag is used to tag personal pronouns that are singular or plural.	Isa “He”, Ishii “She”, Isaan “them”

		17.	/PPO	This tag is used to tag possessive pronouns that are singular and plural.	Kanisaa “His”,kanishii ”Her’s”, Kanisaanii “Theirs”
		18.	/PD	This tag is used to tag demonstrative pronouns.	Kun ”this”,achii ”there”, as “Here”
		19.	/PI	This tag is used to tag interrogative and indefinite pronouns.	Maal “ what” , essa ”where”, enyuu ”who “,kam “which”
		20.	/PPR	This tag is used to tag Pronouns that are suffixed with prepositions.	Enyuuraa ”From whom”, Enyuuf for whom
		21.	/PC	This tag is used to tag Pronouns that are suffixed with conjunctions.	Inni “him”,ishiifi ”here and”,achhis ”there also”
7.	NUMBERS	22.	/CN	This tag is used to tag cardinal numbers.	Tokko ”one”, lamma ”two”
		23.	/ON	This tag is used to tag ordinal numbers.	Jalqaba ”First”,lammata ”second”
8.	CONJUNCTIONS	24.	/CC	This tag is used to tag conjunctions.	Fii ”and”, yookan “or”
9.	INTERJECTIONS	25.	/IN	This tag is used to tag interjections.	Ishoo , wayyoo “Ohh”
10.	PUNCTUATIONS	26.	/PU	This tag is used to tag all punctuation marks.	Tuqaa ”period”, ? mallato gaafii ”question mark”, ‘ hudhaa ”apostroph”
11	NEGATION	27.	/NG	This tag is used to show negation.	Hin sobin “don’t lie”,hin nyaatin ”don’t eat”,hin taa’in ”don’t sit”

CHAPTER FOUR

4 DESIGN OF AFAAN OROMOO PART-OF-SPEECH TAGGER

4.1 Introduction

To tag words with their appropriate part-of-speech tag different tools and techniques are used. Different corpus types are used, such as manually annotated corpus and unannotated corpus, for training and testing the tagger model. This chapter discusses the tools used to implement the POS tagger, corpus preparation, and design of an HMM and Rule-based in detail.

4.2 Corpus Preparation

A corpus which is a plural form of corpora is a collection of a huge number of texts or spoken data [22]. The corpus is tagged manually with its suitable POS tag by linguistic experts before training in supervised learning methods. The unsupervised learning method doesn't need a pre-tagged corpus.

A collected corpus can be classified as a balanced corpus and category-specific corpus [73]. In a balanced corpus, the text can be collected from various sources, such as newspapers, NEWS, magazines, scientific research, fiction, stories, etc. This can maximize the performance of the tagger by providing a variety of texts for training [73]. But preparing a balanced corpus requires time, money, experts' effort as it needs the data to be collected from different sources or areas. The category-specific corpus contains the text that is compiled from a specific or single origin [73].

The performance of the tagger is directly linked to the variety of the corpus it uses for training [74]. When the tagger trains using a variety of text its performance can increase. The tagger that trains using a balanced corpus can perform better in tagging new words than the category-specific corpus [74].

Not only has the choice of the methods/approaches, but the coverage of domain of training data also had a great impact on the performance of the taggers [74]. This specific domain coverage caused of occurrence of many unknown words which greatly degraded the performance of the taggers.

For this study, the corpus was collected from the NEWS of and the bible. The NEWS is collected from BBC Afaan Oromo, VOA Afaan Oromo 2 months NEWS and the bible collected from Afaan Oromo New Testament bible specifically Mathew gospel.

Linguistic experts tagged the collected corpus using the tagsets which are identified and described in this study. The manually tagged corpus is used both to train the tagger and as a reference to evaluate the performance of the trained tagger. The researcher collected more than 25,512 words from these sources that can be used to train the tagger and test the tagger's performance.

4.3 Corpus Split Methods

In supervised methods, the corpus has been used in different ways to evaluate the performance of the taggers such as train/test split and cross-validation [75], [76]. In train/test split the dataset is divided into two parts and the majority is used for training while the rest is used for testing the trained model [76]. This method usually split our data into the 80:20 or 70:30 ratios between the training and test data [76]. In cross-validation, the data is split into k subsets and trained on $k-1$ of those subsets, and the last subset is reserved for testing [76].

To experiment the corpus that was collected from different sources is divided into a training set and test set using Train/test split. The tagger was trained on 80% of the collected corpus and tested by using the rest 20% of the data. The accuracy of the tagger is directly related to the size corpus that is used in training and testing [76].

Table 4-1. Pros and cons of train/test split and cross-validation

No		Pros	Cons
1	Train/test split	<ul style="list-style-type: none"> • Easy to implement and interpret • Less time-consuming in execution • The computing power is low. • The feedback for model performance can be obtained quickly. 	<ul style="list-style-type: none"> • If the dataset is small, keeping a portion for testing would decrease the accuracy of the predictive model. • If the split is not random, the output of the evaluation matrices is inaccurate.
2	K-fold Cross-Validation	<ul style="list-style-type: none"> • More realistic evaluation matrices can be generated. • Reduce the risk of over-fitting models. 	<ul style="list-style-type: none"> • May take more time in evaluation because more calculations to be done. • The computing power is high. • So, it may take some time to get feedback on the model's performance in the case of large data sets. • Slower feedback makes it take longer to find the optimal hyper-parameters for the model.

4.4 Tools Used

To tag Afaan Oromo text with their appropriate POS tag, Natural Language Toolkit (NLTK) and python programming language are used. The rationale behind the selection of tools is described below.

Natural Language Toolkit (NLTK) is a platform that is used to build NLP programs in python [54]. It is used in many NLP tasks because of its simplicity, consistency, extensibility, modularity and it is well-documented and it contains useful libraries and tools that are used for text preprocessing and analysis when compared to other libraries [77]. It supports many languages and enough approaches for every NLP task with easy third-party extension and higher flexibility [77].

Python is powerful programming for text processing in Natural Language processing due to its high functionality [77]. Python is widely used in NLP because it has simple syntax, structure, good string handling functionalities and it is rich in text processing tools.

Python is an open-source language with a large number of ready-to-use libraries for NLP [77]. It is an object-oriented, scripting, and dynamic language; as an object-oriented language, python allows data and methods to be encapsulated and reused easily, as a scripting language python allows the execution of procedures and programs repeatedly [77]. This can reduce human effort, time, and execution cost also as a dynamic language python allows dynamic typing of variables and provides an easy memory management mechanism [78].

The flexible nature of python makes it a powerful language for rapid application development in many areas especially in NLP tasks [77].

4.5 Designing Hidden Markov Model Tagger

The HMM approach is distinct from the other POS tagging approaches, In the sense that it considers the best combination of tags for a sequence of lexicons, while the other tagging methods greedily tag one word at a time, regardless of the optimum combination [22].

A Hidden Markov Model (HMM) is the most common statistical approach widely used in speech and language processing [23]. An HMM is used to find the most optimal tag sequences for a given sequence of words. It deals with both observed and hidden events of the part-of-speech tagging problem [22]. In POS tagging, the sequence of words in a sentence are observed events and parts of speech tags are hidden events.

An HMM determines the optimal tag sequence for a given word sequence by using the formula [22]:

$$T_n = \underset{t_n}{\text{Argmax}} P(t_n | w_n) \dots \dots \dots \text{eq.1}$$

Eq.1 implies the argument that maximizes the probability of a word occurring with a specific part-of-speech tag. Where, t_n is the sequence of tags (the hidden states) and w_n is a sequence of words (observable states).

The eq.1 can be simplified by using the generative model Naïve Bayes rule to transmit it to a set of probabilities that are easier to compute [22]. It can be simplified as:

$$T_n = \underset{t_n}{\text{Argmax}} \frac{P(w_n | t_n) * P(t_n)}{P(w_n)} \dots \dots \dots \text{eq. 2}$$

The eq. 2 can be further simplified by dropping the denominator $P(w_n)$ because the sequences of words for each tag sequence cannot be changed. Hence it can be simplified as:

$$T_n = \underset{t_n}{\text{Argmax}} P(w_n|t_n) * P(t_n) \dots \dots \dots \text{eq. 3}$$

Based on the eq. 3 the Hidden Markov Model suggests two different assumptions. In the first assumption, the probability of a word is dependent on its tag only. A word will be tagged most probable tag by the HMM tagger without considering the neighboring words or tags. Computed by a lexical model that is a component of the Hidden Markov Model.

$$P(w_n|t_n) = \prod_{j=1}^n P(w_j|t_j) \dots \dots \dots \text{eq. 4}$$

The second assumption describes that the probability of a tag appearing with a specific word is dependent on the most recent tag only. Computed by a lexical model that is a component of the Hidden Markov Model.

$$P(t_n) = \prod_{j=1}^n P(t_j|t_{j-1}) \dots \dots \dots \text{eq. 5}$$

The eq. 4 and 5 make the Hidden Markov Model tagger component that is the lexical model and contextual model respectively.

4.5.1 Lexical Model

The lexical model computes the probability of each word appearing with a specific tag in the training set [22]. The probabilities are calculated by counting the data on the training corpus. The lexical model calculates the emission probability. The lexical probability $P(w_i|t_i)$ can be computed by counting the labeled corpus by using the formula:

$$P(w_i|t_i) = \frac{\text{count}(t_i, w_i)}{\text{count}(t_i)} \dots \dots \dots \text{eq. 6}$$

4.5.2 Contextual Model

The contextual model computes the contextual probabilities of the tag occurring with given previous tags [34]. These models compute the transition probabilities. The contextual model is also known as the N-gram model [22]. It computes by changing the value of N. In this study unigram model and bigram model (n = 1 and 2) are used. The transition probability is computed by counting the frequency of the tag followed by a specific tag in the training corpus. The probability is computed by

$$p(t_i|t_{i-1}) = \frac{\text{count}(t_{i-1}|t_i)}{\text{count}(t_{i-1})} \dots \text{eq. 7}$$

HMM uses a Viterbi algorithm to find the most likely sequence of hidden states or tag sequences for a sequence of words [22]. Viterbi algorithm is a dynamic programming algorithm that can solve decoding problems of HMM [22]. Decoding is the task of identifying hidden sequences of states using observed states' sequences.

The Viterbi algorithm computes all previous states which have the highest probability such as transition probability and observation probability to find the possible sequence of tags and it minimizes the volume of calculation and provides better speed and accuracy [11]. Transition probability and observation probability are given to the Viterbi matrix calculator to compute the most probable tag sequence probability [8].

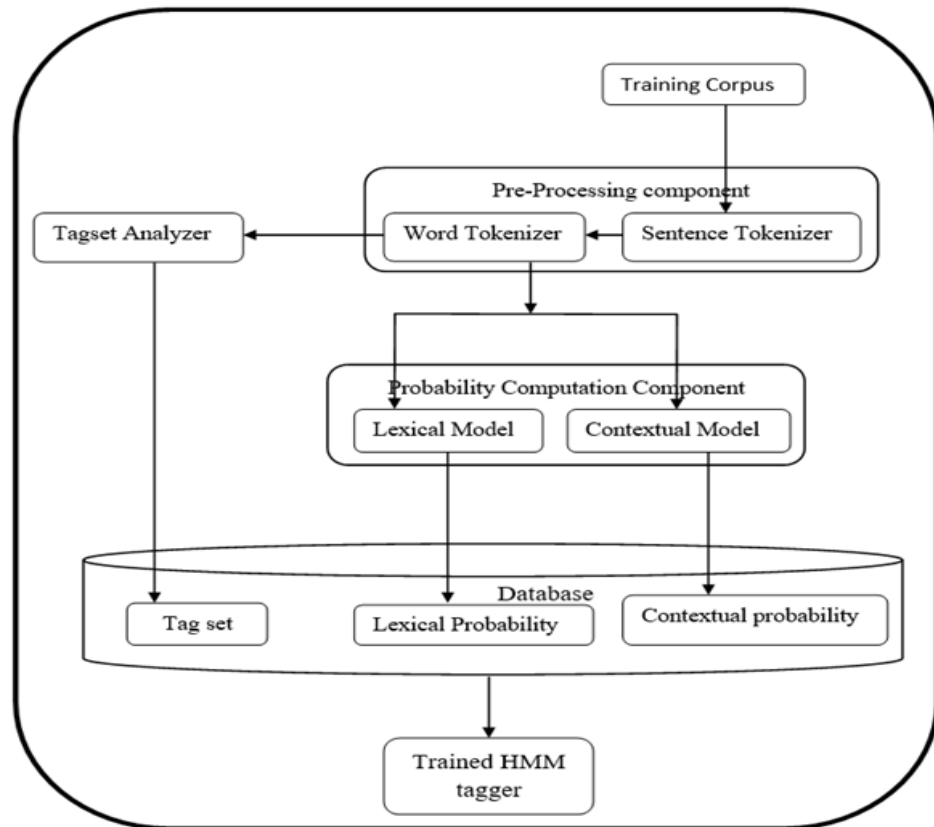


Figure 4-1. HMM tagger training model

In the supervised learning model, the tagger trains by using a manually tagged corpus [11]. In the above Figure 4-1, the manually tagged corpus is provided to the preprocessing component where it can be tokenized into sentence level and word level. This is conducted by a sentence tokenizer and a word or morphological tokenizer module located in the

preprocessing component. The sentence tokenizer module splits the corpus into a manageable sentence and the word tokenizer module splits the sentence into a manageable word level. After all, sentences are tokenized into a word, a tagsets analyzer extracts the tags from each word and stores them in a database. Tokenized words are given to the lexical model to compute the observation likelihood and store the result in a lexical probability matrix and the contextual model computes the transition probability and stores the result in the contextual probability matrix.

The learned HMM model takes the raw text and tags using the Viterbi algorithm. The Viterbi algorithm uses the stored information i.e., likelihood probability and observation probability to tag the words with their most likely tag. The tagging process is illustrated in Figure 4-2.

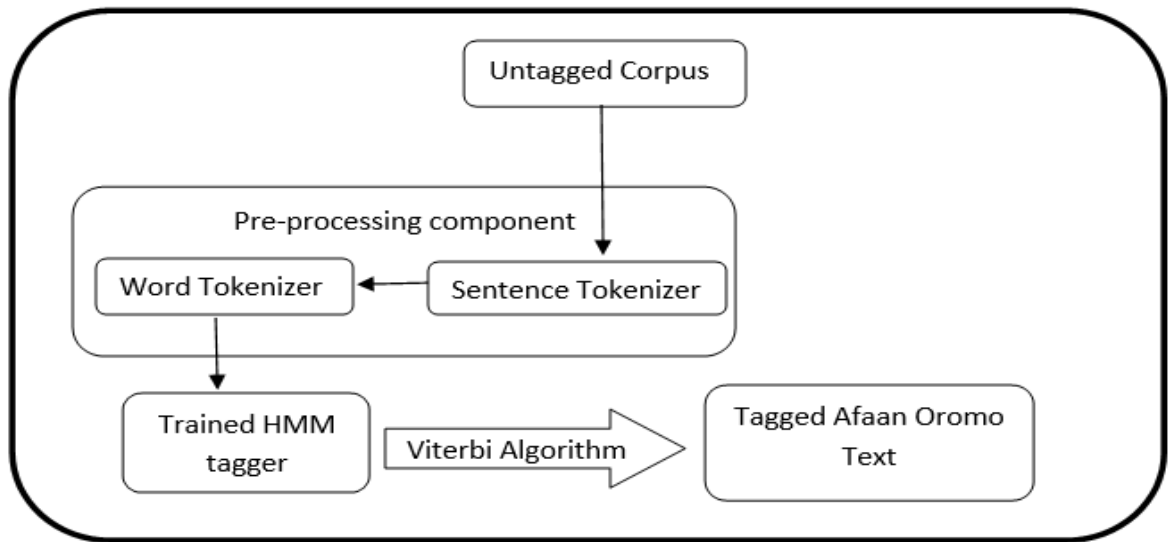


Figure 4-2. HMM tagging process

The text tagged by an HMM model is taken as input by the performance analyzer, then the performance analyzer compares the text tagged by an HMM with the other input which is manually tagged corpus to evaluate the performance of the tagger. The performance analysis of HMM is illustrated in Figure 4-3.

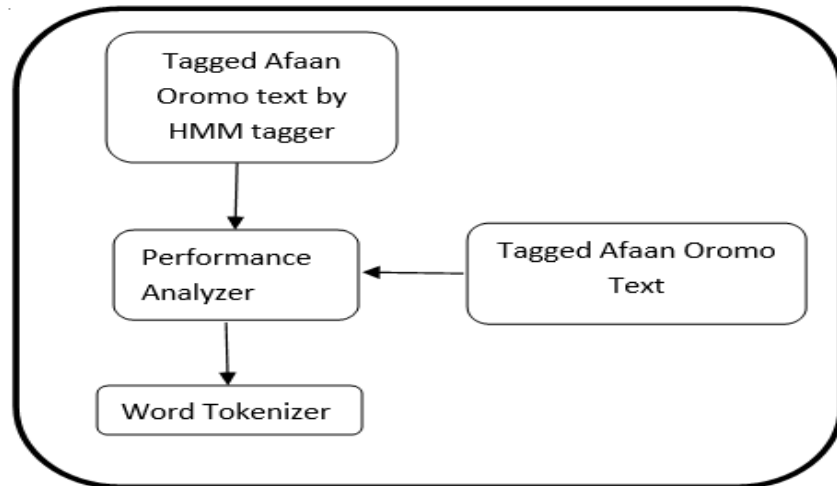


Figure 4-3. HMM performance analyzer

4.6 Hidden Markov Model Architecture

As illustrated in figure below the raw text is given to the system as input and The model parameters $\mu - \{\boldsymbol{\pi}, \mathbf{A}, \mathbf{B}\}$ reflect the language model specifically for HMM. Using corpora, we hope to estimate the HMM's model parameters $\mu - \{\boldsymbol{\pi}, \mathbf{A}, \mathbf{B}\}$. During supervised learning, the HMM's model parameters are estimated based on the labeled data. The Viterbi algorithm tags the input raw data based on the prior learning from the training data and gives the tagged data as an output.

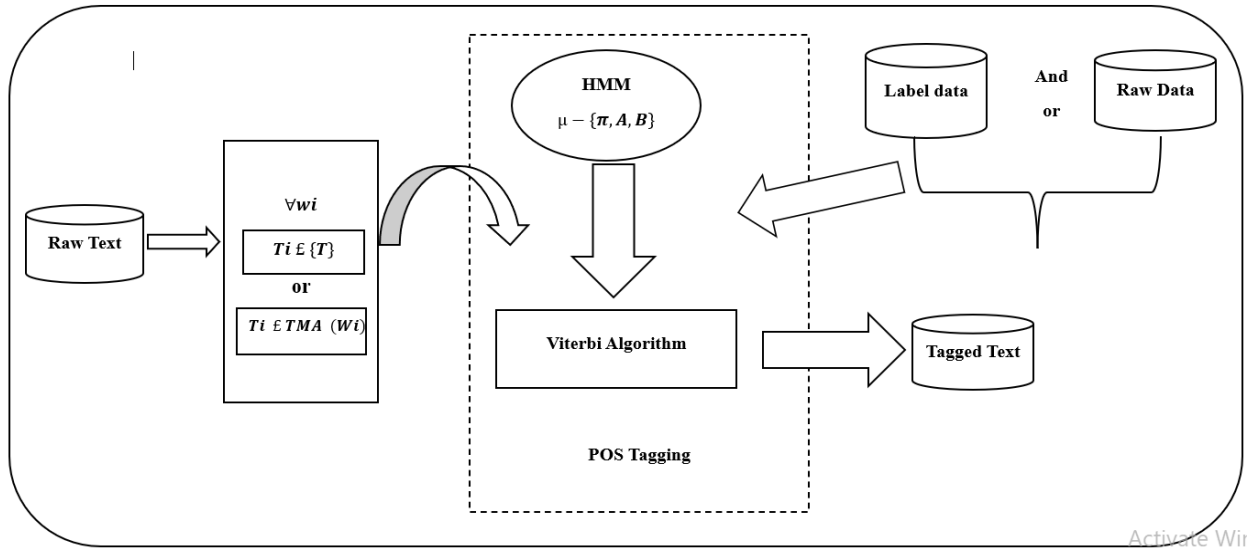


Figure 4-4. Hidden Markov Model architecture

4.7 Designing Rule-Based Tagger

The Rule-based tagger tags the words in a corpus using the rules generated by the TEL [9]. TEL learns the rules and stores them in a specified component for storing rules and for learning rules, it uses the lexical and contextual information of the words [9]. The components of the rule-based tagger model are discussed below.

4.7.1 Transformation-Based Error-Driven Learning

Transformation-based error-driven learning (TEL) is a framework developed for Rule-based learning by Brill [9]. TEL is based on machine-learned rules or transformations which are learned automatically from training corpus without human or expert intervention through error detection [9]. It only needs training data which is a manually annotated corpus that is assumed to be correct. Then the system drives lexical and contextual information from the training corpus that can be used later to label new words with their likely POS tag.

The TEL tagger doesn't use pre-specified grammatical rules that linguistic experts prepare. It learns the rules from the manually tagged corpus through training and this minimizes the errors made while crafting rules, the time and effort of the linguistic expert [54].

TEL works as follows: first the untagged corpus is passed through the initial state tagger, the initial state tagger tags the most likely tag for each word and creates a temporary corpus. The learning phase then takes both temporary and manually tagged corpus, which is

assumed to be correct and compares them for rule derivation. Each time the temporary corpus passes through the learner component a new transformation or rule can be generated after the learner compares the temporary corpus with the manually tagged corpus. This process continues until no transformation or rule can improve the tagging of the temporary corpus. This process finally produces transformations or rules that are going to be used to tag words.

To get final transformations, the learner tries every possible transformation and counts the number of tagging errors after each transformation is applied. After all possible transformations have been applied to the corpus. Then the transformation that resulted in the greatest error reduction is selected.

The learning component has two sub-components namely the lexical rule learner module and contextual rule learner module. Lexical rule learner module drives a rule that assigns the most likely tag to each word that may or may not appear in the training corpus. Contextual rule learner generates a rule for tagging a word in context with their environment. The rule component stores the rules learned by the lexical rule learner module and contextual rule learner module. The transformation-based error-driven learning process is illustrated in Figure 4-5

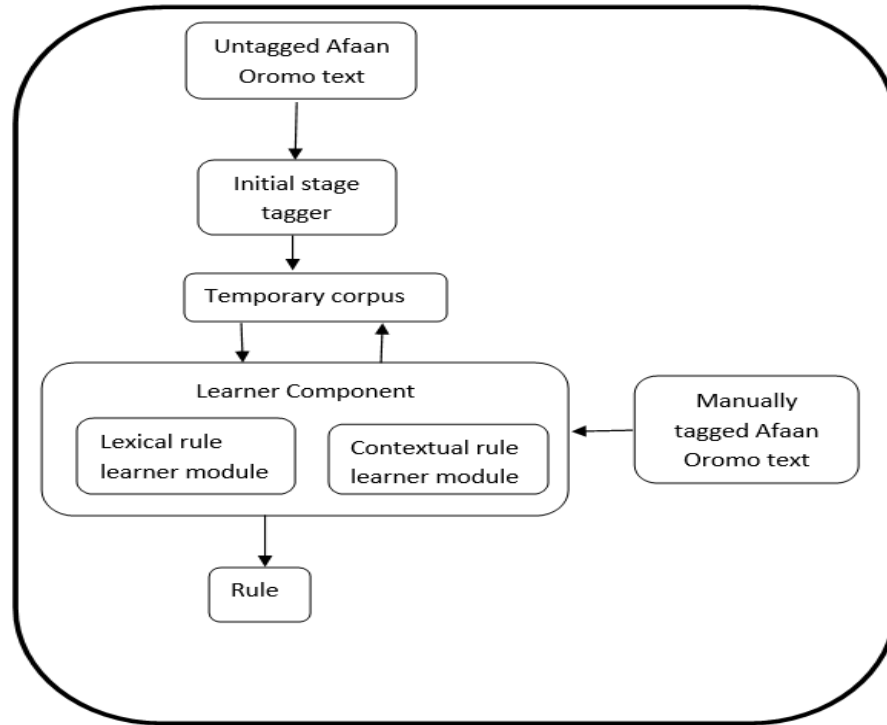


Figure 4-5. Rule based tagger training model

4.7.1.1 Initial Stage tagger

The initial stage tagger tags a word with its most likely tag and produces a temporary corpus. Different taggers can be used as initial stage taggers ranging from statistical n-gram taggers (unigram, bigram, trigram, etc.) taggers to default taggers (a tagger that tags a noun class tag for every word in the corpus).

In this study unigram, bigram, and trigram taggers can be used as initial stage taggers to check the Brill tagger's performance.

4.7.1.2 Learning Phase

The learning component of TEL has two sub-components: lexical rule learner module and contextual rule learner module which are used to generate transformations or rules.

4.7.1.2.1 Lexical Rule Learner

A lexical rule learner is a sub-component of a learning module that is used to drive rules that are used to assign the most likely tag to any word that has appeared in the training corpus or not appeared in the training corpus. Lexical rule learner uses both the text annotated by the initial state tagger and the manually tagged corpus. Lexical rule learner is based on the statistical methods to find the most frequent tag for a word.

In the lexical rule learner, the initial state tagger annotates the text with its most likely tag and produces an output temporary corpus TC_0 , and then the learner produces a set of rules of transformations from pre-specified lexical rule templates that are going to give the best score when applied to the temporary corpus TC_0 [12]. After computing the rules' score, the rules with the best score will be applied to the temporary corpus TC_0 then produce a temporary corpus TC_1 and these rules are stored in the Rule-based component [45]. The best score shows the rule's score, which can give a tagged corpus that resembles the manually tagged corpus.

Again, the same process continues until there are no transformations or rules that are going to change the tag of the temporary corpus. Each time the learner tries to identify the best transformation rules that are going to increase the quality and accuracy when applied to the temporary corpus and stores these transformations or rules in the rule component.

The score of the transformation rules may be positive, negative, or zero. The positive value shows that the rule has improved the tag, the negative value shows that the rule has been worsened the tagging and the zero value shows that there is no change in the tag.

The score for a rule can be computed as [45]:

$$\text{Score}(s) = P(Y|W) - P(X|W)$$

Where Y is the new tag, X is the old tag and W is the word.

Some of the rule templates used in a lexical rule learner are [9]:

1. Change the most likely tag to Y if the character Z appears anywhere in the word.
2. Change the most likely tag to Y if the current word has suffix S , $S \leq 10$.
3. Change the most likely tag to Y if adding /deleting the suffix S , $|S| \leq 10$ results in word.
4. Change the most likely tag to Y if word W ever appears immediately to the left/right of the word.
5. Change the most likely tag from X to Y if the character Z appears anywhere in the word.
6. Change the most likely tag from X to Y if the current word has suffix S , $S \leq 10$
7. Change the most likely tag from X to Y if adding/deleting the suffix S , $|S| \leq 10$ results a word.

8. Change the most likely tag from X to Y if the word W ever appears immediately to the left/right of the word.

The rule template listed 1-4 is applied as a new without considering the current tag's existence. But in the templates listed 5-8 they change the tag from the current tag to a new tag.

Threshold value x is given to restrict the use of the template to n words occurring most frequently in unannotated corpus to increase computation speed.

4.7.1.2.2 Contextual Rule Learner

The contextual rule learner module is the sub-class of the learning phase in transformation-based error-driven learning [45]. It uses the context or environment of a word to make the most likely tag prediction for a word. The manually tagged corpus is divided into two and the lexical rule learner uses the first half and the second half is used by the contextual rule learner [45].

The contextual rule learner module accepts both the output of the initial state tagger or temporary corpus and the goal corpus to generate contextual rules [9]. The goal corpus is used as a reference for comparison. Similar to the lexical rule learner, the contextual rule learner module produces a set of possible rules from pre-specified contextual rule templates. The rule generation process continues until no rule can change or improve the tagging.

The contextual rules are learned for disambiguation and provide better accuracy in tagging unknown words

A set of rule templates used by contextual rule learners are: adopted from [9].

1. The preceding/following word is tagged x .
2. The word two before/after is tagged x .
3. One of the two preceding/following words is tagged x .
4. One of the three preceding/following words is tagged x .
5. The preceding word is tagged x and the following word is tagged y .
6. The preceding/following word is tagged x and the word two before/after is tagged y .

For every rule applied on a particular word the contextual rule, the learner computes its score and the rule with the highest score will be stored in the rule subcomponent for later use in tagging.

$$\text{Score}(R) = \text{number of errors corrected} - \text{numbers of errors introduced}$$

The score value indicates the impact of the rule on the tag. The score value +1 shows the applied rule has corrected the tag of the word. The -1 value of the score shows the applied rule has not corrected the tag of the word and the zero (0) value of the score shows the rule does not affect the tag of the word.

4.7.1.2.3 Rules

The rule component is a sub-component of transformation-based error-driven learning. The rules are instantiated from the set of pre-specified rule templates. A rule consists of two parts: triggers (condition or current tag) and the resulting tag [9].

The rules are generated by the lexical rule learner and contextual rule learner and stored in the rule component. The Rule-based component stores the rules in a file and the rules can be identified as lexical or contextual depending on the information content of the rule [9]. The rules can be in the form of:

If the trigger, then change the tag X to tag Y

Or

If trigger, then change the tag to the tag y,

Where x and y are variables.

In the first template, the rule changes the current tag to the resulting tag if the trigger or the condition has been satisfied. In the second template if the trigger or condition is satisfied regardless of the current tag change the tag of the word to the resulting tag.

4.7.2 Brill Tagger Architecture

Brills Rule-based tagger is adapted from a work of Brill [9]. It takes the untagged Afaan Oromoo text and applies the rules learned by transformation-based rule-learner to tag the words with their appropriate POS tag. Brill tagger which is used in the study illustrated in Figure 4-6.

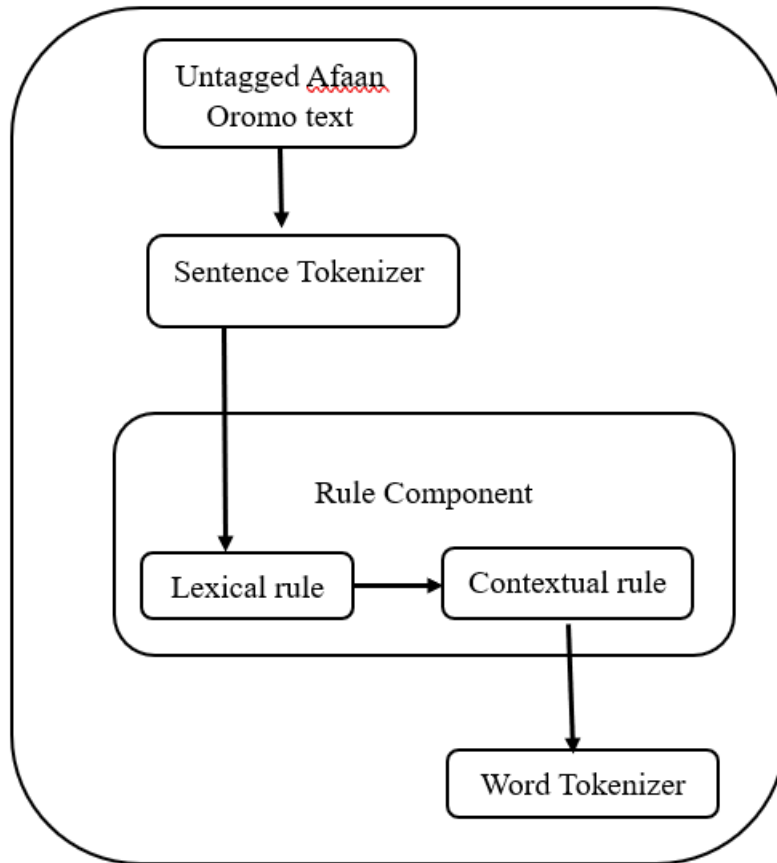


Figure 4-6. Rule based tagger model

The text tagged by a Rule-based model is taken as input by the performance analyzer then the performance analyzer compares the text tagged by a Rule-based model with the other input which is manually tagged corpus to evaluate the performance of the tagger. The performance analysis rule-based model is illustrated in Figure 4-7.

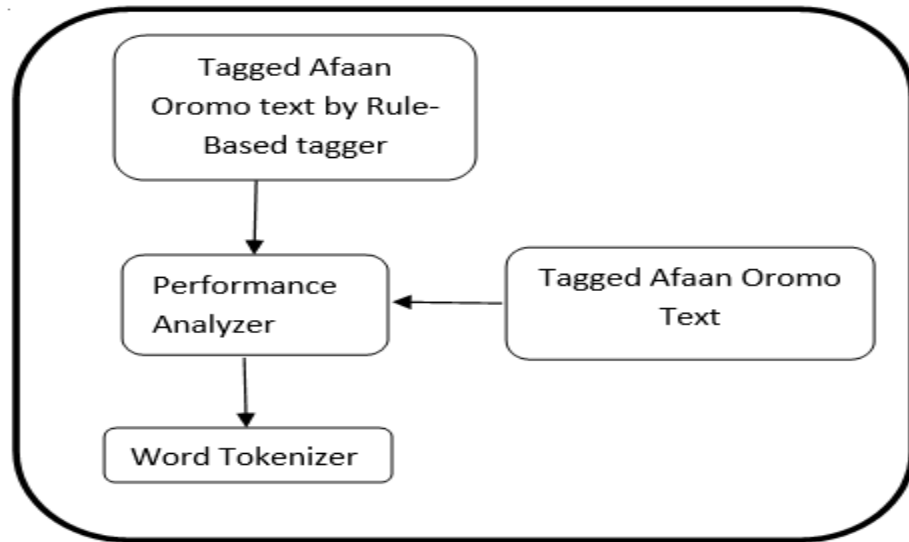


Figure 4-7. Rule based performance analyzer

4.8 Summary

To design the POS tagger system the study was implemented by combining a Hidden Markov Model and a Rule-based approach to work a single hybrid approach. NLTK modules and python are used as implementation tools. As supervised learning, it uses a pre-tagged corpus and untagged corpus. The pre-tagged corpus is used to train and as a reference to test the performance of the tagger. The untagged corpus is used to test the tagger model.

A Hidden Markov Model (HMM) is the most common statistical approach widely used in speech and language processing. An HMM is used to find the most optimal tag sequences for a given sequence of words. It deals with both observed and hidden events of the part-of-speech tagging problem. The HMM uses the Viterbi algorithm to identify the sequence of tags for sequences of words.

A Rule-based tagger tags sequences of words using the rules generated in the learning phase. The rules are generated from a set of rule templates during rule learning. Rules are called Brill transformations.

CHAPTER FIVE

5 EXPERIMENTS AND PERFORMANCE ANALYSIS

5.1 Introduction

This chapter mainly presents experimental results of each tagger model are discussed briefly, the experiments carried out by the HMM taggers and Rule-based taggers, and finally a summary of the chapter.

5.2 Performance Evaluation Metrics

The performance of the HMM tagger and Rule-based tagger has been evaluated using standardized metrics: i.e. Accuracy and precision [79].

Accuracy - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations [79].

$$\text{Accuracy} = \frac{\text{Number of correctly tagged tokens}}{\text{Total tagged Tokens}}$$

Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations which means when the tagger predicts tag T, how often is it correct in the dataset [79].

$$\text{Precision} = \frac{\text{Number of Correctly Tagged Tokens returned by Tagger}}{\text{Total Number of Tagged Tokens as returned by Tagger}}$$

5.3 Test Result of HMM Tagger

To conduct the experiment the corpus that was collected from different sources is divided into training set and test set. The tagger was trained on 80% of the collected corpus and tested by using the rest 20% of the data. The NLTK HMM tagger modules were used with some modifications based on the nature of the language for conducting experiments on the HMM tagger. The words that are not present in the training set but which appear in the test set are tagged as nouns (NN). The performance of the tagger is improved in each tagger models unigram tagger obtained an accuracy of 87.3% and tagging precision 88% is obtained while using bigram tagger an accuracy of 88.4% and tagging precision 89.5% and while using trigram tagger the accuracy of 89.3% and tagging precision 90.6%, is obtained.

Table 5-1. Test result of HMM taggers

NO	HMM Taggers	Accuracy	Precision
1	Unigram Tagger	87.3	88
2	Bigram Tagger	88.4	89.5
3	Trigram tagger	89.3	90.6

5.4 Test Result of Brill Tagger

The training of the taggers was conducted with 80% of collected data and the rest 20% was used to test the performance of the tagger. The experimental results show the tagging accuracy of the tagger and the tagging precision of the tagger. The NLTK brill rule-based tagger modules were used with some modifications based on the nature of the language for conducting experiments on the Rule-based tagger. The words that are not present in the training set but which appear in the test set are tagged as nouns (NN). The rule-based tagger has used unigram, bigram, and trigram taggers as initial stage taggers. The performance of the tagger is improved in each tagger model while using unigram tagger as initial stage tagger the accuracy 88.6%, and precision 90.2% is obtained while using bigram tagger as initial stage tagger the accuracy 89.3% and precision 90.9% is obtained and while using trigram tagger as initial stage tagger the accuracy 89.9% and tagging precision 91.3% is obtained.

Table 5-2. Test Result of Rule-based taggers

NO	Brill Taggers	Accuracy	Precision
1	Unigram Tagger	88.6	90.2
2	Bigram Tagger	89.3	90.9
3	Trigram tagger	89.9	91.3

5.5 Experimental Analysis

To analyze the performance of an HMM tagger and Rule-based tagger for the identified part of speech tags, the frequency of the tagsets in the entire corpus, training set, and testing set is analyzed and Also the confusion matrix has been developed for the taggers. In this study, 27 tagsets are identified and used to tag words. The tagsets were classified into two categories for ease of presentation, 20 tagsets which are the basic tag sets and tags derived from basic tags with high frequency, and the rest of the tagsets as others. The basic tagsets and high frequency derived tag category include tags (AD, AJ, AJP, CC, CN, NC, NN, NP,

NV, NZ, ON, PC, PD, PI, PP, PPO, PU, VC, VP and VV) and the other category includes tags (NPO, VNE, AJC, PR, PPR, IN and NG).

5.5.1 Experimental Analysis of HMM Taggers

The performance of HMM taggers (unigram, bigram, and trigram taggers) is analyzed using the confusion matrix and presented in the tables below. A confusion matrix tabulates all the mistakes committed by a tagger in the form of a matrix $C[t_i, t_j]$. $C[t_i, t_j]$ counts the number of times the tagger predicted t_i instead of t_{ij} .

As indicated in Table 5-3, the confusion matrix of an HMM unigram tagger shows that the tagger has tagged 5413 correctly and 790 tags incorrectly. The tagger has confused the tags with different parts-of-speech tags. The performance of the HMM unigram-based tagger varies from one parts-of-speech tag to another part-of-speech tag. The below analysis shows that out of 257 AD in the test set only 228 or 88.72% is correctly tagged as AD and the rest 29 or 11.28% are confused to a different part of speech tags, out of 284 AJ in the test set only 243 or 85.56% is correctly tagged as AJ and the rest 41 or 14.44% are confused to a different part of speech tags, out of 1438 NN in the test set only 1381 or 96.04% is correctly tagged as NN and the rest 51 or 3.06% are confused to a different part of speech tags, out of 310 PPO in the test set only 61 or 19.68% is correctly tagged as PPO and the rest 249 or 70.42% are confused to a different part of speech tags and out of 1272 VV in the test set only 1113 or 87.50% is correctly tagged as VV and the rest 159 or 12.50% are confused to a different part of speech tags.

As indicated in Table 5-4, the confusion matrix of an HMM bigram tagger shows that the tagger has tagged 5481 correctly and 772 tags were tagged incorrectly. The tagger has confused the tags with different parts-of-speech tags. The performance of the HMM bigram-based tagger varies from one part-of-speech tag to another part-of-speech tag. The below analysis shows that out of 257 AD in the test set only 230 or 89.49% is correctly tagged as AD and the rest 27 or 10.51% are confused to a different part of speech tags, out of 284 AJ in the test set only 253 or 89.08% is correctly tagged as AJ and the rest 41 or 10.92% are confused to a different part of speech tags, out of 1438 NN in the test set only 1404 or 97.64% is correctly tagged as NN and the rest 34 or 2.36% are confused to a different part of speech tags, out of 310 PPO in the test set only 52 or 26.77% is correctly tagged as PPO and the rest 258 or 73.33% are confused to a different part of speech tags

and out of 1272 VV in the test set only 1133 or 89.07% is correctly tagged as VV and the rest 139 or 10.93% are confused to a different part of speech tags.

Table 5-3. Experimental analysis of HMM unigram tagger

		Predicted Label																						Total	100%
		AD	AJ	AJP	CC	CN	NC	NN	NP	NV	NZ	ON	PC	PD	PI	PP	PPO	PU	VC	VP	VV	Others			
True Label	AD	228	12	0	1	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	4	0	257	88.72	
	AJ	4	243	2	0	7	0	13	1	0	2	0	0	0	3	7	0	0	0	0	2	0	284	85.56	
	AJP	0	0	38	0	0	0	4	1	0	0	0	0	0	0	2	0	0	0	0	0	0	1	46	82.61
	CC	0	0	0	88	0	0	1	0	0	0	0	0	0	0	3	0	0	0	0	1	0	93	94.62	
	CN	0	1	0	1	292	0	2	0	0	1	0	0	0	8	0	0	0	0	0	0	0	0	305	95.74
	NC	0	0	0	0	0	43	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	55	78.18
	NN	1	6	0	9	3	1	1381	9	0	14	0	0	0	1	3	0	0	0	1	7	2	1438	96.04	
	NP	0	0	0	0	0	0	41	247	0	0	0	0	0	0	0	0	0	0	0	3	0	0	291	84.88
	NV	0	0	0	0	0	1	8	1	23	2	0	0	0	0	0	0	0	0	0	0	0	0	35	65.71
	NZ	0	11	0	0	0	0	27	3	0	176	0	0	0	0	0	0	0	0	0	0	0	0	217	81.11
	ON	0	0	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0	0	0	0	0	0	23	95.65
	PC	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	9	100.00
	PD	1	0	0	0	0	0	2	0	0	0	0	1	83	0	1	0	0	0	0	0	0	0	88	94.32
	PI	1	0	0	0	0	0	3	0	0	0	0	0	0	60	1	0	0	0	0	0	0	0	65	92.31
	PP	2	0	0	0	0	0	13	0	0	0	0	0	1	0	515	26	0	0	0	0	0	0	557	92.46
	PPO	0	1	2	0	0	0	232	0	0	0	0	0	0	0	13	61	0	0	0	0	1	310	19.68	
	PU	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	608	0	0	0	0	0	609	99.84
	VC	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	23	52.17
	VP	0	0	1	0	0	0	36	6	0	0	0	0	0	0	0	0	0	0	0	153	7	0	203	75.37
	VV	1	9	0	3	1	0	106	6	8	6	0	0	0	0	0	0	0	0	18	1113	1	1272	87.50	
Others	0	0	1	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	1	18	23	78.26		
Total	238	283	44	102	304	45	1906	275	31	201	22	10	84	72	545	88	608	12	175	1135	23	6203	100.00		

Table 5-4. Experimental analysis of HMM bigram tagger

		Predicted Label																						Total	100%
		AD	AJ	AJP	CC	CN	NC	NN	NP	NV	NZ	ON	PC	PD	PI	PP	PPO	PU	VC	VP	VV	Others			
True Label	AD	230	9	0	1	0	0	14	0	0	0	0	0	0	0	1	0	0	0	0	2	0	257	89.49	
	AJ	3	253	3	0	2	0	13	1	0	2	0	0	0	2	5	0	0	0	0	0	0	284	89.08	
	AJP	0	0	39	0	0	0	4	0	0	0	0	0	0	0	2	0	0	0	0	0	1	46	84.78	
	CC	0	0	0	89	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	1	93	95.70	
	CN	0	1	0	1	292	0	2	0	0	1	0	0	0	7	1	0	0	0	0	0	0	305	95.74	
	NC	0	0	0	0	0	43	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	55	78.18	
	NN	1	5	0	6	1	1	1404	5	0	9	0	0	0	0	0	0	0	0	0	4	2	1438	97.64	
	NP	0	0	0	0	0	0	37	251	0	0	0	0	0	0	0	0	0	0	2	1	0	291	86.25	
	NV	0	0	0	0	0	1	8	0	25	1	0	0	0	0	0	0	0	0	0	0	0	35	71.43	
	NZ	0	11	0	0	0	0	30	3	0	173	0	0	0	0	0	0	0	0	0	0	0	217	79.72	
	ON	0	0	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0	0	0	0	0	23	95.65	
	PC	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	9	100.00	
	PD	1	0	0	0	0	0	2	0	0	0	0	1	83	0	1	0	0	0	0	0	0	88	94.32	
	PI	1	0	0	0	0	0	3	0	0	0	0	0	0	60	1	0	0	0	0	0	0	65	92.31	
	PP	1	1	0	0	0	0	11	0	0	0	0	0	1	0	528	15	0	0	0	0	0	557	94.79	
	PPO	0	1	0	0	0	0	234	0	0	0	0	0	0	0	22	52	0	0	0	0	1	310	16.77	
	PU	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	608	0	0	0	0	609	99.84	
	VC	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	12	0	0	0	23	52.17	
	VP	0	0	1	0	0	0	36	7	0	0	0	0	0	0	0	0	0	0	157	2	0	203	77.34	
	VV	1	4	0	2	0	0	100	5	4	5	0	0	0	0	0	0	0	0	17	1133	1	1272	89.07	
Others	0	0	1	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	1	18	23	78.26		
Total	238	285	44	99	296	45	1925	273	29	191	22	10	84	69	562	68	608	12	176	1144	23	6203	100.00		

Table 5-5. Experimental analysis of HMM trigram tagger

		Predicted Label																						Total	100%
		AD	AJ	AJP	CC	CN	NC	NN	NP	NV	NZ	ON	PC	PD	PI	PP	PPO	PU	VC	VP	VV	Others			
True Label	AD	236	6	0	1	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	257	91.83	
	AJ	3	259	1	0	1	0	13	0	0	3	0	0	0	0	4	0	0	0	0	0	0	284	91.20	
	AJP	0	0	41	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	46	89.13	
	CC	0	0	0	90	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	0	93	96.77	
	CN	0	1	0	1	293	0	2	0	0	1	0	0	0	7	0	0	0	0	0	0	0	305	96.07	
	NC	0	0	0	0	0	43	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	55	78.18	
	NN	1	0	0	7	0	0	1415	4	0	7	0	0	0	0	0	0	0	0	0	2	2	1438	98.40	
	NP	0	0	0	0	0	0	37	254	0	0	0	0	0	0	0	0	0	0	0	0	0	291	87.29	
	NV	0	0	0	0	0	0	8	0	26	1	0	0	0	0	0	0	0	0	0	0	0	35	74.29	
	NZ	0	11	0	0	0	0	30	4	0	172	0	0	0	0	0	0	0	0	0	0	0	217	79.26	
	ON	0	0	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0	0	0	0	0	23	95.65	
	PC	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	9	100.00	
	PD	1	0	0	0	0	0	2	0	0	0	0	1	83	0	1	0	0	0	0	0	0	88	94.32	
	PI	1	0	0	0	0	0	3	0	0	0	0	0	0	60	1	0	0	0	0	0	0	65	92.31	
	PP	1	1	1	0	0	0	11	0	0	0	0	0	1	0	529	13	0	0	0	0	0	557	94.97	
	PPO	0	1	0	0	0	0	234	0	0	0	0	0	0	0	9	65	0	0	0	0	1	310	20.97	
	PU	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	608	0	0	0	0	609	99.84	
	VC	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	12	0	0	0	23	52.17	
	VP	0	0	1	0	0	0	36	6	0	0	0	0	0	0	0	0	0	0	159	1	0	203	78.33	
	VV	2	4	0	1	0	0	97	5	3	5	0	0	0	0	0	0	0	0	12	1142	1	1272	89.78	
Others	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	0	20	23	86.96		
Total	245	283	44	100	295	43	1930	274	29	189	22	10	84	67	545	79	608	12	171	1148	25	6203	100.00		

As indicated in Table 5-5, the confusion matrix of an HMM trigram tagger shows that the tagger has tagged 5538 correctly and 665 tags were tagged incorrectly. The tagger has confused the tags with different parts-of-speech tags. The performance of the HMM trigram-based tagger varies from one part-of-speech tag to another part-of-speech tag. The above analysis shows that out of 257 AD in the test set only 236 or 91.83% is correctly tagged as AD and the rest 21 or 8.72% are confused to a different part of speech tags, out of 284 AJ in the test set only 259 or 91.20% is correctly tagged as AJ and the rest 25 or 9.8% are confused to a different part of speech tags, out of 1438 NN in the test set only 1415 or 98.40% is correctly tagged as NN and the rest 23 or 1.6% are confused to a different part of speech tags, out of 310 PPO in the test set only 65 or 20.97% is correctly tagged as PPO and the rest 245 or 79.03% are confused to a different part of speech tags and out of 1272 VV in the test set only 1142 or 89.78% is correctly tagged as VV and the rest 130 or 10.22% are confused to a different part of speech tags.

1.1.1. Experimental Analysis of Rule-based Taggers

The performance of Rule-based taggers (unigram, bigram, and trigram taggers) is analyzed using the confusion matrix and presented in the tables below.

As indicated in Table 5-6, the confusion matrix of a Rule-based unigram tagger shows that the tagger has tagged 5493 correctly and 710 tags incorrectly. The tagger has confused the tags with different parts of speech tags. The performance of the Rule-based unigram tagger varies from one part-of-speech tag to another part-of-speech tag. The below analysis shows that out of 257 AD in the test set only 231 or 89.88% is correctly tagged as AD and the rest 26 or 10.22% are confused to a different part of speech tags, out of 284 AJ in the test set only 256 or 90.14% is correctly tagged as AJ and the rest 28 or 9.86% are confused to a different part of speech tags, out of 1438 NN in the test set only 1397 or 97.15% is correctly tagged as NN and the rest 41 or 2.85% are confused to a different part of speech tags, out of 310 PPO in the test set only 68 or 21.94% is correctly tagged as PPO and the rest 242 or 78.16% are confused to a different part of speech tags and out of 1273 VV in the test set only 1126 or 88.45% is correctly tagged as VV and the rest 147 or 11.55% are confused to a different part of speech tags.

Table 5-6. Experimental analysis of Rule-based unigram tagger

		PREDICTED TABLE																						Total	100%
		AD	AJ	AJP	CC	CN	NC	NN	NP	NV	NZ	ON	PC	PD	PI	PP	PPO	PU	VC	VP	VV	Others			
True Table	AD	231	6	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	5	0	257	89.88	
	AJ	4	256	2	0	3	0	13	1	0	2	0	0	0	0	1	0	0	0	0	2	0	284	90.14	
	AJP	0	0	42	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	46	91.30	
	CC	0	0	0	90	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	93	96.77	
	CN	0	1	0	0	294	0	2	0	0	1	0	0	0	7	0	0	0	0	0	0	0	305	96.39	
	NC	0	0	0	0	0	43	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	55	78.18	
	NN	1	3	0	5	2	0	1397	5	0	11	0	0	0	1	3	0	0	0	1	7	2	1438	97.15	
	NP	1	0	0	0	0	0	41	247	0	0	0	0	0	0	0	0	0	0	0	2	0	291	84.88	
	NV	0	0	0	0	0	1	8	1	23	2	0	0	0	0	0	0	0	0	0	0	0	35	65.71	
	NZ	0	11	0	0	0	0	27	3	0	176	0	0	0	0	0	0	0	0	0	0	1	218	80.73	
	ON	0	0	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0	0	0	0	0	23	95.65	
	PC	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	9	100.00	
	PD	1	0	0	0	0	0	2	0	0	0	0	1	83	0	1	0	0	0	0	0	0	88	94.32	
	PI	1	0	0	0	0	0	3	0	0	0	0	0	0	60	1	0	0	0	0	0	0	65	92.31	
	PP	2	0	0	0	0	0	11	0	0	0	0	0	1	0	535	7	0	0	0	0	1	557	96.05	
	PPO	0	1	0	0	0	0	234	0	0	0	0	0	0	0	7	68	0	0	0	0	0	310	21.94	
	PU	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	608	0	0	0	0	609	99.84	
	VC	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	12	0	0	0	23	52.17	
	VP	0	0	1	0	0	0	36	6	0	0	0	0	0	0	0	0	0	0	154	6	0	203	75.86	
	VV	0	8	0	1	0	0	106	6	3	5	0	0	0	0	0	0	0	0	17	1126	1	1273	88.45	
Others	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	1	17	21	80.95		
Total	241	286	45	96	300	44	1925	270	26	197	22	10	84	68	550	75	608	12	172	1150	22	6203	100.00		

Table 5-7. Experimental analysis of Rule-based bigram tagger

		Predicted Label																						Total	100%
		AD	AJ	AJP	CC	CN	NC	NN	NP	NV	NZ	ON	PC	PD	PI	PP	PPO	PU	VC	VP	VV	Others			
True Label	AD	234	6	0	0	0	0	14	0	0	0	0	0	0	1	0	0	0	0	2	0	257	91.05		
	AJ	3	260	2	0	1	0	13	1	0	2	0	0	1	1	0	0	0	0	0	0	284	91.55		
	AJP	0	0	41	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	45	91.11		
	CC	0	0	0	90	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	93	96.77		
	CN	0	1	0	0	293	0	2	0	0	1	0	0	0	7	1	0	0	0	0	0	305	96.07		
	NC	0	0	0	0	0	43	11	1	0	0	0	0	0	0	0	0	0	0	0	0	55	78.18		
	NN	1	3	0	2	0	0	1413	5	0	8	0	0	0	0	0	0	0	0	0	4	2	1438	98.26	
	NP	0	0	0	0	0	0	37	251	0	1	0	0	0	0	0	0	0	0	1	1	0	291	86.25	
	NV	0	0	0	0	0	0	8	0	26	1	0	0	0	0	0	0	0	0	0	0	0	35	74.29	
	NZ	0	11	0	0	0	0	30	2	0	174	0	0	0	0	0	0	0	0	0	0	0	217	80.18	
	ON	0	0	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0	0	0	0	0	23	95.65	
	PC	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	9	100.00	
	PD	1	0	0	0	0	0	2	0	0	0	0	1	83	0	1	0	0	0	0	0	0	88	94.32	
	PI	1	0	0	0	0	0	3	0	0	0	0	0	0	60	1	0	0	0	0	0	0	65	92.31	
	PP	1	0	0	0	0	0	11	0	0	0	0	0	1	0	534	9	0	0	0	0	1	557	95.87	
	PPO	0	1	0	0	0	0	234	0	0	0	0	0	0	0	7	68	0	0	0	0	0	310	21.94	
	PU	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	608	0	0	0	0	609	99.84	
	VC	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	12	0	0	0	23	52.17	
	VP	0	0	1	0	0	0	36	7	0	0	0	0	0	0	0	0	0	0	157	2	0	203	77.34	
	VV	0	3	0	2	0	0	97	5	2	5	0	0	0	0	0	0	0	0	17	1141	1	1273	89.63	
Others	2	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	19	23	100.00		
Total	243	285	44	94	296	43	1928	272	28	192	22	10	84	68	541	77	608	12	175	1151	27	6203	100.00		

As indicated in Table 5-7, the confusion matrix of a Rule-based bigram tagger shows that the tagger has tagged 5538 correctly and 665 taggers incorrectly. The tagger has confused the tags with different parts-of-speech tags. The performance of the Rule-based bigram tagger varies from one part-of-speech tag to another part-of-speech tag. The above analysis shows that out of 257 AD in the test set only 234 or 91.05% is correctly tagged as AD and the rest 23 or 8.95% are confused to a different part of speech tags, out of 284 AJ in the test set only 260 or 91.55% is correctly tagged as AJ and the rest 24 or 8.45% are confused to a different part of speech tags, out of 1438 NN in the test set only 1413 or 98.26% is correctly tagged as NN and the rest 25 or 1.84% are confused to a different part of speech tags, out of 310 PPO in the test set only 68 or 21.94% is correctly tagged as PPO and the rest 242 or 78.16% are confused to a different part of speech tags and out of 1272 VV in the test set only 1141 or 89.63% is correctly tagged as VV and the rest 131 or 9.37% are confused to a different part of speech tags.

As indicated in Table 5-8, the confusion matrix of a Rule-based trigram tagger shows that the tagger has tagged 5572 correctly and 631 tags were tagged incorrectly. The tagger has confused the tags with different parts-of-speech tags. The performance of the Rule-based trigram tagger varies from one part-of-speech tag to another part-of-speech tag. The below analysis shows that out of 257 AD in the test set only 237 or 92.22% is correctly tagged as AD and the rest 20 or 7.8% are confused to a different part of speech tags, out of 284 AJ in the test set only 264 or 92.96% is correctly tagged as AJ and the rest 20 or 7.4% are confused to a different part of speech tags, out of 1438 NN in the test set only 1423 or 98.96% is correctly tagged as NN and the rest 10 or 1.1% are confused to a different part of speech tags, out of 310 PPO in the test set only 70 or 22.58% is correctly tagged as PPO and the rest 240 or 77.42% are confused to a different part of speech tags and out of 1272 VV in the test set only 1149 or 90.33% is correctly tagged as VV and the rest 123 or 9.7% are confused to a different part of speech tags.

Table 5-8. Experimental analysis of Rule-based trigram tagger

		Predicted label																						Total	100%
		AD	AJ	AJP	CC	CN	NC	NN	NP	NV	NZ	ON	PC	PD	PI	PP	PPO	PU	VC	VP	VV	Others			
True label	AD	237	5	0	1	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	257	92.22	
	AJ	3	264	1	0	0	0	13	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	284	92.96
	AJP	0	0	42	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	46	91.30
	CC	0	0	0	91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	93	97.85
	CN	0	0	0	0	295	0	2	0	0	1	0	0	0	7	0	0	0	0	0	0	0	0	305	96.72
	NC	0	0	0	0	0	43	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	55	78.18
	NN	1	0	0	2	0	0	1423	4	0	4	0	0	0	0	0	0	0	0	0	2	2	1438	98.96	
	NP	0	0	0	0	0	0	37	254	0	0	0	0	0	0	0	0	0	0	0	0	0	0	291	87.29
	NV	0	0	0	0	0	0	8	0	26	1	0	0	0	0	0	0	0	0	0	0	0	0	35	74.29
	NZ	0	11	0	0	0	0	30	2	0	174	0	0	0	0	0	0	0	0	0	0	0	0	217	80.18
	ON	0	0	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0	0	0	0	0	0	23	95.65
	PC	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	9	100.00
	PD	1	0	0	0	0	0	2	0	0	0	0	1	83	0	1	0	0	0	0	0	0	0	88	94.32
	PI	1	0	0	0	0	0	3	0	0	0	0	0	0	60	1	0	0	0	0	0	0	0	65	92.31
	PP	1	0	0	0	0	0	11	0	0	0	0	0	1	0	531	12	0	0	0	0	1	557	95.33	
	PPO	0	1	0	0	0	0	234	0	0	0	0	0	0	0	5	70	0	0	0	0	0	0	310	22.58
	PU	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	608	0	0	0	0	0	609	99.84
	VC	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	23	52.17
	VP	0	0	0	0	0	0	36	6	0	0	0	0	0	0	0	0	0	0	0	160	1	0	203	78.82
	VV	1	3	0	1	0	0	93	5	2	5	0	0	0	0	0	0	0	0	12	1149	1	1272	90.33	
Others	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	1	19	23	82.61		
Total	245	284	43	95	296	43	1934	272	28	188	22	10	84	67	539	82	608	12	172	1156	23	6203	100.00		

5.6 Discussions

In Afaan Oromo language syntactic and semantic attributes are ambiguous in most cases and these ambiguities can be solved by using automated Natural Language Processing applications. As far as the knowledge of the researcher is concerned there is no effort has been made to develop such automated Natural Language Processing applications for Afaan Oromo language. The parts of speech tagger is a basic subcomponent and precondition to many Natural Language Processing applications so the absence of parts of speech tagger limits the development of other Natural Language Processing applications and linguistic usability of Afaan Oromo language in the digitized world.

Besides the limitations of literature, readily available corpus, readily available word classes, and shortage of linguistics in the area, this study has provided a solution to those problems by developing a part-of-speech tagger model for Afaan Oromo language. The development of part of speech tagger for Afaan Oromo language was achieved by using both the HMM and rule-based models.

The taggers tag a sentence or a word provided to the tagger with their appropriate part of speech tags. Since the HMM taggers try to find the most probable path for a given sequence of words, there is no defined pattern of confusion like that of the rule-based tagger; rather the confusion is distributed across most parts of speech tags. It is thought that the lack of a large standard training corpus caused the HMM tagger to score less than the rule-based tagger. The number of training data affects the rule-based tagger less than the HMM tagger. In the future using large balanced data for training, large tagsets, and developing combination or hybrid taggers will improve the accuracy of the taggers and wide-range use of the taggers.

CHAPTER SIX

6 CONCLUSION AND RECOMMENDATION

6.1 Conclusion

Natural language processing (NLP) is a field of study that deals with computing natural language with a machine and a way for computers to analyze, understand, and derive meaning from human language in a smart and useful way. NLP can be used in organizing and structuring knowledge to perform automatic text summarization, named entity recognition, word sense disambiguation, grammar checker, sentiment analysis, part-of-speech tagging, relationship extraction, question answering, Topic extraction, and stemming.

Part-Of-Speech (POS) tagging is an application of NLP whose task is labeling or tagging each word in a sentence with its appropriate word classes to get the correct sense of the word from in the context. It is a precondition or subcomponent for most natural language processing tasks rather than being an application to stand by itself. The problem of assigning a word to their appropriate word classes can be solved by different approaches. Some of the most commonly used approaches are rule-based, HMM, artificial neural network, memory-based, and a combination of individual approaches.

The corpus has been the vital thing while working with NLP applications and the amount of the corpus makes a difference in the performance of the applications. Well organized and large amount of balanced corpus can produce better performance than the category-specific corpus. To conduct this study the corpus has been collected from different sources, from BBC Afaan Oromo, VOA Afaan Oromo and Afaan New Testament bible. The corpus comprises 1346 sentences which are composed of a total of 30,165 words in which the 8366 words uniquely appear.

In this study, 27 tagsets are selected with the help of linguists and used to tag the words in the training corpus. The selected tagset only indicates word-class rather than gender, number, tenses, negation, classes, etc.

In this thesis NLTK 3.4.5 and python 2.7 are used to implement and test the performance of both the HMM and rule-based part-of-speech taggers for the Afaan Oromo language. While performing experimental analysis, different test results were obtained by both Rule

based and HMM taggers. As a result of analysis, a tagging accuracy of 87.3%, 88.4% and 89.3, and tagging precision of 88%, 89.5%, 90.6% obtained by HMM taggers unigram, bigram, and trigram taggers respectively and of 88.6%, 89.3% and 89.9% and tagging precision of 90.2%, 90.9% and 91.3% obtained by rule-based taggers unigram, bigram, and trigram taggers respectively. Based on the experimental result the performance level of the rule-based taggers is somehow better than the HMM taggers.

6.2 Recommendation

The researchers have collected the corpus from two different sources and identified the tagsets with the help of linguistics to tag the words in the corpus. But there are advantages of using the tagger as a pre-component for other NLP applications, limitations, and gaps which can open doors for the other NLP researches for Afaan Oromo language.

Therefore, the researchers suggest the following future research directions:

- This research work can be expanded by using large balanced data for training and large tagsets which can be used to identify gender, tense, negation, classes, etc.
- Conducting similar researches which work by identifying different dialects in the Afaan Oromo language.
- By combining two approaches or a hybrid approach that can give better results than the individual taggers by getting more accurate results and handling rare words and unknown words.
- Making comparative study by using other approaches rather than rule-based and HMM that is used in this research.
- A similar approach and method can be used for developing a part-of-speech tagger for other local languages such as Amahric, Afar, Somali, etc.

References

- [1] H. Hermann, Knowledge representation and the semantics of natural language, Springer-Verlag Berlin Heidelberg, Germany, 2006.
- [2] M. Christopher and S. Hinrich, Foundations of Statistical Natural Language Processing, London: The MIT Press cambridge,, 1999.
- [3] G. Mamo, *Part of Speech Tagging for Afaan Oromo Language*, MSc Thesis, Addis Ababa: Addis Ababa University, 2009.
- [4] Algorithmia, "Algorithmia," 11 AUGUST 2016. [Online]. Available: <https://algorithmia.com/blog/introduction-natural-language-processing-nlp>. [Accessed 10 DECEMBER 2019].
- [5] G. Kebede, *The Application of Decision Tree for Part of Speech (POS) Tagging for Amaharic*, MSc Thesis, Addis Abeba: Addis Abeba University, 2009.
- [6] A. Ratnaparkhi, "A Maximum Entropy Model for Part-Of-Speech Tagging," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1996.
- [7] D. Jain and V. Kumawat , "POS Tagging Approaches: A Comparison," *International Journal of Computer Applications*, vol. 118, no. 6, pp. 32 - 38, 2015.
- [8] T. Gebregzabiher, *Part Of Speech Tagger Tor Tigrigna Language*, MSc Thesis, Addis Abeba: Addis Abeba University, 2010.
- [9] E. Brill, "Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-Of-Speech Tagging," *Associations for Computational Linguistics*, vol. 21, no. 4, pp. 553 - 565, 1995.
- [10] J. Singh, N. Joshi and I. Mathur, "Development of Marathi POS Tagger Using Statistical Approach," in *International Conference on Advance in Computing, Communication and Informatics*, 2013.
- [11] L. V. Guilder, "Automated part of speech tagging: a brief overview," Georgetown University, 1995.

- [12] Z. Mekuria, *Design and Development of POS Tagger for Kafi-Noonoo language*, Addis Abeba: Addis Abeba University, 2013.
- [13] O. Gyorgy and N. Attila, "Pure Pos: An Open Source Morphological Disambiguator," in *Proceedings of the 9th International Workshop on Natural Language Processing and Cognitive Science*, 2012.
- [14] T. Brants, "TnT- A Statistical Part-Of-Speech Tagger," in *Proceedings of the Sixth Conference on Applied Natural Language Processing*, 2000.
- [15] O. Gyorgy and N. Attila, "PurePos 2.0: A Hybrid Tool For Morphological disambiguation," in *Proceedings of International Conference on Recent advances in Natural Language Processing*, 2013.
- [16] A. Gulen and S. Esin, "Part of Speech Tagging," *Middle East Technical University*, pp. 1-23, December 2001.
- [17] S. Alansary, "Part-of-Speech Tagging and Disambiguation for Arabic Language Understanding," in *The 4th Egyptian Society of Language Engineering Conference*, 2015.
- [18] M. H. Fahim , U. Naushad and K. Mumit, "Comparison of Unigram, Bigram, HMM and Brill's POS Tagging Approaches for Some South Asian Languages," in *In Proceedings of Center for Research on Bangla Language Processing*, 2007.
- [19] P. Alan , *Hidden Markov Models: A Guided Tour*, Princeton, 1988, pp. 7 - 13.
- [20] P. Blunsom, *Hidden Markov Models*, vol. 15, citeseer, 2004.
- [21] T. Scott and H. Mary, "A Second-Order Hidden Markov Model for Part-of-Speech Tagging," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, 1999.
- [22] J. Daniel and M. James, *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition.*, 2007.
- [23] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," in *Proceedings of the IEEE*, 1989.

- [24] P. Avinesh and G. Karthik, "Part-Of-Speech Tagging and Chunking Using Conditional Random Fields and Transformation Based Learning," in *Shallow Parsing for South Asian Languages*, Hyderabad, 2007.
- [25] D. Shivangi and A. Bhavna, "Preprocessing for Parts of Speech (POS) Tagging in Dorgi Language," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 8S3, pp. 114 - 119, 2019.
- [26] J. Lafferty, A. McCallum and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning*, 2001.
- [27] S. Charles and M. Andrew, "An introduction to conditional random fields for relational learning," *Introduction to statistical relational learning*, vol. 2, pp. 93 - 128, 2006.
- [28] P. Chirag and G. Karthik, "Part-Of-Speech Tagging for Gujarati Using Conditional Random," in *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, Hyderabad, 2008.
- [29] E. Asif, H. Rejwanul and B. Sivaji, "Bengali part of speech tagging using conditional random field," in *Proceedings of seventh international symposium on natural language processing*, 2007.
- [30] H. Jiawei, K. Micheline and P. Jian, *Data Mining Concepts and Techniques*, Waltham: Elsevier, 2012.
- [31] J. R. Quinlan, *C4. 5: programs for machine learning*, San Mateo, CA.: Morgan Kaufmann, 1993.
- [32] B. J. Daniel and L. S. Harold, *New Methods In Language Processing*, London: Association with the center for Computational Linguistics, 1997.
- [33] L. M. Villodre, *Part-of-speech Tagging: A Machine Learning Approach based on Decision Trees*, Barcelona, 1999.
- [34] O. Giorgos, K. Dimitris, P. Thanasis and C. Dimitris, "Decision trees and NLP: A case study in POS tagging," in *Proceedings of annual conference on artificial intelligence*, 1999.

- [35] V. Vapnik, *The Nature of Statistical Learning Theory*, Holmdel NJ: Springer-Verlag New York, Inc, 1995.
- [36] B. Stecanella, "monkeylearn," 22 June 2017. [Online]. Available: <https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/>. [Accessed 24 9 2019].
- [37] F. Sandareka, R. Surangika, J. Sanath and D. Gihan, "Comprehensive Part-Of-Speech Tag Set and SVM Based POS Tagger for Sinhala," in *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing*, Sri Lanka, 2016.
- [38] E. Asif and B. Sivaji, "Part of Speech Tagging in Bengali using Support Vector Machine," in *International Conference on Information Technology*, 2008.
- [39] A. Parikh, "Part-Of-Speech Tagging using Neural network," in *International Conference on Natural Language Processing*, Hyderabad, 2009.
- [40] B. Rosemary and . A. K. Snezana, "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research," *Journal of pharmaceutical and biomedical analysis*, vol. 22, no. 5, pp. 717 - 727, 2000.
- [41] N. Indurkha and J. F. Damerau, *Handbook of Natural Language Processing*, USA: Chapman & Hall/CRC, 2010.
- [42] H. Schmid, *Part-of-Speech Tagging with Neural Networks*, Stuttgart, 1994.
- [43] A. Ratnaparkhi, "A simple introduction to maximum entropy models for natural language processing," *IRCS Technical Reports Series*, p. 81, 1997.
- [44] G. Bjorn , O. Fredrik, A. A. Argaw and A. Lars, "Methods for Amharic Part-of-Speech Tagging," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, 2009.
- [45] A. G. Ayana, *Improving Brill's Lexical Transformation Rule For Afan Oromo Language MSc Thesis*, Addis Ababa, 2013.
- [46] A. Ratnaparkhi, "Maximum entropy models for natural language ambiguity resolution," *IRCS Technical Reports Series.*, Pennsylvania, 1998.
- [47] E. Brill, "A Simple Rule-Based Part of Speech Tagger," pp. 152 - 155, 1992.

- [48] D. Aniket , K. Nagraj, S. Uma, S. Sandeep and B. Pushpack, "Building Features Rich POS Tagger for Morphologically Rich Languages: Experiences in Hindi," in *International Conference on Natural Language*, 2007.
- [49] K. Samuel, A. Wilson, S. Carberry and K. Vijay-Shankerand, "Randomized rule selection in transformation-based learning: a comparative study," *Natural Language Engineering*, vol. 7, no. 2, pp. 99 - 116, 2001.
- [50] T. Jalaj, Python Natural Language Processing, Birmingham, Livery Place: Published by Packt Publishing Ltd., 2017.
- [51] G. Vipul, J. Rutva and P. Ekta, "A Review on Part-Of-Speech Tagging on Gujarati Language," *International Research Journal of Engineering and Technology*, pp. 3113-3120, 2019.
- [52] M. Argaw, *Amharic Parts-of-Speech Tagger using Neural Word Embeddings as Features*, Addis Ababa: Addis Ababa University, 2019.
- [53] S. F. Adafre, "Part of Speech tagging for Amharic using Conditional Random Fields," in *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, Ann Arbor, 2005.
- [54] G. Emiru, *Development of Part-of-Speech Tagger Using a Hybrid Approach*, MSc Thesis, Addis Abeba: Addis Abeba University, 2016.
- [55] B. H. Ganta, *Part-of-Speech Tagging for Wolayta Language*, Addis Abeba: Addis Abeba University, 2015.
- [56] N. V. Patil, "POS Tagging for Marathi Language using Hidden Markov Model," *International Journal of Computer Sciences and Engineering*, pp. 409-412, 2018.
- [57] A. Yajnik, "Part of Speech Tagging Using Statistical Approach for Nepali Text," *International Journal of Cognitive and Language Sciences*, pp. 76-79, 2017.
- [58] A. Anisha and C. Sunitha, "A Hybrid Parts Of Speech Tagger for Malayalam Language," in *International Conference on Advances in Computing, Communications and Informatics*, Thrissur,Kerala, 2015.
- [59] G. Vishal, P. Suman and G. Navneet , "Rule-Based Hindi Part of Speech Tagger," in *Proceedings of COLING*, Mumbai, 2012.

- [60] K. Gimpel, "Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments," in *Proceedings of the Association for Computational Linguistics: short papers*, Oregon, 2011.
- [61] M. Nidhi and M. Amit, "Part of Speech Tagging for Hindi Corpus," *International Conference on Communication Systems and Network Technologies*, pp. 554-558, 2011.
- [62] L. Yuan, "Improvement for the automatic Part-of-speech Tagging Based on Hidden Markov Model," *International Conference on Signal Processing Systems*, pp. 744-747, 2010.
- [63] I. Bedane, "The Origin of Afaan Oromo: Mother Lanugage," vol. 15, no. 12, 2015.
- [64] T. Gamta, "Qube Afaan Oromo: Reasons for Choosing the Latin Script for Developing an Oromo Alphabet," *The Journal of Oromo Studies*, no. 1, 1993.
- [65] H. Mekonnen, "Lexical Standardization in Oromo," *Masters Thesis, School of Graduate Studies, Addis Ababa*, 2002.
- [66] K. Hordofa, "Towards the Genetic Classification of the Afaan Oromoo Dialects," *Department of Linguistics, and Scandinavian Studies: The University of Oslo*, 2009.
- [67] J. Martin and J. H., *Speech and Language Processing: An introduction to natural language processing, Computational linguistics, and speech recognition.*, 2007.
- [68] D. Tesfaye, "Designing a Stemmer for Afaan Oromo Text: A Hybrid Approach," *Masters Thesis, Addis Ababa University*, 2010.
- [69] T. Debela, "Afaan Oromo Search Engine," *Masters Thesis, Department of Computer science, Addis Ababa University*, 2010.
- [70] T. Tadele, "Applying Thesaurus Based Semantic Compression For Afaan Oromo Text Retrieval," *Masters Thesis, School of Computing, Jimma University*, 2019.
- [71] A. Mideksa, "Statistical Afaan Oromo Grammar Checker," *Masters thesis, Addis Ababa University, School of Information Science*, 2015.
- [72] C. Fita, "Afaan Oromo List, Definition and Description Question Answering System," *Masters Thesis, Addis Ababa University, Department of Computer Science*, 2016.

- [73] N. Ide, "Preparation and analysis of linguistic corpora," *A Companion to digital Humanities*, vol. 27, pp. 289 - 305, 2004.
- [74] F. Oroumchian, S. Tasharofi, H. Amirii, H. Hojjat and F. Raja, *Creating a feasible corpus for Persian POS tagging*, Citeseer, 2006.
- [75] A. Anber, "Dataaspirant," 3 12 2020. [Online]. Available: <https://dataaspirant.com/cross-validation/#t-1607016614332>. [Accessed 5 8 2020].
- [76] T. Haritha, "NaadiSpeaks," 12 10 2017. [Online]. Available: <https://naadispeaks.wordpress.com/tag/train-test-split/>. [Accessed 5 8 2020].
- [77] B. Steven , E. Klein and L. Edward, *Natural language processing with Python: analyzing text with the natural language toolkit*, O'Reilly Media, Inc., 2009.
- [78] J. Perkins, *Python 3 text processing with NLTK 3 cookbook*, Packt Publishing Ltd, 2014.
- [79] N. Fareena, A. Waqas, I. B. Usama and U. M. Ehsan, "Urdu Part of Speech Tagging Using Transformation Based Error Driven Learning," *World Applied Sciences Journal*, vol. 16, no. 3, pp. 437 - 448,, 2012.
- [80] Sulabh Bhatt , Krunal Parmar and Miral Patel, "Sanskrit Tag-sets and Part-Of-Speech Tagging Methods-A Survey," *International Journal of Innovative and Emerging Research in Engineering*, vol. 2, no. 1, 2015.

Appendixes

Appendices A: Generated Lexical Rules

PP->AJ if Word:kan@[0] & Pos:CC@[-1]

CN->AJ if Word:tokko@[0] & Pos:PP@[-1]

NP->NN if Word:Gaammo@[-1]

PP->AJ if Word:akka@[0] & Word:kan@[-1]

AJ->NN if Word:hacuucaa@[-1]

VP->VV if Word:dhalatan@[-1]

CC->NN if Word:jiru@[-3,-2,-1]

V->NN if Pos:NPO@[1]

CC->NN if Pos:NC@[-2]

VC->VV if Pos:NG@[-2]

NP->NZ if Pos:AJC@[2]

PPO->PP if Pos:IN@[2]

Appendices B: Generated Contextual Rules

AD->AJ if Pos:CC@[-1] & Pos:CN@[1]

AD->NN if Pos:VP@[-1] & Pos:AJ@[1]

AD->VV if Pos:NC@[-1] & Pos:PU@[1]

AJ->AD if Pos:NP@[-1] & Pos:AD@[1]

AJ->NN if Pos:NP@[-1] & Pos:VC@[1]

AJ->NN if Pos:ON@[-1] & Pos:NN@[1]

AJ->VV if Pos:NP@[-1] & Pos:PP@[1]

AJ->VV if Pos:NZ@[-1] & Pos:CN@[1]

AJP->AD if Pos:AD@[-1] & Pos:AJ@[1]

AJP->AJC if Pos:AJ@[-1] & Pos:CN@[1]

AJP->NP if Pos:AD@[-1] & Pos:NP@[1]

CC->NN if Pos:AJ@[-1] & Pos:PP@[1]

NN->NC if Pos:PPO@[-1] & Pos:PI@[1]

NP->VP if Pos:VP@[-1] & Pos:CN@[1]

NP->VV if Pos:AD@[-1] & Pos:PU@[1]

Appendices C: Sample Untagged Corpus

hindandeenye sooruuf Daa'imman dhukkubsatani fayyan ulfiina qaama hirrisan akka deebi'u fi guddina sirri akka itti fufu danda'an isa dur nyaatan caalaa akka nyaatan jajjabessuun barbaachisa Daa'imman dhaloota gad aanaa .

qaban Haatii yoo kan vaayirasii HIV Waliin jiraatu taate daa'imman sirritti sooruuf deegarsa addaa isaan barbaachisa Aannan harma haadhaa daa'imman gad aanaa qabaniif soorata filmaata hin qabine dha Daa'imman ulfiina qaama gadi .

aanaa qaban tokko tokko harma hoodhuu waan hindandeenyeef aannan eelmamee kubbaayyaan dhuguu qabu Annisa qulqulluu beeyladaaf qaamaa dabaluuf gargaaru Bishaan dabalatee waliigalati hooriidhaaf jireenya guudina qama dabaludhaa fi sadarkaa oomishtummaa horii .

irraa barbaadamu dhugoomsuuf nyaata barbachisaan jechuudha Beeyladoonni kun osoo nyaata fooyya'aa hinargatiin gabaatti waan dhiyaataniif carraa oomisha fooni fi uumurii ni xinesa Beelladoonni nyaata qulqullina hinqabne yeroo nyaatan qalamuu .

isaan andeessisu jalaa tursa foon beeyladoo qulqullina barbaadamaa ta'e hinqabaatu Baayyinni nyaata goggogaa beeyladni tokko fudhachuu danda'u tilmaamaan dhibbantaa qaama isaanii 23 kan ta'uu danda'u dha jedhamee ni yaadama .

Haaluma kanaan hojii furdisuu keessatti oomisha jechuun dabaliinsa qaamaa fi fooyya'iinsa haala qaamaa beeylada jechuudha Kanaaf horiiwwan sagantaa furdisuu keessa galan nyaata qulqulluu jireenyaa fi qaamaa akka dabalan isaan gargaaru barbaadu Fedhiin annisaa beeyladoonni barbaadan annisaa qulqulluu lubbuu tursuuf gargaaru ; Nem ykn annisaa qulqulluu qaamaa dabaluuf gargaaru .

jedhamee ibsama Annisaa qulqulluun lubbuu tursuuf gargaaru kan hundaa'u qaamaa meetaboliikii beeyladaa irratti yoo ta'u kunis haala kanan gaditti .

Appendices D: Corpus Tagged by HMM Tagger

hindandeenye/VNE sooruuf/VV Daa'imman/VV dhukkubsatani/VV fayyan/NN
ulfiina/AJ qaama/NN hirrisan/VV akka/PP deebi'u/VV fi/CC guddina/AJ sirri/NN
akka/PP itti/PP fufu/NN danda'an/VP isa/PP dur/AD nyaatan/VV caalaa/AD akka/PP
nyaatan/VV jajjabessuun/VV barbaachisa/AD Daa'imman/VV dhaloota/NN gad/NN
aanaa/NN ./PU

qaban/AD Haatii/NN yoo/CC kan/AJ vaayirasii/NN HIV/NN Waliin/NN jiraatu/VV
taate/VV daa'imman/VP sirritti/AD sooruuf/VV deegarsa/PP addaa/AJ isaan/PP
barbaachisa/AD Aannan/AD harma/NN haadhaa/NN daa'imman/VP gad/NN aanaa/NN
qabaniif/AD soorata/NN filmaata/NN hin/NG qabine/NN dha/AD Daa'imman/VP
ulfiina/AJ qaama/NN gadi/AD ./PU

aanaa/NN qaban/AD tokko/NN tokko/AJ harma/NN hoodhuu/VV waan/PP
hindandeenyeef/VNE aannan/AD eelmamee/VV kubbaayyaan/NN dhuguu/VV qabu/AD
Annisa/NN qulqulluu/VV beeyladaaf/VV qaamaa/NN dabaluu/VV gargaaru/NN
Bishaan/NN dabalatee/VV waliigalati/NN hooriidhaaf/VV jireenya/NN guudina/AJ
qama/NN dabaludhaa/NN fi/CC sadarkaa/AJ oomishtummaa/NN horii/NN ./PU

irraa/PP barbaadamu/VV dhugoomsuuf/VV nyaata/VV barbachisaan/NN jechuudha/VV
Beeyladoonni/NN kun/PP osoo/CC nyaata/NN fooyya'aa/NN hinargatiin/VNE
gabaatti/PP waan/PP dhiyaataniif/AD carraa/NN oomisha/NN fooni/VV fi/CC uumurii/AJ
ni/PP xinesa/NN Beelladoonni/NN nyaata/NN qulqullina/AJ hinqabne/VNE yeroo/AD
nyaatan/VV qalamuu/VV ./PU

isaan/PP andeessisu/VV jalaa/PP tursa/PP foon/NN beeyladoo/NN qulqullina/AJ
barbaadamaa/NN ta'e/VV hinqabaatu/VNE Baayyinni/AJ nyaata/NN goggogaa/NN
beeyladni/NN tokko/CN fudhachuu/VV danda'u/VV tilmaamaan/NN dhibbantaa/NN
qaama/NN isaanii/PP 2/CN 3/CN kan/PP ta'uu/VV danda'u/VV dha/AD jedhamee/VV
ni/PP yaadama/NN ./PU

Haaluma/PP kanaan/PP hojii/NN furdisuu/VV keessatti/PP oomisha/NN jechuun/VV
dabaliinsa/NN qaamaa/NN fi/CC fooyya'iinsa/VV haala/AD qaamaa/NN beeylada/AJ
jechuudha/VV Kanaaf/VV horiiwwan/AJ sagantaa/NN furdisuu/VV keessa/PP galan/NN
nyaata/NN qulqulluu/VV jireenyaa/NN fi/CC qaamaa/NN akka/PP dabalani/NN isaan/NN
gargaaru/NN barbaadu/NN Fedhiin/NN annisaa/NN beeyladoonni/NN barbaadan/VV

annisaa/NN qulqulluu/VV lubbuu/VV tursuuf/VV gargaaru/NN ;/PU Nem/NN ykn/CC
annisaa/NN qulqulluu/VV qaamaa/NN dabaluu/VV gargaaru/VV ./PU
jedhamee/VV ibsama/NN Annisaa/NN qulqulluu/VV lubbuu/VV tursuuf/VV
gargaaru/NN kan/PP hundaa'u/VV qaamaa/NN meetaboliikii/NN beeyladaa/NN irratti/PP
yoo/CC ta'u/VV kunis/VV haala/AD kanan/AD gaditti/PP ./PU

Appendices D: Corpus Tagged by brill Tagger

hindandeenye/VNE sooruuf/VV Daa'imman/VV dhukkubsatani/VV fayyan/NN ulfiina/AJ qaama/NN hirrisan/VV akka/PP deebi'u/VV fi/CC guddina/AJ sirri/NN akka/PP itti/PP fufu/NN danda'an/VP isa/PP dur/AD nyaatan/VV caalaa/AD akka/PP nyaatan/VV jajjabessuun/VV barbaachisa/AD Daa'imman/VV dhaloota/NN gad/NN aanaa/NN ./PU

qaban/AD Haatii/NN yoo/CC kan/AJ vaayirasii/NN HIV/NN Waliin/NN jiraatu/VV taate/VV daa'imman/VP sirritti/AD sooruuf/VV deegarsa/PP addaa/AJ isaan/PP barbaachisa/AD Aannan/AD harma/NN haadhaa/NN daa'imman/VP gad/NN aanaa/NN qabaniif/AD soorata/NN filmaata/NN hin/NG qabine/NN dha/AD Daa'imman/VP ulfiina/AJ qaama/NN gadi/AD ./PU

aanaa/NN qaban/AD tokko/NN tokko/AJ harma/NN hoodhuu/VV waan/PP hindandeenyeef/VNE aannan/AD eelmamee/VV kubbaayyaan/NN dhuguu/VV qabu/AD Annisa/NN qulqulluu/VV beeyladaaf/VV qaamaa/NN dabaluu/VV gargaaru/NN Bishaan/NN dabalatee/VV waliigalati/NN hooriidhaaf/VV jireenya/NN guudina/AJ qama/NN dabaludhaa/NN fi/CC sadarkaa/AJ oomishtummaa/NN horii/NN ./PU

irraa/PP barbaadamu/VV dhugoomsuuf/VV nyaata/VV barbachisaan/NN jechuudha/VV Beeyladoonni/NN kun/PP osoo/CC nyaata/NN fooyya'aa/NN hinargatiin/VNE gabaatti/PP waan/PP dhiyaataniif/AD carraa/NN oomisha/NN fooni/VV fi/CC uumurii/AJ ni/PP xinesa/NN Beelladoonni/NN nyaata/NN qulqullina/AJ hinqabne/VNE yeroo/AD nyaatan/VV qalamuu/VV ./PU

isaan/PP andeessisu/VV jalaa/PP tursa/PP foon/NN beeyladoo/NN qulqullina/AJ barbaadamaa/NN ta'e/VV hinqabaatu/VNE Baayyinni/AJ nyaata/NN goggogaa/NN beeyladni/NN tokko/AJ fudhachuu/VV danda'u/VV tilmaamaan/NN dhibbantaa/NN qaama/NN isaanii/PP 2/CN 3/CN kan/PP ta'uu/VV danda'u/VV dha/AD jedhamee/VV ni/PP yaadama/NN ./PU

Haaluma/PP kanaan/PP hojii/NN furdisuu/VV keessatti/PP oomisha/NN jechuun/VV dabaliinsa/NN qaamaa/NN fi/CC fooyya'iinsa/VV haala/AD qaamaa/NN beeylada/AJ jechuudha/VV Kanaaf/VV horiiwwan/AJ sagantaa/NN furdisuu/VV keessa/PP galan/NN nyaata/NN qulqulluu/VV jireenyaa/NN fi/CC qaamaa/NN akka/PP dabalann/NN isaan/NN gargaaru/NN barbaadu/NN Fedhiin/NN annisaa/NN beeyladoonni/NN barbaadan/VV

annisaa/NN qulqulluu/VV lubbuu/VV tursuuf/VV gargaaru/NN ;/PU Nem/NN ykn/CC
annisaa/NN qulqulluu/VV qaamaa/NN dabaluu/VV gargaaru/VV ./PU
jedhamee/VV ibsama/NN Annisaa/NN qulqulluu/VV lubbuu/VV tursuuf/VV
gargaaru/NN kan/PP hundaa'u/VV qaamaa/NN meetaboliikii/NN beeyladaa/NN irratti/PP
yoo/CC ta'u/VV kunis/VV haala/AD kanan/AD gaditti/PP ./PU