ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUTE STUDIES
SCHOOL OF INFORMATION STUDIES FOR AFRICA

AUTOMATIC PART OF SPEECH TAGGING FOR AMHARIC LANGUAGE
AN EXPERMENET USING STOCHASTIC HIDDEN MARKOV (HMM) APPROACH

A THESIS SUBMITED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCINECE IN INFORMATION SCIENCE

BY MESFIN GETACHEW

JUNE, 2001

I would like to thank Solomon Nega and Estifanos Gashaw, graduating class students in the Computer Science Department of the Science Faculty of Addis Ababa University, for devoting their time and providing me a technical support during the program development.

I thank associate professor Getahum Amare and Ato Leul Teklemedhin for their significant advice and editing some parts of my thesis.

I also thank my friend Ato Ermiyas Tilahum, staff of Information Systems Section of ECA, for his unreserved support in providing me Internet documents with out which my thesis would be unthinkable.

I thank also Ato Abiyot Bayou and my friend Ato Aydefer Negash for their significant advice and locating materials and facilities useful for my research work.

Finally, I should have to give my gratitude to people who are not mentioned in name but their effort helped me much to full fill the road of success.

I thank God for allowing me to graduate!

Abbreviations and symbols used

Symbols

{ } What is inside is an affix
/ / What is inside is a vowel
( ) What is inside is optional except those used to indicate cite (source) of documents used

\ To separate words from their tags

Abbreviations

Neg.	Negative
Aff.	Affirmative
Def.	Definiteness
3rd.	3rd person
sing.	Singular
pl.	Plural
masc.	Masculine
fem.	Feminine
Pol.	Polite
N	Noun in all forms
NP	A preposition not separated from a noun
NC	A conjunction not separated from a noun
NV	Verbal nouns
NB	Noun prefixed with balä
V	Verb in all forms except auxiliary, compound and all forms of auxiliary and compound verbs
AUX	Auxiliary verbs and all their other forms
VCO	**C**ompound verbs
VP	**A p**reposition not separated from noun
VC	A verb prefixed or suffixed by a conjunction
J	An adjective
JC	A conjunction not separated from an adjective

JNU	A numeral used as an adjective
JPN	A noun not separated from a preposition and that **function** as an adjective
JP	An adjective not separated from a preposion
PREP	A preposition
ADV	An adverb
ADVC	An adverb not separated from a conjunction
C	A conjunction
REL	Relative clause
ITJ	Interjections
AmharicText	The original Amharic text
LatAmharicText	Latin version of the original Amharic text
LexicoProb	Lexical probabilities matrix
TransProb	Transitional probabilities matrix
ManuTaggedText	Manually tagged text
TrainingSet	Training set
TestSet	Test set
Sampledb	Sample database

Abstract

Natural Language processing, as a field of scientific inquiry, plays an important role in increasing computers capability to understand natural languages. Part of speech (POS) tagging is one effort in the task of understanding natural language, the language by which most human knowledge is recorded.

The task of POS tagging is to assign unique part of speech tags to sentences that are presented as a linear string of words. POS tagging systems, which annotate corpora written in various languages (e.g. English), are used as components in many applications including phrase recognition, word sense disambiguation, grammatical function assignments and many others.

Today, taggers of different kinds have been developed for languages, which have relatively wider use nationally and/or internationally. The same story is not true for Amharic, the working language of the Federal Government of Ethiopia, and one of the major languages of Ethiopia (Bender, 1976) for there are no systems (taggers of any sort) that annotate corpora written in this language.

This study is, thus, an attempt to develop a simple automatic part of speech tagger for Amharic language.

In the study, the Viterbi algorithm was used without any modification. A module for sentence splitter was also developed in order to facilitate the preparation of texts in a file to be tagged with appropriate parts of speech. POS tags were also designed on the basis of the review made regarding the linguistic properties of the Amharic word classes. These tages, in fact, are the first in their kind for this language.

The study adopts the Stochastic Hidden Markov Model approach to develop a prototype, which is a simple Amharic tagger for the language.

The thesis, in short, describes processes of automated part of speech tagging from manually tagging texts to developing a prototype and conducting an experiment with it. The result obtained using the small manually tagged corpus encourages further research to be launched, especially with the aim of developing a full-fledged Amharic POS tagger.

LIST OF TABLES AND FIGURES

TABLES

FIGURES

TABLE OF CONTENTS

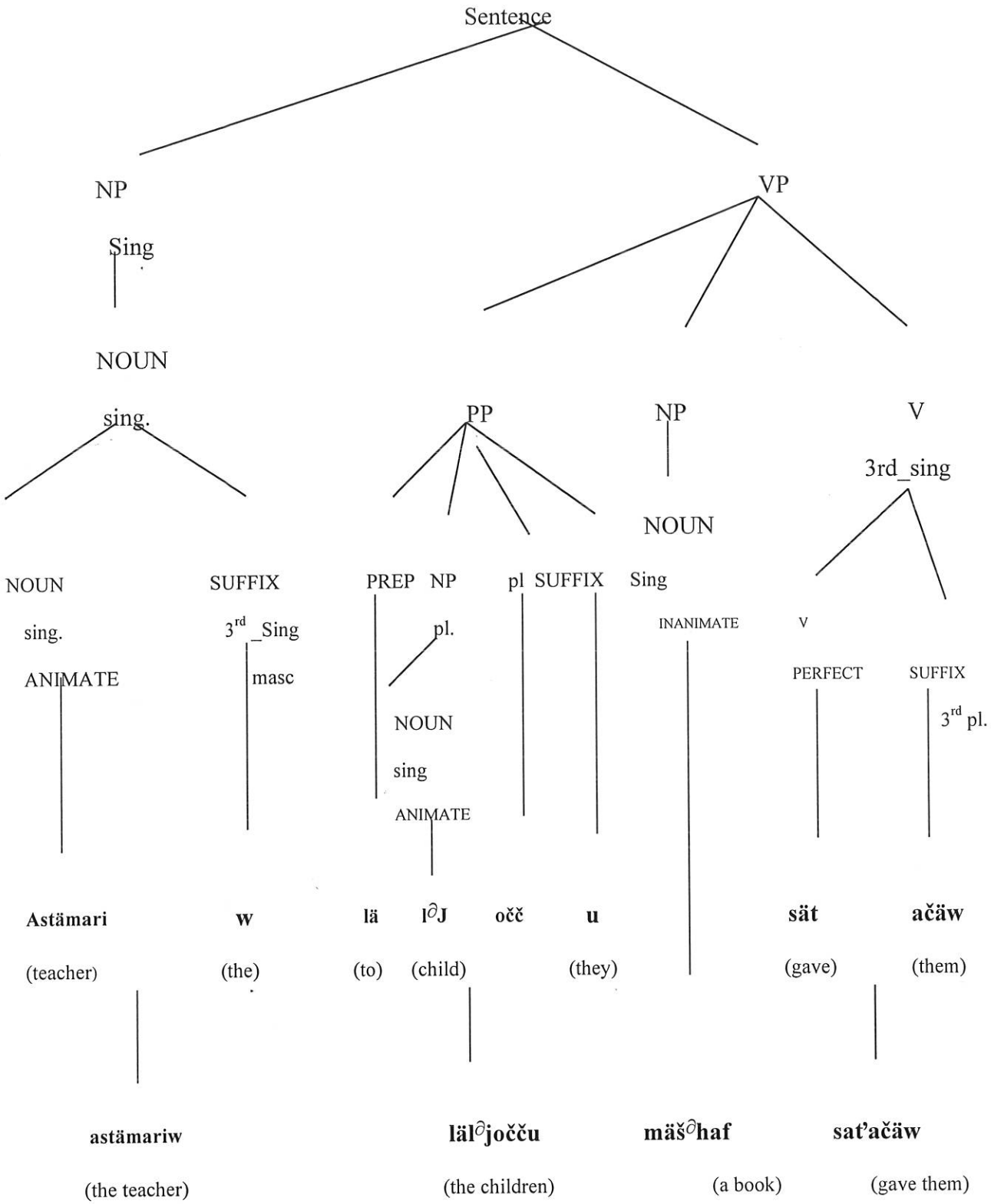
CHAPTER ONE
INTRODUCTION

1.1 Background

Natural language is any language that human beings learn from their environment and make use of it to communicate with each other in their day–to-day activities. As Abiyot (2000) clearly puts, natural languages are used to express our knowledge and emotions and to convey our responses to others. English, French, Arabic and Amharic are all examples of such natural languages. Each of these languages have structural descriptions[1]. The structural description of the Amharic sentence **astämariw läl∂jočču mäšïhaf satačäw** "The teacher had given a book to the children" for instance, is shown in the next page.

[1] See (Culicover,1976; Brill, 1993)

Figure 1.1: An example of Amharic Structural information



The notations in the figure are coding devices and are used in this chapter to mean the following.

NP	Noun phrase	PP	Prepositional phrase
VP	Verb phrase	3rd_pl	Third person plural
sing.	Singular	PREP	Preposition
pl	Plural	V	Verb

The figure in 5.1, which is a tree-diagram, shows the structural information contained in the Amharic sentence mentioned earlier. For instance, the diagram shows that **ləjočč** "children" is the plural form of ləj "child" and that the {-u} "they" and the "**lä**" "to" are subject suffix and preposition, respectively, and that the four words lä, **lə**j, o**čč** and **u** form läləjo**čč**u "to the children", which is a prepositional phrase.

NLP, as a field of study, studies ways on how to increase computer capabilities to process such structural information contained in the Natural Language (NL). It, in particular, deals with devising a mechanism (e.g. developing an algorithm) so that they capture the sort of structural information typified in the example and contained in the natural language

NLP requires Natural Language Understanding (NLU) and modelling the Natural language so that computer programs that act appropriately on the information contained in the text or utterance of the language can be developed. Being a sub area of NLP, NLU attempts to understand a NL. But the problem is that the effort to understand a natural language seems not to be easy and straightforward. In fact, as Mao (1997) states, NLU is an extremely complicated problem. The complexity is mainly due to the fact that natural language involves large number of classes and relationships whose existence is not transparent from the surface structure[2] of the natural language. This makes, for instance, difficult the assignment of words

[2] See (Stockwell, 1997)

to their appropriate classes and also to identify the relation that they have with other words in the sentence, paragraph or document.

Inflection and derivation are also other reasons that make NLU complex. By attaching suffixes and prefixes or by adding or removing vowels between consonants, a number of different forms with different meanings can be formed from a single basic unit. Especially, in highly inflected semantic Languages like Amharic (Girmay, 1992), different word forms can be generated from a single basic unit. The following example shows just a few of the many word forms that can be obtained from a single basic unit of the Amharic word **gädäl** "kill" **by simply attaching suffixes and combination of suffixes only.**

gäddäl-ku	"I killed"
gäddäl-ku-at	"I killed her"
gäddäl-ku-l-at	"I killed(sth. or sb.)for her"
gädäl-at	"He killed her"
gädäl-u-l-at	"They killed sb/sth for her"
gädäl- äčč	"She killed(sth. or sb.)"
gädäl- äčč-b-ačč-w	"She killed (sb/sth.) of them"
gädäl- äčč-l-ačč-w	"She killed (sb/sth.) for them"
gädäl-n	"We killed (sb/sth.)"
gädäl-ä-ňň	"He killed me"
gädäl-u-n	"They killed us"
gädäl-u-b-n	"They killed (sb/sth.) of us
gädäl-u-b- aččäw	"They killed (sb/sth.) of them **and so on**.

Hence, if computers are to understand NL, they should be capable of handling such (thousands and millions) variants of the same basic word form together with the unique meanings and the specific interpretations that each form has. This further complicates the task for computers to understand NL.

In general, **subtle semantic properties** (e.g. **still** used as noun, adjective and verb have different meanings), **relationships that are not apparent**, **inflectional**, **derivational** and **phonological** properties of the NL are major reasons that make computers capability to understand NL a more complex task.

Although NLU by computers is a complicated problem, there are various approaches under investigation and some are succeeding to some extent in making computers understand NL. Part of speech Tagging (POST), which involves selecting and tagging the most likely sequence of word categories for words in a sentence, is one instance in the effort to increase computer capabilities to understand NL. In with this, it is the purpose of this study to explore the possibility of developing an automatic POS tagger useful for tagging words in Amharic texts with their appropriate parts of speech.

1.2 Application of Natural Language Processing

NLP can be applied in a number of areas. According to Abiyot (2000), the following are some application areas of NLP.

- Designing a friendly and flexible interface
- Structuring large bodies of textual information for the purpose of automatic indexing and automatic abstract generation

- Machine translation
- Spelling and grammar checking
- Analysis of language
- Language learning
- Speech recognition and speech synthesis
- Text compression

Other applications of NLP include **term and name identification, word sense disambiguation, morphological analysis, unknown word processing, natural language generation, data mining and entity extraction** and **part of speech tagging** (Mao, 1993).

1.3 POS Tagging Systems and their Benefits

To the best of my knowledge, no work has ever been reported in the area of automatic **POST** for Amharic Language. Apart from the absence of researches in this area, there are many other reasons that make such researches on POST attractive in general and in Amharic language in particular. These are discussed below in this section

Sentence parsing[3] (e.g. syntactic or semantic) is a task of **NLP**. It is used to solving basic problems (such as language comprehension and production work) that most **NLP** applications face (Mao, 1997). **POST** is a step forward towards such approach. That is, subsequent **NLP** such as syntactic and semantic representation of the language will be very difficult, if not impossible, without first having the underlying corpus tagged with appropriate parts of

[3] is a technique or method that deals with the decomposition of a sentence into its major sub parts, namely noun phrase (NP), Verb phrase (VP), Noun (N), Verb (V) etc.

speech. In other words, POST is usually taken to be the first step in processing a language at sentence level (Mao, 1997). Before a sentence is parsed (i.e. **assigned a structure and/or meaning**), the sentence must be annotated with parts of speech using a POS tagger. Then after, the tagged or annotated sentence will be submitted to the sentence parser so that the parser parses the sentence effectively and easily.

A POST system is also useful in works related to **machine translation**. Especially, the increasing use of Internet in this information age makes machine translation a pressing need. This is mainly attributed to the fact that many people (the majority) cannot make use of the huge information available on the Internet unless translated to local languages for they are ignorant to foreign languages. This makes, in one way or another, works on tagging (and also syntactic and semantic parsing) more useful. Being a component in machine translation systems, a POS tagging system serves as one important tool in the attempt to develop systems that translate foreign languages into local languages (e.g. Amharic) and vice versa.

A POST system is also crucial for such tasks as **word sense disambiguation** or **ambiguity resolution** (Mao, 1997). Ambiguity resolution or disambiguation is a key task in natural language processing. In ambiguity resolution, the disambiguator decides on the appropriate class or POS of an ambiguous lexical item or word, and an item that is in a different context can be classified differently. For instance, the English word "**can**" is an ambiguous word for it has the following three possible parts of speech

Can MD=Modal V=Verb N=Noun.

In the sentence **"Can we can the can?"**, for instance, the disambiguator should decide in which sense "**can** "is used in the different positions. A part of speech tagger is one possible answer to such problem. The tagger basically uses **contextual rules** and a **lexicon** to assign

tags to ambiguous words. If the input text is an output from a POS tagger, the problem of disambiguation will be minimized, if not removed at all.

Automating the tagging process will also **address the difficulty of hand tagging** needed to build large corpora useful for many phenomena. For instance, a large corpora tagged with parts of speech is essential to study a number of linguistic phenomena (Brill, 1993). Lack of such large tagged corpora may limit, if not completely halt, researches that heavily relay upon large tagged corpora.

An automatic POS tagger is also a useful tool for works related to **spell checking** and **speech recognition and generation**. Emphasizing this, Brill (1993) made the following statements.

> A reliable part of speech tagger is also a useful tool in isolation. Part of speech tags can aid systems such as spelling correctors and speech recognition and generation systems. For instance, if a speech system is to properly pronounce the word record it must know whether the word is being used as a noun or a verb.

POS tagging systems are also used as component in many other applications including **phrase recognition, word sense disambiguation, grammatical function assignment** and so on (Cutting et al, no year). According to Brants (1997), POS tagging systems are also used as component in **parsing, for recognition in message extraction systems, for generation of intonation in speech production systems** and many others.

In short, an automated POS tagger is useful to increase retrieval and analysis of structural or linguistic information contained in the NL. It forms an essential foundation for further forms of analysis (such as syntactic and semantic parsing and also machine translation). It allows us to distinguish between homographs. It is useful for extracting meaning from a sentence and checking the well formedness of a sentence, which is useful in a number of applications.

1.4 Statement of The Problem

As Abiyot stated, researches made in the area of Amharic language processing using computers reveal that only limited number of researches are conducted in relation to computer processing of the Amharic language. In fact, most of such researches conducted mainly focus in the area of Character Recognition (Worku, 1997; Dereje, 1999), Text to Speech Synthesis (Laine, 1998), Text Retrieval (Biru, 1992; Nega, 1999) and so on.

Apart from these and other limited researches in the area of computer processing of the Amharic Language, there is only one research conducted in the area of NLP (Abiyot, 2000). Though not complete, this research tried to address the need for having a word parser for Amharic Language, which its absence, as Abiyot clearely stated in his work, will make NLP for Amharic difficult.

Like word parsing, POS tagging is another task that should be addressed in the area of NLP of the Amharic Language. Researches in POS tagging will contribute a lot in the effort to increase computers processing of the Amharic language. The absence of POS tagging systems limits our effort to make computers understand the Amharic language. Especially, further and higher forms of researches such as parsing (syntactic and semantic) and machine translation

that helps to quickly and easily retrieve information of the language will be difficult without first having a part of speech tagging systems.

The absence of a POS tagging system limits also researchers who require annotated [4]corpora to address different issues in linguistics and computational linguistics. The absence of a POS tagging system limits also works related to spell checking and speech recognition and generation in the area of Amharic language

Thus, having mentioned all such pitfalls, it is worth to conduct a research and develop a simple automatic part of speech tagger for Amharic language based on the property of the Amharic word classes.

1.5 Objective of the Study

1.5.1 General

The general objective of this study is to explore the possibility of designing a simple automatic POS tagger for Amharic texts (sentences).

1.5.2 Specific Objectives

In line with achieving the above general objective, the study will attempt to address the following specific objectives.

[4] Discussed throughly in chapter two of this study.

1. Review the basic word categories (or POS) of the Amharic language with the aim to derive the POS tags and hence the tag set for the language, which in turn allow computer representations of the structural or linguistic information contained in the NL, Amharic.
2. Review the morphological property of the language to identify properties useful for POST
3. Review the structure of Amharic sentences
4. Study the type of lexicon required for Amharic POST and design the appropriate lexicon
5. Review the various techniques (or approaches) suggested (elsewhere) for the development of a POS tagger for Amharic language.
6. Select and customize (if necessary) a POST algorithm for use with the Amharic language
7. Develop a prototype, an Amharic POS tagger.

1.6 Method

Developing an automatic POS tagger for Amharic language requires one to investigate and identify the properties of this language. For this purpose, a **Review** of literature was made in the area of Amharic word classes and its morphological property. Literatures in the area of POST (e.g. Brill's tagger and statistical tagger) were also reviewed for this study.

Discussion with linguists and experts in the area of Amharic language were made to better understand the language and get suggestions that are invaluable for the study.

Based on the analysis of the Amharic language, a POST algorithm that suits the language was selected and used with no modification. A **lexicon, tagset** and **statistical databases** were designed and a prototype developed to tag sentences appropriately.

The prototype developed was **tested** using a small sample text selected for the test purpose. The evaluation of the performance of the tager was made based on only the percentage of correct tag assignments. The experiment was made in two phases, on training set and test set, and was repeated several times until the tagger performance was found to be satisfactory. Test results achieved are finally reported together with the discussions made.

1.7 Application of Results And Beneficiaries

As outlined in section 1.3 and also presented in the statement of the problem, POS tagging systems are useful in many areas of NLP for Amharic language. Thus, the beneficiaries of the results of this study includes researchers involved or wants to be involved in increasing computers capability of processing Amharic Language. Among them are researchers in the area of parsing, spell checking and speech recognition and generation of Amharic Language.

The out put of this research may also be used in language teaching to create **awareness of the word and phrase** classes of the language, **the member of** each class and the relationships that holds between them. In this respect, POS tagging is useful to discover the linguistic structure of large corpora. The part of speech information obtained using the POS tagger as a component facilitates higher levels of analysis such as noun phrase recognition and so on.

1.8 Limitations of The Study

1. The tagger must have been trained and tested several times (at least 10 folds) on different segments of the manually annotated corpus to see its performance. Unfortunately this was not done due, mainly, to time constraint and arduous nature of preparing the lexical and transitional probabilities required for the knowledge base.

2. This study is limited to a small sample due to a lack of large corpora in the language annotated with POS and also generating such large corpora is very laborious, expensive and time consuming.

3. The tagger developed is purely statistical. That is, lexical(or morphological) rules are not incorporated. These rules were useful to make reasonable guesses as to the POS category of unknown words, rather than simply attaching them the tag UNC.

1.9 Scope

The scope of the thesis is limited to demonstrate the potential of a Stochastic Hidden Markov approach to develop a simple Automatic Part of speech tagger for Amharic texts.

1.10 Organization of The Thesis

This section describes the organization of the rest of the thesis. Chapter two presents what POST is, approaches to automatic POST, issues such as corpus annotation, lexicon for a tagger and steps involved in POST. Terms such as tag, tagset, lexical probabilities and contextual (or transitional) probabilities are defined. The chapter also attempts to discuss Stochastic tagging to the level of presenting all computational works needed for developing a Hidden Markov model tagger.

The third chapter describes in detail word classes in Amharic language. Included here are word formation processes such as inflections (also called conjugations) and derivations.

Chapter four, the core of this study, dwells completely to draw part of speech tags and hence the tag set for Amharic language. The tag set drawn was obtained from the analysis of word classes made in Chapter three.

The lexicon designed, the algorithms used to develop the automatic POS tagger and the computations made to get the lexical and transitional probability matrixes are presented in chapter five, which is also another **core** chapter in this thesis. The database designed to hold the knowledge base, the evaluation procedures, and the experiments made and the results achieved are also discussed in this chapter.

Chapter six, the last chapter, concludes the thesis presenting conclusions and recommendations. This chapter will also indicate some pointers to future works. A bibliography to be used for further reading is also included at the end of this chapter. Appendices for this thesis are found separately in the **Bibliographic Lab of the school of information studies for Africa [SISA].** The appendices are not compiled with this thesis due to the requirement that the thesis is limited all in all to 150 pages. These appendices provide additional information on some of the topics discussed in the different parts of this thesis and are thus compiled in the form of booklet to be used together with this thesis.

CHAPTER TWO

Automated Part of Speech Tagging

2.1 Introduction

POS tagging, as defined earlier in chapter one, is the task of assigning an appropriate part of speech (**POS**) or word category (e.g. noun, verb, adjective and so on) to each word in a sentence, or corpora. In this task, special codes or labels will be attached (or assigned) to each word in the corpora indicating its particular category (and other features). While the codes attached are known as "tags" the process of attaching is called tagging. Tagging can be done either manually or automatically. Here is an example used to elaborate these concepts using the Amharic sentence **zare hämus näw "** Today is Thursday**".**

The process that accepts this sentence as an input and generates an output that appears like

Zare\ADV **hämus**\N **näw**\AUX

is a tagging process. The codes ADV, N and AUX are tags that indicate part of speeches of the words in the given sentence. The tags ADV, N and AUX in the above example represent an adverb, a noun and a verb respectively.

According to the Oxford Advanced Learner's dictionary, a corpus (plural corpora) is a collection of written and/ or spoken texts. The term can also refer to any collection of texts or a body of text that is carefully sampled to maximally represent the entire corpus.

A corpus can either be **annotated** or **unannotated**. Unannotated corpora are texts that appear in their raw form. That is, they are free or fresh or plain texts. They are not enhanced with any kind of **structural** or **linguistic** information. An annotated corpus, on the other hand, is a text

that is enhanced with various types of linguistic information. The difference between the two is illustrated using the Amharic word gäddälä "he killed".

If the word "gäddälä" appears as it is, it is an unnotated or untagged. But, even in this case, the word "gäddälä", implicitly indicates the POS and other features such as third person, singular, masculine in the perfect tense. If the word "gäddälä " appears as **"gäddlä**\VI3sm" then, it is annotated because it is enhanced **explicitly** with such structural or linguistic information as third person (3), singular(s), masculine (m) and a verb in the imperfect tense (VI).

Such annotations of words in corpora have many advantages. Linguists, for instance, use corpora annotated (or tagged) with appropriate parts of speech to study a number of linguistic phenomena. In computational linguistics, annotated corpora are also useful to train part of speech taggers and parsers. Lack of such annotated corpora limits researchers and makes also difficult the task of quickly and easily retrieving and analyzing information about languages contained in the corpus.

A corpus can be annotated either manually (by hand) or automatically (using computers). Manually **tagging** corpora is expensive and laborious. Besides these, manual annotation requires annotators to have knowledge of the structure of the language under consideration. This might be a problem if such annotators are not available. In fact, even where they are available, different experts may annotate the same corpora differently, there by creating bias in works that later use such annotated corpora.

On the other hand, a given corpora can be annotated automatically (e.g. with appropriate POS) using computers with little human involvement. Computer-based Taggers and parsers are systems that do this job of annotating a corpus. In fact, POS tagging is the most basic type and is the first type of linguistic corpus annotation[5].

Today there are many systems that annotate corpora written in various languages. This makes annotated corpora to be widely available than before. This, in turn, has become a relief for many researchers who make use of corpora in their work. The same story is not true for Amharic language for there are no systems (taggers or any other) that annotate corpora written in Amharic. **This study aims to address this problem by developing an automated POS tagger with a very minimal accuracy and aiming to shade light for future works.**

2.2 Approaches To Automatic POS Tagging

As it can be inferred from the previous section, the need for accurately tagged texts has increased in the past few years. As a consequence, a number of taggers have been built to tag each word in a corpus written in different languages (e.g. English, French, Romanian and so on)[6] automatically. Because of this, today, there are many approaches in developing such automated POS taggers. The most basic, popular and competing approaches are two: **Stochastic approach** and **Rule-based approaches.** This section discusses these two approaches giving much emphasis to the first approach for it is the approach selected for use in this study.

[5] Anaphoric and prosodic are other examples of annotation.
[6] See (Chanod, no year), (Brill, 1993), (Eynde, no year)

2.2.1 Stochastic Approach to Tagging

The term "Stochastic approach" is a generic name to such approaches that make use of probability (i.e. statistics) to tackle the problem of POS tagging. Probabilistic approach and statistical approach are other terms that seem to be used interchangeably with stochastic approach both in the literature and in here.

In stochastic approach, probability concepts such as Bayes' theorem, the notation of independence and the Markov Assumption play important roles. The approach applies to a great extent such probability theories in part of speech identification. The approach, for instance, applies such probability theories to determine the most likely lexical category for each ambiguous word in a given sentence. Understanding how probability theories are applied in POST requires understanding the underlying probability theory related to POST. Thus, this section and most of the other parts of this chapter are devoted in introducing the statistical tools needed for designing a POS tagger.

Stochastic approach will be presented in this section assuming the Hidden Markov Model approach. The Hidden Markov Model taggers are selected for they are efficient and effective statistical taggers developed using stochastic approach.

Taggers developed using stochastic approach are classified into two depending on whether the taggers are trained on **pre-tagged** or **untagged (fresh)** texts. Taggers that need pre-tagged corpora throughout the learning processes are called **supervised taggers and the approach is called supervised approach.** Markov-model based taggers are examples of such supervised taggers. Those taggers that do not require pre-tagged corpus during the learning process are called **unsupervised taggers and the approach is called unsupervised approach** (Guilder,

1995,). Thus, the stochastic approach to the problem of tagging can either be supervised or unsupervised.

In supervised approach two things are important to serve as a source of information. The first one is a lexicon[7] that lists each word with all the allowable POS (or tags) that each word can have with its respective estimate of **lexical probabilities[8].** The dictionary and the lexical generation probabilities, p (word\tags), can be stored in either the same or different databases. The presence of such dictionary with/and lexical probabilities provide information on the possible parts of speech for a particular word.

The second one is a list of contextual probabilities for each tag (or POS) to occur in a sequence of tags. In other words, the probability of a certain tag to occur in a sequence preceded by one or more previous tags, denoted as p (C_i\ C_{i-1} C_{i-2} … C_{i-n}) or (p (tag_i\tag_{i-1} tag_{i-2}…. tag_{i-n}) is the other source of information. Such list of contextual probabilities, like lexical probabilities, is a table of statistics that provides **contextual information**. It indicates the particular tag that is appropriate for a particular context.

In a supervised approach, the tagger works using such two sources of information that can easily be generated from a pre-tagged corpus. A good and readable account on how to get estimates for contextual and lexical probabilities using a pre-tagged corpus is found in (Alen, 1995).

Once the dictionary and the database for lexical and contextual probabilities are created and stored, the tagging process in Stochastic approach, as any other approach, begins by looking

[7] A lexicon is a linguists term for dictionary
[8] Lexical probability is the probability of a word to have a particular tag and is usually denoted by p(word/tag).

up each word of the text in the lexicon. While looking up the word in the lexicon, the word may happen to be found in the dictionary or not found in the dictionary. A word not found in the dictionary is called an **unknown word**. Both Unknown words and words found in the dictionary may be categorized into different parts of speech. For instance, the English word **still** can be categorised into such parts of speech as **Noun, Verb, or Adverb** depending on the context in which the word is found in a sentence. Such words or lexical items that can fit into different categories depending on the context are called **ambiguous words.**

The difficult task in any tagging processes is to deal with unknown and/or ambiguous words. Lexical items that are ambiguous and /or unknown may be tagged wrongly. Different approaches handle such problems differently. In stochastic approach the following three alternatives are used to **disambiguate ambiguous words**.

Word frequency approach: The simplest stochastic taggers are the ones that mostly use this technique. In this approach, calculation of disambiguation is based on the probability that the word occurs with a particular tag. Meaning, if an ambiguous word appears most frequently with a certain tag in the training set, then that tag will be assigned to the ambiguous tag. Although this approach yields a valid tag for a given word, it has its own problem. It may yield sequences of tags that are not acceptable or admissible (Guilder, 1995).

n-gram Approach The n-gram approach is an alternative approach to overcome the problem of word frequency approach in disambiguating ambiguous words. In n-gram approach, the best tag for a given word is determined by calculating the probability that the word occurs with n-previous tags (Guilders, 1995). The Viterbi Algorithm, to be discussed later in this chapter, is an implementation of the n-gram approach

Hidden Markov Model (HMM) Approach: The HMM approach to the problem of disambiguation combines both the word frequency approach and the n-gram approach. The approach uses both approaches to determine the best tag for a given word. That is, word frequency measurements, i.e. (p (word\tag)), and tag sequence probabilities, i.e. p (tag_i\tag_{i-1}.... tag_{i-n}), are taken into consideration to yield an admissible sequence of tags besides yielding a valid tag for a given word. As it will be discussed later in this section, this approach makes heavy use of statistical calculation to get the most appropriate **tag sequences** for words in a given sentence (or corpora).

For HMM approach to yield correct results, the basic underlying assumption often called Markov assumption in probability theory must be valid. The Markov assumption states that in calculating the probability of any sequence of categories, the probability of a category occurring in the sequence must depend only on the n categories where n is an integer equal to the number of categories before the category under consideration (Allen, 1999; Guilder, 1995). This study assumes this assumption for the study will use similar approaches that are used to develop HMM tagger while developing an automatic POS tagger for Amharic language.

One problem in developing all supervised taggers in general and HMM taggers in particular is the lack of manually or automatically tagged corpus. This is mainly attributed, as explained earlier, to the fact that tagging texts manually is expensive and time consuming. Another problem associated with supervised taggers, taggers that need tagged text in the training process, is the need to manually tag text each time the tagger is applied to a new type of text or in the same language (Brill, no year). But if pre–tagged corpora are easily available, the

HMM approach in particular and stochastic taggers in general can be adopted in new languages with little effort.

Contrary to the supervised stochastic approach, taggers developed using unsupervised stochastic approach do not require pre-tagged text in the training process. The unsupervised and supervised HMM taggers are generally similar in the following respects.

- Both of them assume the same underlying HMM
- They use a large dictionary (that lists all allowable POS for each entry word) and **inflectional information** to determine the possible allowable parts of speech for words in a corpora
- Both of them involve calculation of taggers accuracy to improve performance and get better results.
- In both cases disambiguation can be achieved using statistical, hybrid or rule based approaches.

The unsupervised stochastic taggers differ from the supervised ones in the following respects.

- Tagged corpus is not necessary during the learning processes. That is training is on an untagged or fresh text.
- Training is more difficult for the set of state transitions used to generate the training corpus are not visible
- They use different algorithms called Baum-Welch algorithm while the supervised ones use the Viterbi algorithm.

2.2.2 Rule-Based Approach to Tagging

This is another approach to tackle the problem of automatic POS tagging. It mainly differs from stochastic approach in that it does not make any use of probabilities (or statistics) to tag and disambiguate words in a corpus with their appropriate POS. The following discussion on this approach is based on Brill's tagger reported by Mao (1997)

Brill's tagger, a tagger developed by Eric Brill, is one of the best rule-based tagger developed using rule-based approach. According to Mao, the Brill's tagger system learns a set of rules automatically based on a given corpus and then tags words following these rules. The Brill's tagger in particular and taggers developed using rule-based approaches in general require different components to effectively tag each word in sentences.

A POS **tagset** is one of the basic components of a rule-based tagger. The tagset is a list of all word categories that will be used in the tagging process. It provides distinct coding for all classes of words having distinct grammatical behaviour.

A **lexicon (or dictionary)** is also required as a component of the rule-based tagger. This is required because the tagger (e.g. Brill's tagger) begins processing a sentence by looking up each word in the lexicon. The lexicon contains a list of all possible parts of speech that the word can be assigned. For example, the word **can** in English has the following possible parts of speech.

Can N V MD Where N=noun V= Verb and MD=Modal

In the sentence **can\Modal we\Pronoun can\Verb the\Determiner can\Noun**?, for instance, the word **Can** has all the three parts of speech mentioned. These are indicated in the sentence by attaching a backslash after each appearance of the word **can** (to serve as a separator) and

following it by the POS that **can** have at the different positions in the sentence. Thus, the word **can** in the lexicon must be stored in such a way that the dictionary provides all such parts of speech information.

In rule-based approach, there are also techniques to tag unknown or ambiguous words. Such techniques are commonly called **tag-changing rules**. These tag changing rules must be incorporated as a component so that the tagger applies the rules as necessary while tagging words with their appropriate POS. Such tag changing rules are necessary to provide information that indicates the particular tag that is appropriate for a particular context (Brill, 1993). In rule-based approach, these tag-changing rules are of two types, **contextual rules and lexical rules.**

The **contextual rules** are pre-specified as a set of transformation templates and provide contextual information to assign tags to unknown or ambiguous words. They revise the tag of a particular word based on the surrounding **words** or the **tags** of the surrounding words. **N V PERVTAG TO,** for instance, is one example of contextual rule in English. This rule, for instance, forces to change tag **N** to tag **V** when the preceding word is tagged (abbreviated as **PREVTAG**) with the tag **TO.** According to this rule

to**TO** draw**N** would **be changed to** to**TO** draw**V**

The rule **det – X - n = X\adj**, in English, is an example of another contextual rule. This rule is used to tag both **unknown** and **ambiguous** words. This contextual rule dictates that if an ambiguous or an unknown word **X** is preceded by a determiner and followed by a noun, then the unknown or ambiguous word must be tagged with an adjective (Guilder, 1995). The abbreviations for the POS codes in the two examples are not consistent. This is to indicate

that people use different notations for the same POS (e.g. N and n are both codes to mean noun).

As can be seen from the two examples, contextual rules are not useful only to tag unknown words but also to eliminate tags that are wrongly or illegally assigned to words.

The **lexical (or morphological) rules** are also useful components in rule-based approach. The lexical rules provide information useful to treat **words that are not in the lexicon of the tagger**. In other words, such rules are useful to make reasonable guesses as to the category (or POS) of **unknown words**. For instance, assuming the English word **worked** is not in the lexicon, this word will morphologically be analyzed, i.e. using the lexical rules, as **work + ed** and it will be tagged as VVD (a code for past tense of the lexical verb work). This is by far better than making a wild guess as to the category of the word **worked.**

In treating words that are unknown during the tagging process, some systems include rules pertaining to **capitalization** and **punctuation.** These rules of capitalization and punctuation are additional rules incorporated in such systems besides contextual and lexical rules. But, such information on capitalization and punctuation may or may not be useful in the tagging process depending on the language being tagged. In German, for instance, information about capitalization proves to be extremely useful in the tagging of unknown **nouns**. On the contrary, rule of capitalization is of no value for Amharic language as there are no such things as capital and small letters.

Like Stochastic approach, the rule-based approach might be supervised or unsupervised. Rule based taggers that do not require pre-tagged text are uncommon and started to appear very

recently. That is, most rule-based taggers are supervised taggers and hence require pre-tagged corpora. Detailed discussion on rule-based tagger may be found in (Brill, 1993)

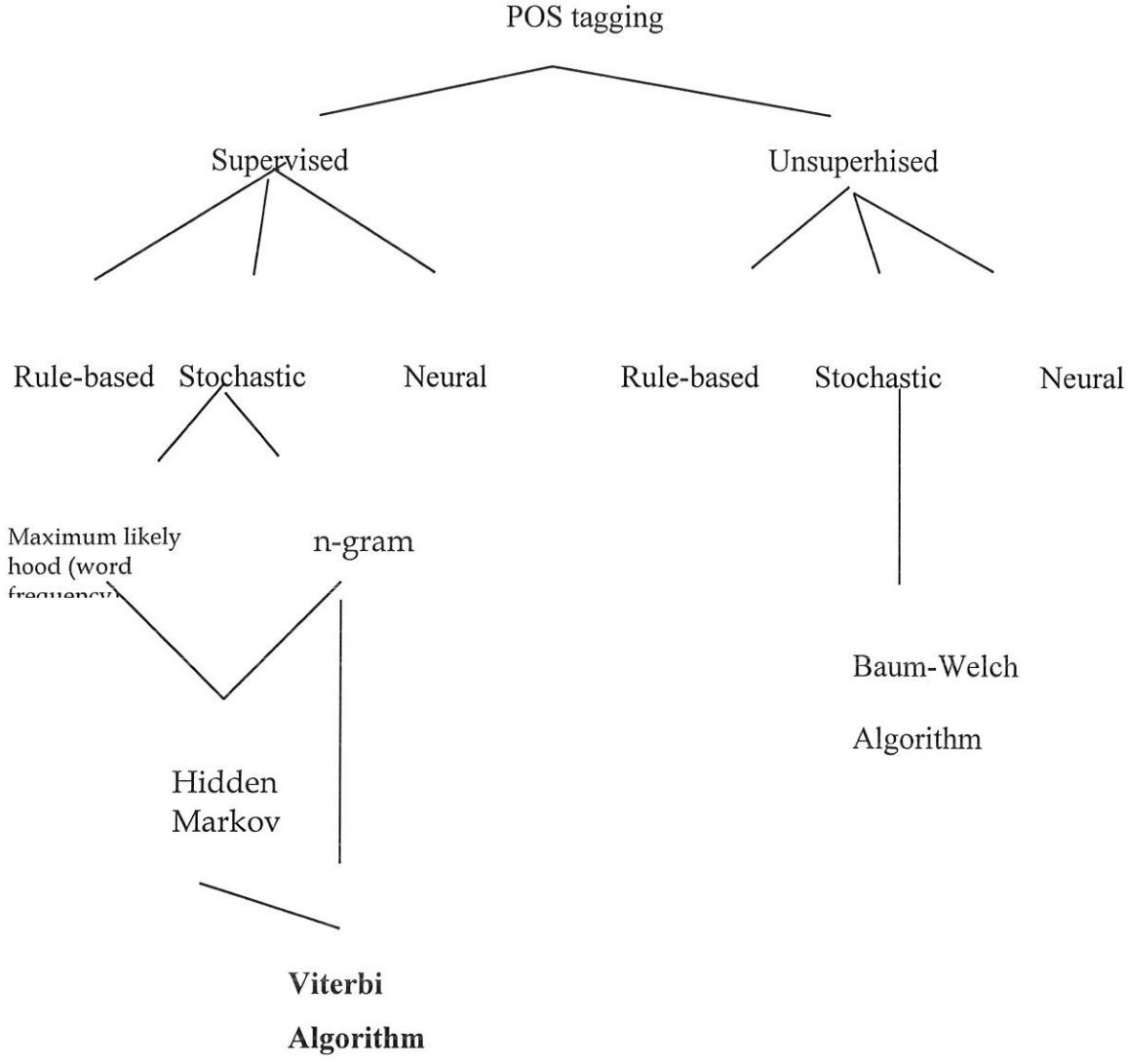
The drawbacks or shortcomings of rule-based taggers include the following (sources).

- They require large amount of effort to write the disambiguation rules
- No unsupervised training algorithm has been presented for learning rules automatically without manually annotated, except the one developed recently by Eric Brill (Brill, no year)

> In addition to the two basic approaches mentioned earlier, review of literatures reads two other approaches to the problem of automatic POS tagging. These are **hybrid approach** and **neural networking approach**. Since the two approaches are not common and as such not widely used, no discussion is made in this paper on such approaches. But, it should at least be mentioned that the hybrid approach is an approach that combines both the rule-based and stochastic approach (source).

The following diagram taken from Guilder (1995) presents a summary of the various approaches to automatic POS tagging presented in section 2.2.

Figure 2.1 Approaches to automatic POS tagging (Guilder, 1995)



2.3 Stochastic Vs Rule-Based Approaches (similarities and difference)

The similarities between rule-based and stochastic approaches include the following.

- Both of them may or may not need pre-tagged corpora
- Both of them use contextual information from an already set of allowable part of speech tags for disambiguating words
- Until recently, non of them handle unknown words in a way that is completely portable

Differences between the two approaches include the following.

- Rule–based approach uses contextual and morphological information to deal with unknown words while stochastic approach uses probabilities to deal with such words.
- A stochastic approach has no rule-based mechanisms for disambiguation. Instead it uses only probabilities for dealing with disambiguation (Brill, no year)
- Rule-based tagers generally perform as good as or better than that of stochastic taggers. In fact; taggers developed using rule-based approach have the following advantages over taggers developed using stochastic approaches.

Compactness: Taggers developed using rule-based approach require less storage than taggers developed using stochastic approach.

Speed: Some argue that taggers developed using rule-based approach are ten times faster than the fastest stochastic tagges (Brill, no year)

Better accuracy: Taggers developed using rule-based approach are reported to have better accuracy than those developed using stochastic approach.

Better adaptability (easy to train): Twisting or modifying rule-based taggers for the purpose of correcting errors is easy and straightforward. On the other hand tuning statistical taggers is very difficult for it is hard to predict the effect of tuning the parameters of the system

Considering all such similarities and differences, i.e. advantages and disadvantages, it seems reasonable to select the rule-based approach to deal with automatic POS tagging for Amharic language. But, due to the reasons listed below the stochastic approach, in particular the HMM

approach, was selected and applied in this study. Reasons for selecting the HMM approach to develop an Amharic tagger includes the following.

- Time available for the work.
- The researcher interest in the HMM approach.

2.4 HMM Stochastic Tagging

AS mentioned earlier, the HMM approach is considered for application in this study. Some important terms and concepts related to stochastic HMM tagging processes are presented in this section for reasons of clarity and better understanding of the materials in this and other chapters. Understanding such terms in advance may ease the actual work to be carried in chapter 5, especially in preparing the statistical database required for the learning process.

This section also presents the formula needed for making calculations necessary to prepare statistical databases. Thus, the processes involved and the calculations to be performed are elaborated in detail considering an ideal pre-tagged corpus. The following notations are used frequently in this chapter.

$C_1\ C_2\ C_3 \ldots\ C_n$ represent (sequence of) categories.

Each C_i, i= 1,2,3, … n represent tags or codes for a certain POS

$w_1\ w_2\ w_3 \ldots\ w_n$ represent sequence of words in a sentence

Please not that most of the things presented in this section are extracted from Allen (1995).

2.4.1 Definitions of Terms and Concepts

Tag

Tags (singular tag) are symbolic representations of word categories, punctuations, delimiters and so on. They are distinct codes to denote the different parts of speech (or word categories) conveniently. The codes N and J are two examples of Old English POS tags for nouns and adjectives.

Tagset

A tagset is a set of tags. One example of tagset is the Penn Treebank tagset[9]. As can be seen from the Penn Treebank tagset or the tagset in table 4.4 of this work, a tagset is a collection of distinct coding (or tags) for all classes of words having distinct grammatical behaviour.

Transition Probabilities

These are probabilities of a tag given one or more previous tags. Transition probabilities are denoted by p ($C_i \backslash C_{i-1} \ldots C_n$). Such probabilities tell us, for instance, the probability of a noun to be preceded by an adjective, p (C_i =Noun$\backslash C_{i-1}$ =Adjective). They can also tell us the probability of a verb to be preceded by a noun, an adjective and a determiner, p (C_i =verb$\backslash$ C_{i-1} =Noun C_{i-2} = adjective C_{i-3} = verb) and so on.

Thus, depending on the value of n, we can have bigram (n=2), trigram (n=3) or in general an n-gram transitional probabilities. If n=2, the model is a bigram model and the notation p ($C_i \backslash C_{i-1} \ldots C_n$) reduces to p ($C_i \backslash C_{i-1}$). If n = 3, the model is called a trigram model and the notation p ($C_i \backslash C_{i-1} \ldots C_n$) reduces to p ($C_i \backslash C_{i-1} C_{i-2}$).

[9] The Penn Tree bank Tagset is listed in Allen (1995)

The bigram model, as described, pairs two categories (or tags), C_i and C_{i-1}, and calculates the probability that a category C_i will follow the category C_{i-1}, written as p ($C_i \backslash C_{i-1}$). This model assumes that the probability of a particular category occurring depends solely on the one category immediately preceding it. The trigram model pairs three categories, C_i, C_{i-1} and C_{i-2}, and calculates the probability that the category (or tag) C_i will follow the two immediate preceding categories, C_{i-1} and C_{i-2}. This is written as p ($C_i \backslash C_{i-1} C_{i-2}$).

The details for calculating such transition probabilities are presented later under step 3 of section 2.4.2. Although trigram probabilities give better results (Alen, 1995), this study assumes bigram transition probabilities in developing the Amharic tagger. That is, all estimates in this study for transitional probabilities are based on the immediate context of words and will not consider any context further than one word away.

Lexical (or emission) Probabilities

These are probabilities of a word given a category (tag) in a given corpora. This is written as p (word\tag) or p (word\category) or shortly as p ($w_i \backslash C_i$). For instance, p (can\N) denotes the (lexical) probability of can to be a **noun,** p (can\MD) denotes the (lexical) probability of can to be a **modal and** p (can\V) denotes the (lexical) probability of can to be a verb in a given pre-tagged corpus. Details on how to calculate lexical probabilities are presented later under step 3 of section 2.4.2 making calculations.

Probabilities of a Particular Part of Speech Sequence

Considering the notations DT, N, MD and V as codes or tags for such POS as determiner, noun, modal and verb respectively, then the notation DT-N-MD-V is an example of a

particular POS sequence among the different possible POS sequences for a certain word sequence w_1 w_2 w_3 w_4.

Thus, having this in mind, the probability of a particular POS sequence is defined as a number (or probability) that is the product of the **transitional** and **lexical** probabilities that the POS sequence contains. As an example to illustrate this definition, consider the sentence **"the garbage can smell"** assuming it is tagged with such parts of speech as follows.

The\DT garbage\N can\MD smell\V

then

- DT-N-MD-V is a POS sequence for the case assumed.
- The probability of this particular part of speech sequence is denoted by p (DT-N-MD-V)
- Considering the bigram model and using the above definition for Probabilities of a particular part of speech sequence, the p (DT-N-MD-V) is calculated as

p (DT-NN-MD-VB)=p (N\DT)P(MD\N)P(V\MD) x p(the\DT)p(garbage\N) p(can\MD) p(small\V).....(1)

Transition probabilities Lexical probabilities

Details on the derivation of this formula are presented in section 2.4.2.

Most Likely Sequence of Tags (or Categories)

A given sequence of words w_1 w_2 w_3 ... w_n can have different allowable sequence of part of speeches tags (or word categories). Each of these allowable sequences of categories has its own probabilities. The **most likely sequence of tags or categories** is the one among these allowable sequences but having the maximum probability. That is, the most likely sequence of tags (or categories) is a **particular POS sequence** having the **maximum probability**

value among the different POS sequence allowable for a given sequence of words. This point will be made clear mathematically later in Section 2.4.2.

In HMM approach, the most likely sequence of tags is considered as the most appropriate sequence of tags (or categories) for the sequence of words in a given sentence. It is this from this sequence of tags that tags will finally be assigned to each word in the input sentence.

Viterbi Algorithm

This is an algorithm that generates the most likely sequence of tags for the sequence of words in a sentence. This algorithm depends heavily on the assumption that the probability of a particular category occurring in a corpus depends solely on the category immediately preceding it. This algorithm helps to determine the optimal path of tags through the search space ranging from one unambiguous tag to the text. This algorithm is employed for the purpose of this study, particularly to search for the most likely sequence of tags.

2.4.2 Steps in POS Tagging

Three things are generally essential to automatically tag (a small) corpora using the HMM approach, an approach selected for this study. These are,

1. Creation of a dictionary listing of the allowable POS for each word in the small corpus
2. Creation of a matrix of lexical probabilities table, and
3. Creation of a matrix of transitional probabilities table

The following steps are in turn considered essential in the creation of the dictionary and the matrix of lexical and transitional probabilities.

Step 1. Preparation of A Tagged Corpus

The tagging process, in this study, begins with a small pre-tagged corpus. This tagged corpus is necessary for it serves as a base to create the dictionary and estimate the probabilities needed for creating the matrix of lexical and transitional probability tables. For the purpose of this study a small sample text is selected from a book by Gethahun (1989)[10]. This small sample corpus is hand tagged with POS by three annotators to make the corpus ready for the next steps.

Step 2. Dictionary (Lexicon)

As can be seen from the formula presented earlier in equation (1), the computation of lexical and transitional probabilities require a summary of word counts in the corpus. A dictionary listing the allowable POS for each word in the corpus will supplement this. This dictionary gives also a statistics on the number of times each word appears in the corpus with a certain category. It gives also the total number of words in the corpus that falls in a certain category.

Step 3. Making the Calculation

The statistical tagger developed in this study follows a birgram-based HMM approach of the kind described in (Allen, 1995). As such the following paragraph present details of the calculations necessary for developing a birgam based HMM taggers and hence the Amharic tagger.

In HMM, the model computes probabilities of co-occurrence of words based on a given tagged corpus and tags texts using these probabilities (Mao, 1997). In calculating such probabilities, the model uses the probability theory often called Markov assumptions. The

[10] See section 5.2 in this thesis for additional information of the sample corpus

model also uses Bayes' rule of conditional probabilities. This rule plays a fundamental role in determining the most likely lexical category for each word in a given sentence. Thus, the section begins with the mathematical definitions of conditional probability, Bayes' theorem, the notion of independence and proceeds step- by- step to more complex computations on lexical, contextual and such probabilities as the probability of the most likely sequence of tags.

Conditional Probability

The general form of conditional probability is given by the formula

P (e\e') = p (e & e')/p (e') .....(2)

Where

1. e and e' are two events
2. P (e\e') is the probability of an event e occurring given the occurrence of another event e'
3. p (e &e') is the probability of the two events e and e' to occur together (at once)
4. P (e') is the probability of the occurrence of event e'.

Bayes' Theorem

Bayes' theorem is an important theorem in that it relates the conditional probabilities of an event A given B, written as p (A\B), to the conditional probability of B given A, written as p (A\B). Mathematically **Bayes'** theorem is written as followes.

$$P(A\backslash B)= \frac{p (B\backslash A) * p (A)}{P (B)} \quad \text{.....(3)}$$

Independent Events

The notion of independent events is also useful for making calculation needed for our task. Informally, independent events are defined as two events in which the occurrence of one does not affect the occurrence of the other. Formally, two events are said to be independent of each other if and only if

$$P(A\backslash B) = p(A) \quad \ldots\ldots(4)$$

Thus, using equation (4), i.e. assuming two independent events A and B, equation (2) can be written as

$$P(A) = p(A\backslash B) = p(A \,\&\, B) / p(B) \quad \ldots\ldots(5)$$

Therefore, for two independent events, equation (2) reduces to

$$P(A \,\&\, B) = p(A) * p(B) \quad \ldots\ldots(6)$$

Equations (2) to (4) are, one way or another, related to the problem of POST. That is, finding the sequence of lexical categories $C_1C_2\ldots$ Cn that maximizes $p(C_1C_2\ldots C_n \backslash w_1\ldots..w_n)$ given the sequence of words $w_1\ w_2\ldots\ldots,w_n$. The procedures to the solution of this problem are exactly the procedures to be followed to develop the Amharic tagger.

$P(C_1\ldots C_n\backslash w_1\ldots w_n)$ is exactly the probability of an event $C_1\ldots$Cn occurring given the occurrence of another event $w_1\ldots w_n$. Using Bayes' theorem, this conditional probability equals

$$(P(w_1\ldots w_n\backslash C_1\ldots C_n) * p(C_1\ldots C_n)) / p(w_1\ldots w_n) \quad \ldots\ldots(7)$$

Since multiplication is commutative, equation (7) can be written as

$$(p(C_1\ldots C_n) * p(w_1\ldots w_n\backslash C_1\ldots C_n)) / p(w_1\ldots w_n) \quad \ldots\ldots(8)$$

Since the aim is to find $C_1 \ldots C_n$ that gives the maximum value for p $(C_1C_{2\ldots}$ Cn \w$_1$...wn) and the denominators in expression (8) are the same for all the cases (i.e. denominators will not affect the answer), the problem reduces to finding the sequence $C_1 \ldots C_n$ that maximizes the formula

$$P\,(C_1 \ldots C_n) * p\,(w_{1 \ldots} w_t \backslash C_1 \ldots C_n) \ldots\ldots\ldots (9)$$

Since calculating the probabilities of these sequences is difficult for it requires far too much data, we turn to approximate it by using probabilities that are simpler. To do this, consider each expression in equation (8) separately, assuming independence. That is we have

$$P\,(C_1 \ldots C_n) \ldots\ldots\ldots (10)$$

$$P\,(w_{1 \ldots} w_t \backslash C_1 \ldots C_n) \ldots\ldots\ldots (11)$$

Approximating p $(C_1 \ldots C_n)$

These are the probabilities of the sequence of categories. Assuming independence and the bigram model, p $(C_1 \ldots C_n)$ can be approximated as follows

$$p\,(C_1 \ldots C_n)\,) \approx p(C_1 \backslash C_0) * p(C_2 \backslash C_1) * p(C_3 \backslash C_2) * \ldots\ldots * p(C_n \backslash C_{n-1})$$

$$= \prod P\,(C_i \backslash C_{i-1})$$

$$\Rightarrow$$

$$P(C_1 \ldots C_n) \approx \prod p\,(C_i \backslash C_{i-1}) \ldots\ldots\ldots (12)$$

To account for the beginning of a sentence, assumption is made to use a pseudocategory, denoted by Φ, at position 0 as the value of C_0, i.e. $C_0 = \Phi$ (Allen, 1995).

As an application of equation (12), the bigram (or transitional) probability of the sequence DT-N-MD-V would be

$$p\ (DT\text{-}N\text{-}MD\text{-}V) = p\ (DT \backslash \Phi) * p\ (N \backslash DT) * p\ (MD \backslash N) * p\ (V \backslash MD)$$

Approximating p ($w_1 \ldots w_n \backslash C_1 \ldots C_n$)

Assuming that a word appears in a category being independent of the words in the preceding or succeeding categories, expression (11) can be approximated by the product of the probability that each word occurs in the indicated POS. That is,

$$P\ (w_1, \ldots\ldots, w_t \backslash\ C_1 \ldots C_n) \approx p\ (w_1 \backslash C_1) * \ p\ (w_2 \backslash C_2) * \ldots * \ p\ (w_n \backslash C_n)$$

$$= \Pi\ P\ (w_i \backslash C_i)$$

$$\Rightarrow$$

$$P\ (w_1, \ldots\ldots, w_t \backslash\ C_1 \ldots C_n) \approx \Pi\ \ p\ (w_i \backslash C_i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (13)$$

Using equations (12) and (13), the expression in (9) can be approximated as

$$P\ (C_1 \ldots C_n) * p\ (w_{1 \ldots} w_t \backslash\ C_1 \ldots C_n) \approx \Pi\ p\ (C_i \backslash C_{i-1}) * p\ (w_i \backslash C_i) \ldots\ldots\ldots\ldots (14)$$

Therefore, the whole problem of finding the sequence of lexical categories $C_1 \ldots C_n$ that maximizes, p ($C_1 \ldots C_n \backslash\ w_1 \ldots w_n$) is reduced to finding a sequence of lexical categories $C_1 \ldots C_n$ where

$$\Pi\ \ p\ (C_i \backslash C_{i-1}) * \ \ p\ (w_i \backslash C_i\ ) \text{ is the maximum}$$

The advantage of using the formula $\Pi\ \ p\ (C_i \backslash C_{i-1})\ * \ p\ (w_i \backslash C_i\ )$ is that the probabilities involved in the formula can easily be estimated from a corpus of text tagged with POS. Expanding $\Pi\ p\ (C_i \backslash C_{i-1}) * \ p\ (w_i \backslash C_i\ )$ results in the following formula

$$\Pi\, p\,(C_i \backslash C_{i-1}) * p\,(w_i \backslash C_i) = p(C_1 \backslash C_0) * p(w_1 \backslash C_0) * p(C_2 \backslash C_1) * p(w_2 \backslash C_2) * \ldots * p(C_n \backslash C_{n-1}) * p(w_n \backslash C_n)$$

$$= p(C_1 \backslash C_0) * p(C_2 \backslash C_1) * \ldots * p(C_n \backslash C_{n-1}) * p(w_1 \backslash C_1) * p(w_2 \backslash c_2) * \ldots * p(w_t \backslash c_t) \ldots \quad (15)$$

Transitional probabilities — Lexical probabilities

Computing Transition Probabilities, i.e. p ($C_i \backslash C_{i-1}$)

Transitional probabilities indicate the probability of one category to follow another category (or other categories). In practice, given a database of texts tagged with part of speech, the bigram or transitional probabilities can be estimated simply by counting the number of times each pair of categories occurs compared to individual category counts. Mathematically this is written as

$$P(C_i = \propto / C_{i-1} = \beta) = \frac{\text{count of the number of times } \propto \text{ and } \beta \text{ occur together in the corpus}}{\text{Number of times } \beta \text{ occurs in the corpus}} \quad \ldots\ldots\ldots(16)$$

Where $\propto$ and β are parts of speech codes.

Thus, using equation 16, the probability that a V (i.e. verb) follows a N (i.e. noun) would be estimated from the tagged corpus as follows.

$$p(C_i = V \backslash C_{i-1} = N) = p(V/N) \approx \frac{\text{the number of times V and N occur together in the corpus}}{\text{Number of times N occurs in the corpus}} \quad \ldots\ldots\ldots(17)$$

Equation (16) is a formula that will generate bigram or transitional probabilities from the training corpora. It is a formula that generates all the transition probabilities required in the process of finding the sequences of categories $C_1 \ldots C_n$ that maximize $\Pi\, p\,(C_i \backslash C_{i-1}) * p\,(w_i \backslash C_i)$. The transitional probabilities that will be obtained using equation (16) will be stored in a database (Matrix of transition probabilities) or be presented using a Markov chain, a chain that capture and illustrate bigram (or transitional) probabilities.

Computing lexical probabilities, i.e. p ($w_i \setminus C_i$)

The lexical generation probability is the probability that a given category is realized by a specific word and is estimated simply by counting the number of occurrence of each word by a category. Mathematically this is given by

$$P(W_i \setminus C_i) = \frac{\text{number of times } W_i \text{ appears in category } C_i}{\text{total number words with category } C_i} \quad \ldots\ldots\ldots\ldots (18)$$

For instance, assuming a certain corpus that contain the word **flower,** then

$$p(\text{flower} \setminus V) = \frac{\text{Number of times flower is used as a verb in the corpus}}{\text{total number of verbs in the corpus}}$$

$$P(\text{flower} \setminus N) = \frac{\text{Number of times flower occurs as a noun in the corpus}}{\text{total number of nouns in the corpus}}$$

Equation (18) generates all lexical probabilities required in the process of finding the sequence of lexical categories $C_i \ldots C_n$ that maximizes

$$\Pi \; p(C_i \setminus C_{i-1}) * p(w_i \setminus C_i)$$

The lexical probabilities obtained using equation (18) are also useful to create the matrix of lexical probabilities.

Step 4. Computing Probabilities of A Particular POS Sequence

Once all possible lexical and transitional probabilities are computed and stored in databases using equation (16) and (18), the next step is to compute probabilities of a particular POS sequence, i.e. p ($C_i \ldots C_n$) which is approximately equal to

$$\Pi \; P(C_i \setminus C_{i-1}) * p(w_i \setminus C_i)$$

The computation on how to find the probability of a particular POS sequence is clearly presented in (Allen, 1995). That will be summarized here to clarify how values presented in the statistical tables, LexicoProb and TransProb, of chapter five of this thesis are obtained. For explanation purpose, consider the corpus used by Allen in (Allen, 1995) [11] together with the following lexical and transitional probabilities calculated from this corpus and obtained from the same source.

Lexical probabilities:

P(flies\N)=0.025,p(like\V)=0.1,p(a\ART)=0.36,p (flower\N)=0.063, p (files\V)=0.076

Contextual probabilities:

p (DT\Φ)=.71, p (N\Φ)=.29, p (N\DT)=1, p(V/N)=.43, p(N\N)=.13, p(P/N)=.44, p(N\V)=.35, p(DT\V)=.65,

Assume now that we need to compute the probability that the sequence N- V- DT-N generates the out put **flies like a flower as**

Flies**N** like**V** a**DT** flower**N** Where N= noun, V= verb DT= article

To deal with this problem, it is necessary to elaborate two terms, Markov assumption and Markov chain.

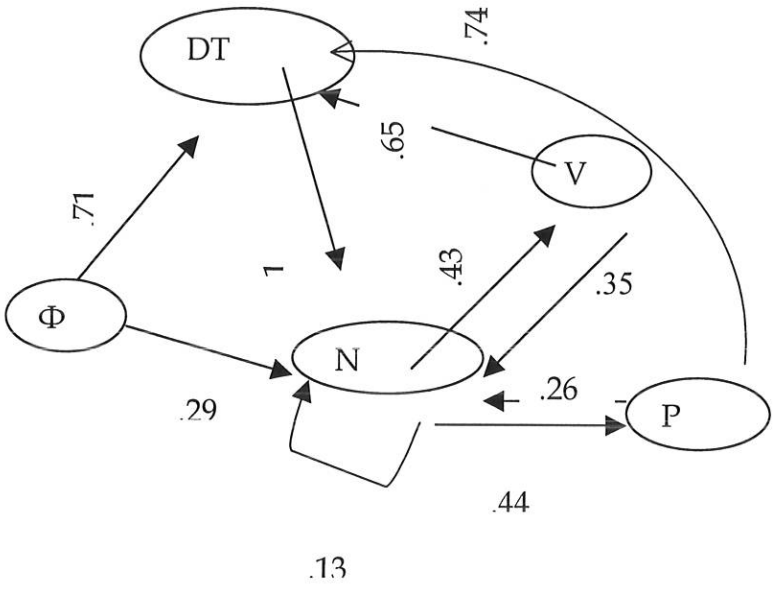
A Markov chain is a special form of probabilistic finite state machine. It is shown in the figure below.

---

[11] The corpus has 300 sentences with 833 nouns, 300 verbs, 558 determiners and 307 prepositions and this counts to a total of 1998 words.

Figure 2.2: Markov chain for bigram probabilities (Allen, 1995)

P= preposition



The following are important points regarding the Markov chain (Allen, 1995).

1. The Markov chain as seen from the figure is a kind of network.
2. The nodes shown represents the possible lexical categories
3. The bigram or transition probabilities, the probability that one category followed another category, are indicated on the arrows.

4. The network can be made to indicate lexical probabilities by allowing each node to have an **output probability (Allen, 1995).** These output probabilities are nothing other than the lexical probabilities of the category at each node. But, such probabilities are not indicted because indicating such probabilities at each node on the network is difficult for the same word can be in different states depending on the category it assumes. Thus, the output

probability at each node can be obtained by referring to the associated matrix of lexical probabilities table.

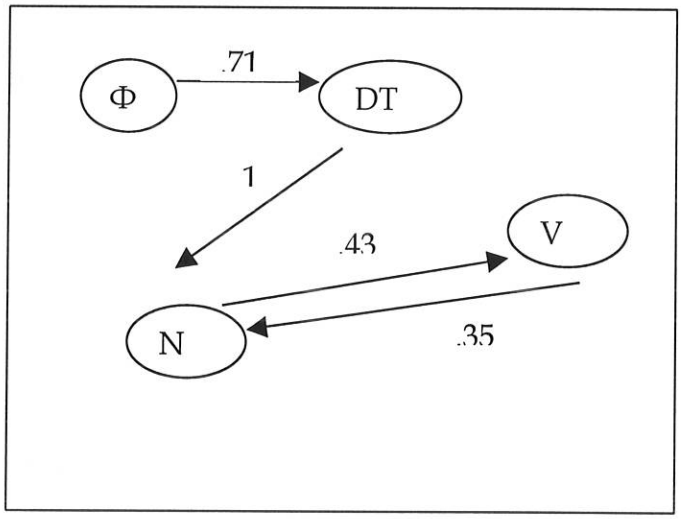
5. Networks like the ones shown above with the output probabilities associated with each node not indicated on the network are called Hidden Markov Model. The model is hidden because for a specific sequence of words, it is not clear what the state of Markov Model is in. For example, the word **flies** could be generated from state N with a certain probability or it could be generated from the state V with a different or in rare cases with the same probability as it is generated from the state N. This implies that computing the transitional probability of a sequence of words from the Network alone, i.e. with out referring to the associated lexical probabilities stored in the matrix of lexical probabilities table, is difficult because of the ambiguity and hence difficulty to identify from which states the word could be generated.

6. The Network is useful to compute **transitional probabilities** of any sequence of categories. To compute transitional probabilities using the Markov chain, simply find the path of the intended (or correct) POS sequence through the network and then multiply the transition probabilities indicated on the path followed. That will give the required transition probabilities of the sequence of categories selected. The advantage of the Markov chain in this sense is that it makes the calculation of **transitional probabilities** easier than using equation (12) together with the matrix of transition probabilities.

Using the above network, for instance, we can find the **transition probability of such sequence of categories** as DT- N- V- N as follows

$$P(DT\text{-}N\text{-}V\text{-}N) = p(DT\backslash\Phi) * p(N\backslash DT) * p(VB\backslash N) * p(N/V) \ldots\ldots \ldots\ldots (19)$$

As explained, the values of $p(DT\backslash\Phi)$, $p(N\backslash DT)$, $p(V/N)$ and $p(N\backslash V)$ can easily be obtained following the path on the Markov chain. Such path is shown in the next page for clarity



Thus, p (DT- N- VB- N) = p (DT\Φ) * p (N\DT) * p (V\N) * p (N/V)

= .71 * 1 * .43 * .35

≈ 0.107

That is, the **transition probability of the particular sequence of categories** DT N V N is 0.107. Similarly, transition probabilities for other sequence of categories can be computed using the Markov chain. In all these discussions, the **Marko assumption** that the probability of a category occurring depends only on the one category before it is assumed.

The entire avenue taken in this chapter are mainly for one purpose, to find the highly sequence of categories $C_1C_2 \ldots Cn$ that are to be assigned to the sequence of words $w_1w_2 \ldots w_n$, respectively. But, before we come to this point, one more step is left. This is on how to find the **probability of a particular sequence categories**, which is given by

$$\Pi \; p(C_i \backslash C_{i-1}) * \; p(w_i \backslash C_i)$$

The problem on how to find the **probability of a particular sequence categories** is illustrated below considering the sentence "**Flies like a flower**" together with the information on the lexical and transitional probabilities that are taken from Allen and that are already used in this and earlier sections.

For the sentence "**Flies like a flower**", the sequence of words become w_1= flies, w_2=like, w_3=a and w_4=flower. Considering also N V DT N as one possible sequence of categories for such sequence of words, the sequence of categories become C_1=N, C_2=V and C_3=DT and C_4=N. C_0 is assigned to Φ, i.e. C_0= Φ. Then the **probability of this particular sequence of categories** is computed as

$$p(N\text{-}V\text{-}DT\text{-}N) \approx \prod p(C_i \backslash C_{i-1}) * p(w_1 \backslash C_i)$$

$$= p(C_i \backslash C_0) * p(w_i \backslash C_i) * p(C_2 \backslash C_1) * p(w_2 \backslash C_2) * p(C_3 \backslash C_2) * p(w_3 \backslash C_3) * p(C_4 \backslash C_3) * p(w_4 \backslash C_4)$$

$$= p(C_i \backslash C_0) * p(C_2 \backslash C_1) * p(C_3 \backslash C_2) * p(C_4 \backslash C_3) * p(w_1 \backslash C_1) * p(w_2 \backslash C_2) * p(w_3 \backslash C_3) * p(w_4 \backslash C_4)$$

$$= p(N \backslash \Phi) * p(V \backslash N) * P(ART \backslash V) * P(N \backslash ART) * P(Files \backslash N) * P(Like \backslash V) * P(a \backslash ART) * P(Flower \backslash N)$$

$$= (.29 * .43 * .65 * 1) * (0.025 * 0.1 * 0.36 * 0.063)$$

$$\approx 0.081 * 5.67 \times 10^{-5}$$

$$= 4.5 \times 10^{-6}$$

Therefore p (N- V- DT- N)= 4.5×10^{-6}

The probability for all other sequence of categories that are permissible for the sentence **flies like a flower** can be computed following similar procedures. The sequence of categories with the maximum of such probabilities is the most likely sequence of categories we opted for.

Note that, the transition probabilities in the calculation can be obtained either using the Markov chain or the matrix of transition probabilities table. The lexical probabilities (the output probabilities) can only be found from the probability table for the lexical probabilities.

Note also that, although the **transitional probabilities** of a **sequence of categories** and the **probability of a sequence of categories** have the same formula, i.e. p (DT N V N), the two are quite different (see equation 19 and 20). This difference is clear from the formula used to compute each of them and also from the results obtained using the formula.

Finding The Most Likely Sequence of Categories (tags)

The final task in tagging a given sentence (or corpus) using the HMM approach is to find the most likely sequence of categories among the sequence of categories that are possible for a given sequence of words. As defined earlier, this sequence is the one with the highest probability out of all possible sequence of categories for a given sequence of words. Finding this sequence of categories might seem very difficult, as the number of possibilities may be very large depending on the number of words and the number of parts of speech. For instance, if there are only four parts of speech and we want to tag four sequences of words with their appropriate POS, the number of possible sequence of categories will be 4^4=256. Finding the most likely sequence of categories out of such 256 possible sequences of categories is really a huge task. Fortunately, due to Markov assumption and the fact that sequences that end in the same category can be collapsed together (since the next category depends on the one previous category in the sequence) there is no need to enumerate all possible sequences to find the most likely sequence of categories for a given sequence of words. This is well discussed in (Allen, 1995).

The method on how to find the most likely sequence of categories for the given sequence of words, as discussed in (Allen, 1995) using **flies like a flower** may be summarized as follows.

Sweep forward through the words one at a time finding the most likely sequence for each ending category. In other words, you first find the four best sequences for the two words, **flies like**, the best ending with **like** as a V, the best as an N, the best as a P, and the best as DT. You then use this information to find the four best sequences for the three words **flies like a**, each one ending in a different category. Repeat this process until all the words in the sentence are accounted for. Even the simplified version of the process seems complicated when done manually. Fortunately, there is an algorithm that will do this for us. This algorithm is the Viterbi algorithm.

To this end, the whole task is reduced only to prepare the dictionary, the lexical and transitional probabilities tables and may be modifying the Viterbi algorithm to fit for Amharic language. The Viterbi algorithm will handle the task of finding the most likely sequence of categories. Once the most likely sequence of categories are identified using the Viterbi algorithm, then that sequence will be considered as the best tag sequence and assigned to the sequence of words in the sentence. Details on Viterbi algorithm are presented in (Alen, 1995) and chapter 5 of this thesis. This will conclude **the detailed treatment of the approach**.

2.5 Conclusion

Discussions were made in this chapter on various issues. The various issues and statistical considerations included in this chapter will practically be used in chapter five to Create a dictionary listing, lexical probability matrix, transition probability matrix and develop the tagger. The next chapter will discuss word classes in Amharic.

CHAPTER THREE

The Structure of Amharic Word Classes

3.1. Introduction

As outlined in the first chapter, this chapter discusses the structure of the Amharic word classes. Among the word classes discussed in this chapter are nouns, verbs, adjectives, adverbs, prepositions and conjunctions. Pronouns are also discussed separately but they fall under noun category. Other Amharic words such as interjections and numerals are also topics discussed in this chapter. To the better understanding the material in this section the chapter begins with a brief review of the writing system and punctuation marks in Amharic.

The analysis and discussions made in this chapter are based on the data extracted from Abiyot (2000), Baye (1987)[12], Getahun (1990) [17], Hirut (1998) and Dawkins (1969). Further details on the subject can be obtained from these sources.

3.2 The Amharic Writing System

3.2.1 The Amharic Alphabet

In this study, the Latin letters are used to represent the Amharic alphabet assuming that it is possible to develop a program that converts Amharic texts, written using the Amharic fidel, to its equivalent Latin form. A list of the Amharic alphabet (fidel) adopted from Leslau and used in this study is found in appendix 3.1. Detaild discussion on Amharic writing systems is found in (Abiyot, 2000).

[12] The year is according to the Ethiopian calender

3.2.2 Punctuation Marks in Amharic

Analysis of Amharic texts reveals that different Amharic punctuation marks are used in Amharic to serve for different purposes. Apendix 3.2 sumeraizes such punctuation marks together with the purpose for which they are used.

3.3 Word Categories in Amharic

For the purpose of this study, works in the area of word categorization are broadly divided into two, namely, " early" and "recent" works.

In early works (Mersi'hazen, 1934) [13] Amharic words are categorized into the following eight categories (or classes or parts of speech). These are the noun, Verb, Adjective, Adverb Preposition, Pronoun, Conjunction and Interjection categories.

In recent works such as Baye (1987)[13] the early categorization of Amharic words is reduced into five putting **pronouns** and **conjunctions** under the **noun** and **preposition** categories respectively. In this categorization **interjections,** which are words with out syntactic functions, are not considered as parts of speech[14].

In this study, the classification by the early scholars is adopted but treating nouns and pronouns in the same POS category as suggested by Baye. This preference is made for the early categorization is more exhaustive and it allows the tagger to tag words exhaustively. The remaining sections of this chapter are devoted to discuss the different POS categories

[13] Date is according to the Ethiopian calendar
[14] See Baye for such categorization and why interjections are not considered as parts of speech

mentioned earlier to lay a background reading for works to be carried out in chapter four and five, which are the central and major contribution of this thesis.

3.4 The Amharic Noun Class

Like English, Amharic nouns are words used to name or identify any of a class of things, people, places or ideas or a particular one of these. For this study, the Amharic noun class consists of nouns and pronouns. These are discussed in some detail in appendix 3.3.

3.5 The Amharic verb class

Most of the information presented in this section is drawn from the data compiled by Abiyot (2000). His work consists most of the information required for the purpose of this study. Due to the requrment imposed by the Postgraguate School to limit a Masters thesis to 150 pages, the review made on the Amharic verb class could not be included her. The Amharic verb Class is discussed in some detail in appendix 3.4.

3.6 The Adjective Class

Adjectives in Amharic usually precede the nouns that they modify or describe.

Example **sänäf** tamari näw "he is a lazy student"

In this example, the adjective sänäf "lazy" precedes the noun tämari "student" which it modifies. But this does not mean that a word is an adjective just because it precedes a noun. For instance, in **yɔ̀h bäg** "This sheep", the word yɔ̀h "This" precedes the noun bäg 'Sheep'. Although the word **yɔ̀h** functionally shares the feature of an adjective (modifier), it is a pronoun, a demonstrative pronoun. Like nouns Amharic adjectives can be either primitive or derived. These and other features of the Amharic adjectives are discussed in appendix 3.5.

3.7 Prepositions in Amharic

The term preposition[15] refers to words, which will have meaning only when they are attached or used together with other words such as nouns, verbs, pronouns and adjectives. Prepositions can appear as

- a simple prepositions that stand alone as separate words

Examples	**sɔ̀lä** tɔ̀mhɔ̀rt	"because of education"
	Wädä bet	"to the house or home"

- a simple prepositions prefixed or attached with other words (e.g. nouns and verbs).

Example	**bä**-mäkina	"by car"
	lä-hizb	" to/for the public"

- As compound prepositions consisting of two parts, prepositional prefixes and post positions put after nouns. The postpositions can either be single preposition that stand by their owen or a preposition not separated from a noun.

Examples **bä**-sat'n-u wɔ̀st "Inside the box"

Preposition box (noun) a postposition (noun)- inside

Kä-angät **bä-lay** "Above the neck"

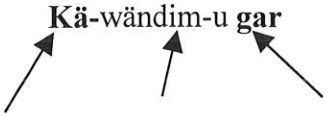
Preposition noun (neck) preposition noun

[15] Two types of prepositions can be recognized in Amharic depending on weather they are bound or free. See (Bay, 1987) and (Gethahun, 1990) for definitions of free and bound prepositions.

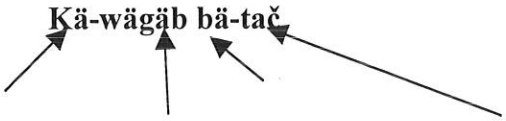
lay in the above example is not a preposition. It is a noun and the whole **bä-lay** is considered as a preposition not separated from a noun. In some cases the whole **bä-lay** is treated as an adverbial nouns.

Preposition noun (tree) preposition

Here **lay** is used as a preposition rather than as a noun.



Preposition noun (brother) preposition



Preposition noun (waist) preposition noun (below)

The whole bä-tač is a noun adverb composed of a preposition not separated from the noun (Baye, 1986)

3.8 The Adverb Class in Amharic

In Amharic, adverbs can be found in either primitive forms (i.e. as separate words that appear by their own) or in compound forms as combinations of prepositions and some other words but that appear as separate or in rare cases as compound words. In each case they refer to place, time, circumstance etc. A long list of such adverbs is found in (Dawkins, 1969).

There are also what are called adverbs of position, which may be referred to as noun adverbs. Noun adverbs can be used either as a noun or as an adverb depending on their context.

Examples wädä **wɔ̀st'** gäbä "He entered inside" wɔ̀st' here is used as a noun

(bä) wɔ̀st' yɔ̀säral "He works in" wɔ̀st' is used here as an adverb

As can be seen from the examples and mentioned above, noun adverbs (or adverbs of position) such as wɔ̀st "inside" can function alone as an adverb or as a noun. But they often have the locative particle **bä** when they are used adverbially. If such noun adverbs are used alone without such prepositions (as **bä, wädä, kä)** they are mostly treated as nouns. On the other hand, noun adverbs prefixed with prepositions are treated as adverbs. That is, wɔ̀st is a noun while bä-wɔ̀st' is an adverb. A list of noun adverbs is also found in (Dawkins, 1969).

Adverbs can also be formed from nouns and adjectives by prefixing { **bä-** } as in **bä**-hayɔ̀l "by force" and **bä**-dähna "being well" respectively. Adverbs formed in this manner are short adverbial clauses. That is, the resulting words are more of adverbs and are considered as adverbs.

Days of the week in Amharic language may be used also either as a noun or as an adverb.

3.9 Conjunctions in Amharic

Conjunctions in Amharic are coordinating or subordinating. They coordinate words, phrases, clause and sentences. A list of Amharic coordinating conjunctions is found in (Dawkins, 1969) together with a detailed discussion on such coordinating and subordinating conjunctions.

Concerning conjunctions there are two views. In early works (e.g. Marsehazn, 1934), conjunctions were viewed as appearing in their own Category. But in recent works (Baye, 1987; Getahun, 1990), they are categorized in the same category as prepositions.

This paper adopts the early view that conjunctions and prepositions appear in different categories, the preposition and conjunction parts of speech categories. One problem that arises by categorizing prepositions and conjunctions into different classes is the problem pertaining to distinguish conjunctions from prepositions. The problem in distinguishing the two mainly arise from the fact that the same words are mostly used as both prepositions and conjunctions. sɔ̀lä, bä and kä- are, for instance, both prepositions and conjunctions. One may think that this will create a problem during the tagging process. But this will not be a problem for the problemit can be minimized if the manual tagging is done very carefully by good annotators, as the two can be identified from the context thy are used. Apart from this problem, conjunctions can be handled following the treatments or approaches discussed earlier for prepositions.

3.10 Numerals

These are words representing numbers. They can be **cardinal** or **ordinal** numbers. A list of the Amharic Cardinal numbers is found in (Dawkins, 1969; Leslau, 1973). In Amharic, the ordinal numbers are formed from the cardinal numbers by suffixing the suffix {**–(ä) ňňa** }.

Example	**Cardinal**	**gloss**	**ordinal**	**gloss**
	hulätt	two	hulät- äňňa	second
	assɔ̀r	ten	assɔ̀r- äňňa	tenth

Like English, compound Amharic numerals are put separately. The following are examples to illustrate this.

Example	hulät mäto sälasa and	"two hundred thirty one"
	hulät mäto sälasa and-ňňa	"231th "

In Amharic, there are also numerals that indicate distribution. These numerals are called distributive numerals.

Example **hulätt hulatt** "two two"

There are also special numerals in Amharic that correspond to the English "half", "quarter" etc.

Examples of these include **gəmaš** "half" and **siso** "one third".

3.11 Interjections

Like English, Amharic has many words or phrases used to express such emotions as sudden surprise, pleasure, annoyance and so on. Such Amharic words are called interjections. These Amharic interjections can stand-alone by themselves outside a sentence or can appear any where in a sentence.

Examples **goš !**

Goš ! abbate mäťťa

abbate mäťťa **Goš !**

səah ťəru näw **Goš !** ləje

A long list of Amharic interjections is found in (Dawkins, 1969).

3.12 Conclusion

This chapter pointed out that, based on recent and early works, Amharic words are categorized in to eight or five parts of speech category. For the purpose of this study, words in Amharic are categorized into seven parts of speech, namely

Nouns	verbs	conjunctions	adverbs
Adjectives	prepositions	interjections	

That is, in this study pronouns are treated as nouns and hence do not have their own part of speech category. This categorization is solely based on what is forwarded by early and recent scholars and also considering conveniences for tagging. In some cases, there might be cases that may not go in line with structural linguistics. The next chapter presents tages and tagset for Amharic word classes.

CHAPTER FOUR

Tags and TagSet for Amharic Word Classes

4. 1. Introduction

This chapter discusses the **POS tags** designed for the Amaharic language and the **tagset** identified for the purpose. The design is mainly based on the properties of the language discussed in chapter three but keeping in mind the Hidden Markove Model approach discussed in chapter two.

This chapter and chapter five are the core of this study. This chapter focuses on drawing the **POS tags** and the **tagset** required for the language and hence for the tagger. **Problems faced** while attempting to tag an Amharic corpus and **possible alternatives** suggested to deal with the problems are also part of the discussion in this chapter.

The chapter begins by briefly introducing knowledge acquisition and appearances of tags in a corpus.

4.2. Knowledge Acquisition

In order to come up with a tagger that annotates a corpus in a given language, it requires to have a secured knowledge of the language to which the tagger is designed. Such knowledge, among other things, may be obtained in various ways. The knowledge required for the NLP in general and POS tagging in particular can be obtained in different ways. In this study, **statistical information** obtained from analysis of NL was used as a knowledge base. Little or

no effort was made to include knowledge of the grammar of the language. Even if the knowledge of the grammar plays a central role in developing an efficient tagger, it is not fully considered in this work due, mainly, to limitation of time.

4.3. Appearance of Word Class Tags

In what follows, the POS tags appear in the form of **word\tag** followed by a single space. The following example illustrates the appearance or general format of a tagged sentence.

Zare\ADV hämus\N näw\ Aux::\PUNC "Today is Thursday"

For the sake of clarity and convenience, tags will only be assigned to words under consideration rather than the whole words appearing in the examples as illustrated in the example below, which considers discussions on nouns. The punctuation mark that marks the end of an Amharic sentence is also omitted from all examples.

Zare hämus\N näw "Today is Thursday"

4.4 Tags for Nouns

4.4.1 General Tags for Nouns

As discussed in Chapter three, nouns are considered as noun proper[16] and pronouns. Nouns can be singular or plural. They can also indicate gender and case. Table 4.1 below shows the attributes of Amharic nouns together with their respective values.

[16] are all nouns that are different from any of the pronouns discussed in Chapter three

Table 4.1: Attribute and attribute values of nouns in Amharic

Word Class: Noun				
Attributes	Gender	Number	Case	Definiteness
Values	{m, f, n}	{s, pl}	(Subject, object, genitive}	{-u, -wa, -itu,-wu}

For the purpose of this study, all nouns other than those treated under **special tags for nouns,** discussed later in this section, are assigned the tag " **N**". That is, there is no such distinction as noun plural, noun singular, common noun, proper noun, and so on. Not that all pronouns (i.e. personal, demonstrative, interrogative, indefinite, impersonal, emphatic and reflexive, reciprocal, distributive and relative pronouns) are also treated as nouns and thus are assigned the same tag **N.** Here are some examples.

∂ssu\N bäg\N gäzä	" He bought a sheep(s)"
∂ne\N bäg-occ\N gäzahu	" I bought sheep(pl)"
yohäns\N bät'äm l∂j\N näw	"Yohans is very young"
d∂mät-it-u\N mot-äč	"The cat(f) died"
d∂mät-u\N mot-ä	"The cat(m) died"

For the purpose of this study, proper nouns like **personal names, geographic names, non-personal names** and **non-geographic names** (e.g. names of organizations, sport names, shops etc) are assigned the tag N as in the following examples.

kassa\N tämari näw	"Kassa is a student"
ityoppiya bäm∂sraq afrika\N t∂gäňaläč	"Ethiopia is found in the East Africa"
mut bähar\N	"The Dead Sea"

Amharic initials in names such as **ato** "Mr.", **Wäyzäro** "Mrs.", **Wäyzärit** " Miss", wätädär "sergon", **lɔj** " prince ", **nɔgɔst** "Queen" **nɔgus** "king" etc are all assigned the tag N as in the following example.

lɔj\N eyasu mäče ɔndä motu aytawäqm "It's unknown when lɔj Eyassu had died"

Days of the week in Amharic can either be used as a noun or as an adverb. When they are used as a noun they are assigned the tag N as in.

ɔhud\N yäräft qän näw "Sunday is a weekend"

Months of the year in Amharic are assigned the tag N as in

mäskäräm\N bä-ityoppɔya yä-mäjämäriya wär näw "September is the first month of the year in Ethiopia"

Directions in Amharic such as sämen "North", däbub "South", mɔsrak "East", mɔrab "West", sämen mɔsrak "North East" etc are also assigned the tag N. The following are some examples of this.

sämen\N ityo**pp**ɔya tärarama näw "The northern part of Ethiopia is mountainous"

däbubu\N yähägäritu kɔfl läm näw "The southern part of the country is fertile"

Compound nouns, which appear either as **multi words**[17] or **fused words,**[18] are assigned the tag N. In case of multi words as **bunna bet** "bar", the tag N will be assigned only once at the end of the multi word rather than assigning the individual components of the multi word, as in

abbatu tɔllɔq **bunna bet\N** aläw "His father has a big bar".

bäg tärä\N täqät'älä "The sheep market place has burnt"

---

[17] are two separate words formed from two words and that belongs to only one POS (eg. buna bet "bar")

[18] are words formed from two words and that they are not separated (eg. betäkɔrɔstiyan "church", which is obtained from bet "house" and kɔrɔstiyan "Christian")

4.4.2 Special Tags for Nouns

As discussed in chapter three, there are Amharic nouns where prepositions and conjunctions are not separated from the nouns. For our purpose, nouns affixed with such prepositions or conjunctions are assigned special tags. The tag NP is used for nouns where prepositions are not separated from nouns. Of particular interest here are Amharic independent personal pronouns attached with prepositions, for the prepositions may not be noticed from the independent personal pronouns. The following are examples of such Amharic independent personal pronouns prefixed with prepositions

(ye)(yä)(slä)(∂ndä)(bä)(kä)(lä)- ∂**ne**

(ye)(yä) (slä)(∂ndä)(bä)(kä)(lä)-**∂ňňa** or **ňňa**

In the language **at most** one of the prepositions can be used at a time with the independent personal pronouns as yä-∂ne "mine", lä-∂ne "to me", s∂lä-∂ne "because of me"

Due to their very nature, such words may not be treated as words that can be categorized in one of the parts of speech categories. Thus, for our purpose such words are tagged the tag NP, a tag for prepositions that are not separated from nouns. Here are some examples

kassa **bä**-mäk∂na\NP mät'a	"kassa came by car"
h∂zbu **lä**-hägäru\NP y∂motal	"Everybody will die for his mother land"
lä(∂)ne\NP b∂lo käsäw tät'ala	"He quarrelled with some one in favour of me"
yä-∂ras-ačäw-n\NP bet šätu	"They sold their house"

Another tag NC is used for nouns suffixed with conjunctions, i.e. where the conjunctions are not separated from the nouns. Here are some examples.

b∂rtukan lomi mango-**na**\NC muz y∂bäq∂lal	"Firuits like orange, lemon mango and banana grew here"
∂ssu mät'ť'ä wänd∂mmu-**m**\NC	"He and his brother came"

Nouns prefixed with balä form special nouns, which show ownership. Such nouns can be identified easily and are assigned the tag NB, as follows

balä-suq-u\NB yäläm "The shopkeeper is not there(or here)"

balä-bäg\NB yɔ̂h bäg sɔ̂nt näw ? "Balä-bäg ! what's the cost of this sheep?"

All **verbal nouns**, nouns formed from verbs, except those formed using **balä** are assigned the tag NV as in

mä-blat\NV yasdästal "Eating food is pleasant"

4.5 Tags for Verbs

4.5.1 General Tags for Verbs

Verbs in Amharic have those attributes and attribute values indicated in the table below.

Table 4.2: Attributes and values of attributes for verbs

Category: Verb					
Attributes	Gender	Number	Person	Tense	Polarity
Values	{m, f, n}	{s, pl }	{1st, 2nd, 3rd}	{1,2,3,4,5, 6}[19]	{Aff., Neg.}[20]

In this study, part of speech tags for verbs do not as such indicate any of the features (or attributes) listed in the above table. That is, no special distinction is made on number, person, gender, tense and polarity. Only two tags are used in this study for verbs other than those indicated by the special tags, (see beyond). These tags are **V** and **AUX**. The tag V is assigned to verbs in all forms except auxiliary and to those indicated by special tags for verbs. The following are examples to illustrate verb tagging.

[19] The numbers indicate the six tenses in Amharic. For details see Baye(1987)
[20] Are abbreviations for affirmatives and negatives, respectively

aster wädä bet **hed-äč**\V perfect, positive, third person singular feminine gender "Aster has gone to the house"

ɔssu sɔra **al-hed-ä-m**\V perfect , negative, third person singular masculine gender "He hasn't gone to work"

lɔj-očč-u-n **gädäl-u-ačäw**\V perfect , positive, third person plural and neuter for gender "They massacred the children"

ɔnnantä bɔrrċɔqqo **t-säbr-u**\V näbär imperfect , positive, second person plural neuter for gender "you were breaking glasses"

kassa ɔnċät **y-fält**\V näbbär "kassa was splitting wood"

The tag AUX is used to indicate auxiliary verbs in all forms. This tag makes also no special distinction for number, gender, tense etc. Words that are assigned the tag AUX include the following.

- The word allä, näw and näbbär
- All forms of allä, näw and näbbär (words presented in tables 3.10, 3.11 and 3.12 in the appendix).
- All forms of the negatives of allä, näw and näbbär (words presented in table 3.13 in the appendix). Here are some examples

  ɔnnässu tämari **hon-äw-al**\AUX "They became students."

  mɔnɔm säw **al-näbbär-ä-m**\AUX "There was no one."

  ɔňňa anbäsa gadl-ä-n **näbbär**\AUX "We killed a lion."

  antä anbäsa t-gadl **aläh**\AUX "You are strong enough to kill a lion."

  ɔssu ɔbet **allä**\AUX "He is in the house"

In cases where the auxiliary verbs appear as multi words as in nore näbär, noräh näbbär (see column 4 of table 3.11 and 3.12 in the aapendix), the tag AUX is assigned once for each component of the multi word, as shown below.

ɔzzih bet nore\AUX näbär\AUX "He used to live in this house."

lät'ʼəqit gize əzzh honäw\AUX näbär\AUX "They were here for a short period of time."

While tagging auxiliary verbs, problems may be encountered if the main verbs are fused with (i.e. not separated from) the auxiliary verbs, as in the following example.

əssu anbäsa **y-gadl-al** "He kills a lion."

In such cases the whole word is assigned the tag V for the auxiliary verb is attached only to indicate tense. Thus, we have the following.

əssu anbäsa **y-gadl-al**\V "He kills a lion."

Please note again that, for the purpose of this work, the two tags, V and AUX, **simply identify whether the word is a verb or not**. That is, the tags V and AUX are applied more widely, and these verb tags are not made to indicate tenses, number, gender, person and polarity.

Note also that the two tags are not used in case where the verb is affixed with either by prepositions or conjunctions. The two tags are assigned only to such verbs that occur without any prepositions or conjunctions attached to them. Such verbs, verbs affixed with prepositions or conjunctions, are treated separately under special tags at end of this section.

For the sake of this study, compound verbs are treated distinctively from other verbs and are assigned a different tag, VC. Words in this category include

- All forms of allä verbs
- Compounds of adärrägä, assäňä and all their other forms

Note that compounds of allä, adärrägä, assäňä and all their other forms are multiple words. Thus, such words are assigned the tag VC only once at the end of the multiple word combination. Here are some examples

bärru **kəffət allä**\VC "The door is opening wide"

tamari-^{w}očč-u t'əyaqe siť'ayäqq-u **zim y-l-al-u**\VC "The students do not utter a word when they are asked questions."

zəm **i-yal-ačhu**\VC atasċärsun "Don't let us perish in vain."

betun **fərs adrrigo**\VC adis gänäba "He demolished the farmer house and built a new one."

ləj-u-n **zəm assäň-äč-w**\VC "She made the child silent."

Remarks

- adärrägä and assäňä are neither auxiliary nor compound verbs and they are as such assigned the tag V.
- If the compounds of **allä**, adärrägä and assäňä appear with forms of the auxiliary verbs allä and näbbär (see table 3.15 in the appendix), then such words are assigned the tag VC while the forms of the auxiliary verbs **allä** and **näbbär** are assigned the usual tag AUX, as in the following.

  ləj-u-n **şäť assäňtot**\VC **näbbär**\AUX "He made the boy shut his mouth."

  at'ərun **fərs adrəgaw**\VC **al-äč**\AUX "She demolish the fence."

- Nouns formed from compounds of allä, adärrägä and assäňä are assigned the usual tag for verbal nouns, NV, as in any other verbal nouns. The following are examples on tagging verbal nouns derived from allä, adärrägä and assaňa.

  hulum **şäť bay-očč**\NV mähon yäläb-ačä-w-m "All of them should not be silent."

lätäfäťäräw čəgr **şäť bay**\NV mähon yäläb-n-m "We should not be silent for the sudden problem

- One problem worth mentioning in this regard is what to do when the compounds of alä, adärrägä and assäňä are fused with (or not separated from) the auxiliary verbs allä, näw and näbbär, as in

  əskä gizzew dəräs **şäť bəläw-al** "They are silent for the time being."

In such cases, the whole word is assigned the tag VC, for the auxiliary verb is attached to indicate tense, as in

əskä gizzew dəräs **şäť bəläw-al**\VC "They are silent until the time comes to speak their mind."

4.5.2 Special Tags for Verbs

- Verbs (main or auxiliary) suffixed or prefixed with **conjunctions** are assigned the tag **VC**, as in for instance

əncät **sə**-säbr\ VC näbär	"I was cutting woods."
zare andärəsm mängädu əruk naw-**na**\VC	"We can not reach today, it is too far."
təməhrətun ə**sk**təcärs\VC əťäbqalähu	"I will wait until she (or you finish)the lesson"

- Verbs (main or auxiliary) suffixed with **preposition** are assigned the tag **VP**, as follows

ləju **ənda**-mäťťa\VP taňa	"The boy slept as soon as he arrived home."
abbate **kä**-mäťťa\VP ťəraň	"If daddy comes, please do call me."
ayäru qäzqäzä **səlä**-näbbär\VP kä-bet al-wäťäm	"Since the weather was too cold, I couldn't come out."
qäns **əyalä**\VP **sə**-kärakär\VP qoyä	"He was nagging for a discount."
wädä gäbäya **li**-hed\VP tänäsa	"He is ready to go to the market."

4.6 Tags for Adjectives

4.6.1 General Tags for Adjectives

Amharic adjectives have the attributes and values of attributes indicated in the table below.

Table 4.3: Attributes and values of attributes for adjectives

Category: Adjective			
Attributes	Gender	Number	Definiteness
Values	{m, f, n}	{s, pl}	{-u, -itu, -wa}

The general tag assigned for all sorts of adjectives (sing., pl., masc., fem., etc.), for the purpose of this work, is **J**. However, the tag J is used more generally and as such it does not indicate the specific features listed in Table 4.3. The following are some examples that illustrate tagging with J.

däg\J säw näw	"He is a kind person."
bet-ä-säb-očč-u **däg-očč**\J näčäw	"His families are kind."
dähnawa\J set ∂sswa näčč	"She is the kind hearted one."
däg-it-u\J l∂j mäťť-äčč	"The kind hearted child(f) is coming."
däg-u\J l∂j mäťť-ä	"The kind hearted boy came."
däg-oč-u\J l∂j-očč mäťť-u	"The kind hearted teenagers came."

Compound adjectives, as discussed in chapter three, can either be fused words (the two words, forming a compound, word not separated) or multiple word combinations. Compound adjectives that are multi word combinations are assigned the tag J once at the end of the

compound adjectives (i.e. multiple word combination) rather than to each component of the compound adjectives. Compound adjectives that are not multiple word combinations are assigned the tag J, like an adjective that is not compound. Here are examples on tagging compound adjectives.

lɔj-u **g^{w}azäbɔzu**\J näw "The lad is always having loads of luggage."

lɔj-u **gädä bis**\J näw "He is unlucky boy."

Special Tags for Adjectives

➢ If a conjunction is suffixed to and not separated from an adjective, then the adjective together with the conjunctions suffixed to it is assigned the tag **JC,** as shown below.

däg-**na**\ JC yäwah säw näw "He is kind and innocence."

tɔqur-**na**\ JC načč bäg "A sheep with black and white complexion."

ɔrajɔm-**na**\ JC wäfram\J lɔj mäťťä "A tall and fat man is coming."

➢ An adjective prefixed with (and not separated from) a preposition is assigned the tag **JP** as shown below.

lɔj-očč-u **bä**-dɔfɔrät\JP ťʼɔyaqeačäwn aqäräbu "The boys posed their question boldly."

kä-wuhäma\JP bota däräsu "They reached at watery land."

ɔsräň-očč-u **bä**-ťenama\JP huneta lay načäw "The prisoners are very well."

bä-bahlawi\JP gabča-na **bä**-zämänaw\JP gabča mäkakäl lɔyunät alä "There is a difference between the traditional and modern way of wedding."

➢ Some times a noun headed by a preposition function as an adjective. In this case, the tag JPN is assigned to the adjective so formed as in the following.

yä-t'älla bɔ̀rrċɔ̀qqo gäza "Please, do buy a glass for "tella", a local drink."

yä- čayina sähan šäťa "He sold a china made plate"

➢ Numerals in a corpus mostly function as an adjective. The tag JNU is assigned for such numerals, as in the following

and\JNU t'ärmus t'äj t'äťa "Do take a bottle of 'tej'."

hulätt\JNU guräno bäg alläw "He has got two enclosures of sheep."

4.7 Tags for Prepositions and their Problems

In the Amharic language, although most words appear exactly in a certain POS category, there are a lot others that cannot be placed in a certain POS category. Most such words contain prepositions and conjunctions.

For instance, in the sentence **kassa bä-mäkina mäť'ä** "kassa came by car", the word bä-mäkina "by car" is composed of the preposition bä "by" and the noun mäkina "car". But, the whole word bä-mäkina belongs neither to the noun nor to the preposition POS category. In fact, the word does not belong to any of the word categories considered in this study. In other words, the word bä-mäkina cannot in particular be assigned the tag N, used for nouns, or PREP, the tag used for prepositions, see below. This is one of the problems encountered in the attempt of tagging Amharic texts. Similar problem also occur for conjunctions.

The problems mentioned above arise because the prepositions and conjunctions are basically not separated from such words as nouns, verbs and adjectives. This is not the case for the English language where each preposition and conjunction appear separately from other words. Thus it seems that this nature of the language makes the Amharic language to have more tags

than English does. This is a disadvantage for having more tags is not recommended as it has overhead on the performance of the tagger.

A closer look at the grammar of the language reveals that words like bämäkina are treated at phrase level rather than at word level. That is, bämäkina is a phrase, a noun phrase.

In the Amharic language, there are quite a large number of words, like bä-mäkina, that must be treated at phrase level rather than at word level. In tagging such words, two alternatives can be considered, at least for the purpose of this work, to resolve such problems. The first alternative is to treat words like bä-mäkina as they are and assign them a special tag that indicates that they are phrases rather than words. The other alternative is to use a pre-processor. This pre-processor separates the preposition bä and the noun mäkina, and inputs these as equivalents of bä-mäkina to the tagger so that the tagger tags the word bä-mäkina separating it as bä\PREP mäkina\N. The first alternative is adopted here in this study.

In addition to what is discussed above, other problems may also arise while tagging prepositions. As presented in chapter three, prepositions may appear

- as separate words in a sentence as in

  sə́la təmhərt t'ərat yəwayayalu "They are discussing about the quality of education."

  ə́ss^{w}a **wädä** bet hed-äč "She had gone home."

- prefixed with other words in a sentence as in

  lə́ju **kä-**bet mät't'a "He just came from home."

  lə́ju **bä-**mäkïna head "The boy went by car."

  lä-hə́zbu yətagälal "He is fighting for the people."

- As special prepositions, which are **-l-** and **–b-** as in

wändmmïwan gädälu-**b**-ät "They killed her brother."

zärafïwočun gädälu-**l**-ačäw "They killed the plunders for them."

- Consisting of two parts as shown below

kä-abbatwa **gar** mät'ť-ač "She came with her father."

kä-wägäb **bä-tač** "Dawn below waist."

bä-satn **wɔ**st' "In a box."

Prepositions that appear, as separate words present no difficulty during the tagging process. Such words are prepositions and are categorized under the POS called prepositions, and are tagged the tag **PREP,** as in the following.

sɔla\PREP tɔmhɔrt t'ɔrat yɔwäyayalu "They are discussed about the quality of education."

lɔju **wädä**\ PREP tɔmhɔrt bet hedä "The boy went to School."

If the prepositions appear prefixed with (i.e. not separated from the) other words (nouns) as **bä-**mäkina, the whole word is assigned a special tag **NP,** which is implemented as discussed under special tags for nouns. If the preposition is not separated from the verb, the tag VP will be implemented as discussed under special tags for verbs.

In this study, the prepositions **-b-** and **-l-** are treated as part of the whole word, which is always a verb and thus assigned the tag V, as in

zärafi-wočč-u-n gäddälu-l-äčw\V "They killed the plunderers."

Tagging prepositions that consist two parts, the last case, is not straightforward. In this study the following alternatives were considered to handle such cases.

- Treat everything, the prepositional prefix, the noun, and the postposition put after the noun, as one word and assign a single tag at the end of the preposition, as in for instance,

k^{w}asu **kä**-t'äräppezaw **h^{w}ala**\?? näw "The ball is behind the table."

from(preposition) table(noun) behind(postposition)

h∂mämu **kä**-wägäb **bä-tač**|?? näw "The pain is down the waist"

Where the two question marks, i.e.??, represent an appropriate tag to be assigned.

- Tag the preposition together with the noun to which it is attached with the tag for a noun headed by a preposition (i.e. NP), and the postposition put after the noun separately with any appropriate tag. Here are examples

k^{w}su **kä**-t'äräppezaw\NP **h^{w}ala**\PRP näw "The ball is behind the table"

h∂mämu **kä**-wägäb\NP **bä-tač**\NP näw "The pain is down from the waist."

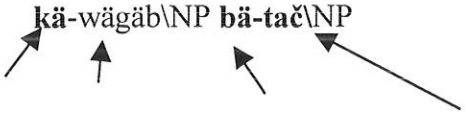
The first alternative is forwarded based on what the early scholars thought of prepositions that consist two parts (mersehazen, 1934). That is, according to the early scholars, prepositions consisting of two parts are treated as one preposition rather than separate parts consisting of a word composed of preposition not separated from the noun, verb or adjective and another word which is a preposition only or a noun headed by preposition (i.e. a word composed of a preposition not separated from the noun). Accordingly, for instance, **kä-wänd∂mm-u gär** "Together with his brother" should not be viewed as composed of a word consisting of two words, **kä-wänd∂mm-u** and **gär**. Rather the whole **kä-wänd∂mm-u gär** is viewed as a preposition.

On the other hand, scholars like Baye (1987)[1] and Gethahun (.1990)[1] treat prepositions consisting of two parts separately as a part consisting of a word composed of a noun headed by a preposition and a post position which is another word formed from a preposition only or a noun headed by a preposition. This second alternative is adopted in this study to solve the problem in tagging prepositions that consist two parts. The following examples illustrate this idea.

kä-wänd∂mm-u\NP gär\PREP "Together with his brother"

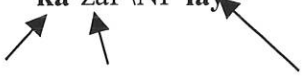
with (preposition) brother(noun)-his together(preposition)

kä-wägäb\NP **bä-tač**\NP "Down below the waist."



from(preposition) wasit(noun) twoards(preposition) below(noun)

kä-zaf \NP **lay** "Up on the tree."



from(preosition) tree(noun) on(preposition)

kä-angät\N^{P} **bä-lay** "Above the neck."

From (preposition) neck(noun) towards(preposition) above(noun)

Note that, in some works (Dawikins, 1969) **bä-tač, bä-lay** and so on are treated as adverbial nouns. Such view is against what is proposed by Baye, where such words are not adverbs but noun phrases composed of prepositions not separated from nouns.

[1] years are according to the Ethiopian calender

4.8 Tags for Adverbs

As discussed in chapter three, Amharic adverbs can appear as

- distinct primitive or compound adverbs appearing as one word (e.g. gäna "Yet", tolo "in a hurry.", ∂ndägäna "Again", s∂läzzih "Therefore", zare "Today.")
- As compound adverbs which are multi words (e.g. t∂kit t∂kit "A small amount", t∂nnš t∂nnš "Small")
- As compounds of preposition and other words appearing as one whole word, called noun adverbs (e.g. wädätač= wädä(PREP) "to" + tač(noun) "down").
- As short adverbial clauses.
- As gerunds used as adverbs and so on

In the first two cases, tag assignment is straightforward: all adverbs are assigned the tag ADV, with compound adverbs that appear as multi words assigned the tag ADV only once at the end of the compound adverb. Here are some examples.

zare\ADV gänna näw "To day is charismas"

l∂ju **t∂nnant∂na**\ADV kähägäru mäťťä "He came yesterday from his country."

ťornätu **bähulum s∂fra**\ADV ∂yatäsfafa näw "The battle went allover the place."

andand gize\ADV yämäbrat mäq^{w}arät y∂käsätal "Some times the light went off."

fätänawun **∂ndägäna**\ADV ∂ndisära täzäzä "They ordered him to take a re-exam."

The treatment discussed under prepositions consisting of two parts is applied in the third case. This is for the purpose of consistency with earlier discussion on nouns, verbs and adjectives headed by prepositions. Thus, instead of treating such words as adverbs, they are treated as nouns headed by prepositions and they are assigned the tag NP.The following illustrates this.

käťornätu bähuwala\NP wädä l∂mat täsämaru "After the battle is over they are engaged in development work"

Which should not be done as

käťornätu bäh^{w}ala\ADV wädä l∂mat täsämaru "They engaged in production after the battle went off."

Short adverbial clauses (prep + noun or prep + Adjective) can either be assigned the tag ADV or the tag for a noun or adjective headed by preposition, i.e. NP or JP. In this thesis, short adverbial clauses are assigned the tag NP or JP depending on whether the preposition is attached with a noun or an adjective. Examples on these are already given under special tags for nouns and adjectives. The following are additional examples but using short adverbial clauses.

bä-dästa\JP rädähu "I helped with joy."

Preposition adjective

bä-dähna\JP gäbu "They arrived in safe."

Preposition Adjective

bä-hay∂l\NP y∂rot'alu "They are running fast."

preposition noun

Adverbs suffixed with conjunction are assigned the tag ADVC, as in

gäbärew ahun-m\ADVC zim alä "Even now, the farmer does not give any response."

Days of the week will be assigned the tag ADV if they appear as an adverb in a corpus, as in

∂hud ∂hud\ADV s∂ra yäläwm "His day off is every Sunday."

If adverbs derived from gerunds[21] appear in a corpus, they are assigned the tag ADV, as in

[21] see(Dawkins,1969)

kähulu **qadmo**\ADV mäťťa "He arrived first."

läzare amät **dägmo**\ADV yagänaňän "May God bless us to next year."

Similarly as with such tags as N, V, J etc., in this work, the use of ADV is applied more widely, and as such it does not differentiate among such adverbs as adverbs of time, place, manner and so on.

4.9 Problems With Multi Words

In Amharic language nouns, adjectives, adverbs, verbs and conjunctions can appear as multi words. Here are some examples.

bunna bet	"Bar" , a noun
nägar gɔ̀n	"However " , a conjunction
tɔ̀llɔ̀k tɔ̀llɔ̀k	" Massive kind", an adverb or an adjective
kɔ̀ffɔ̀t allä	"It opens wide" , verb

Because multi words of the sort typified in the above examples function as one word, only one tag should be assigned to such multi words. Such cases for nouns, adjectives and verbs are already considered when dealing with compound nouns, adjectives and verbs. But, there is a need to generalize such special cases. For instance, the multi word **bunna bet** "bar" in the sentence Kassa tɔ̀llɔ̀k **bunna bet**\N aläw "Kassa has a high standard bar." should not be tagged as Kass tɔ̀llɔ̀k **bunna**\N **bet**\N aläw. Similarly, **nägär gɔ̀n** in abbatu tämäw näbbär **nägär gɔ̀n**\ADV ahun dɔ̀näwal should not be tagged as abbatu tämäw näbbär **nägär**\ADV **gɔ̀n**\ADV ahun dɔ̀näwal.

Unless there are especial cases to consider, in this study, multi words in any part of speech are treated as explained above. That is, the space between the multi words are retained and the tags are attached only once at the end of the multi words.

4.10 Tags for Conjunctions

Among the problems that will be encountered while tagging conjunctions are the following.

- Most of the conjunctions are also prepositions. That is, the same word can be used as both conjunction and preposition (see conjunctions and prepositions under chapter three).

- Conjunctions may be found attached with other words (e.g. nouns, verbs, adjectives and adverbs). Here are some examples.

wändɔmm-**na**\NC ɔhɔt načäw "They are brother and sister"

noun conjunction

Zare andärsɔm mängädu ruk naw-**na**\VC "It is far for us to reach to day."

Verb conjunction

gäbärew **ahun-m**\ADVC zim alä "The farmer still rested not to give a response."

Adverb Conjunction

In such cases, the tag NC, VC or ADVC are assigned as discussed per the discussion under verb tagging and special tags for nouns, and adverbs.

- Conjunctions may appear distinctively as separate single words. Such conjunctions pose no problem and are assigned the tag C, as in the following.

šay yəfälgalu **wäyəss**\C bunna? "Would you like tea or coffe."

abbatu alu ənatu **gən**\C motäwal "His father is alive but his mother has passed away."

➢ Conjunctions may appear distinctively as multi words. Such cases may be treated as per the discussion for multi words. That is, the multi word is tagged the tag C once at the end of the multi word rather than tagging the components separately, as in the following.

baləwa tämo näbär **nägär gən**\C ahun dənoal "Her husband was ill, but he has now recovered" which should not be tagged as

baləwa tämo näbär **nägär**\C **gən**\C ahun dənoal "Her has band was sick but he recovered now."

➢ Conjunctions may appear as compound conjunctions which consist of two parts, as for instance, in

bä-mäťťu\VP **gəze**\ bärun kəffät "at the moment when they came, open(m) the door" **(bä... gəze)**

abbatačäw **kä**-motu\VP **ba-h^{w}ala\NP** betun šätu "After their father passed away, They sold the house" **(kä... ba-h^{w}ala)**

bä-mäťťa\VP **qut'ər\N** betun yəräbšal "Whenever he comes he disturbs the family." **(bä ...qut'ər)**

In such cases, the solution suggested for prepositions consisting of two parts may be applied, as shown in the examples above.

4.11 Tags for Numerals

Numerals are mostly used **in the sense of an adjective or a noun**. If the numerals are used in the sense of adjective, they are assigned the tag JNU. If they are used as a noun indicating the English dozen, gross, one, two, three, four and so on they are assigned the tag CRD, as in

bät'ornätu **arat**\JSU säw motä"Four men were killed in the batlle."

The tag ORD is assigned to the Amharic ordinals such as **andäňňa** "first", **hulätäňňa** "second" and so on, as in

lɔju bäfätäna **andäňňa**\ORD honä "He stands first by the exam."

Compound numbers such as hulätt mätto sälasa ammɔsɔt "two hundred thirty five" or assɔrä ammɔstäňňa "fifteenth" may require special treatment. Two alternatives where considered in this study to handle such cases. The first alternative is to treat such numerals as multi words and hence assign only one tag appropriate form JNU, ORD or CRD, at the end of the multi word. The second alternative is to assign each component of the numeral an appropriate tag. In the following examples, (a) refers to the first and (b) the second.

1 a. hulätt mätto sälasa ammɔsɔt\JNU säw allä "There are two hundred and thirty five people."

b. hulätt\CRD mätto\CRD sälasa\CRD ammsɔt\CRD säw alä "There are two-hundred and thirty-five people."

2 a. haya andäňa\ORD honä "He stands twenty first."

b. haya\ORD andäňa\ORD honä "He stands twenty first "

From the two alternatives, the first (option) is used to tag such compound numerals.

Prepositions may also appear being not separated from numerals as in the examples below.

lä-hulätt qän sɔra fätɔto näbbär "He was idle for two days."

kä-hulätt sä'at bäh^{w}ala head "He had gone after two hours."

In such cases, since the words function as an adjective, they are assigned the same tag as numerals that function as an adjective, as in the following.

Examples **lä-hulätt**\JNU qän sɔ̀ra fätɔ̀to näbär "He hadn't do any thing for two days."

Kä-hulätt\JNU sä'at bäh^{w}ala head "He had gone to after two hours."

The case of distributive numerals, as in below may require special treatment.

hulätt hulätt bɔ̀rtukan sät'äčäw "Share them two oranges for each."

In such cases, distributive numerals are assigned the tag CRD once at the end of the multi word for the distributive numerals are more of nouns as they indicate the English one, two, three and so on as in

hulätt hulätt\N bɔ̀rtukan sät'äčäw "Do give each of them two oranges."

4.12 Tags for Interjections

According to Baye, interjections are not treated as words that are at word or phrase level, and as such they do not fall in any of the Amharic POS categories.

Options were also considered to handle this situation. The first alternative is to assign interjections with the tag for **unrecognized words**[22], which are denoted by UNC. The second alternative is to assign them a special tag for them denoted by ITJ. The later option is preferred for the purpose of this study. The following is an example of tagging interjections.

Goš\ITJ **!** tolo na "Please come soon."

[22] are words that are not in the lexicon of the tagger

4.13 Tags for Punctuation

In this study, the following characters are considered as a **delimiter** to identify individual words in a corpus written in Amharic: the white space, the colon (:), the square formed by four dotes (::), the question mark (?), the exclamation mark (!), tab, carriage return, line feed characters and such orthographic signs as ፤, ፣, ÷ and so on.

Besides serving as delimiter for words, the punctuation mark:: (the four dotes or double colon) marks also the end of an Amharic sentence. With no exceptions, all these punctuation marks or characters are assigned the tag PUNC, although it is possible to assign each a unique tag. The following are examples on tagging punctuation markes.

nägä sɔ̀ra alä wäyɔ̀s yälam?\PUNC "Is there a job tomorrow or not?"

goš !\PUNC tolo na "Please come soon."

läjäbäna ፤\PUNC lässɔ̀ni ፤\PUNC läbɔ̀rrɛ̀ɔ̀qqo ÷\PUNC läsk^{w}arna ÷ \PUNC läbunna÷\PUNC assɔ̀r ÷\ PUNC bɔ̀rr ÷\PUNC käfälu::\PUNC "They pay ten birr for kettle, cup, glass, sugar, coffee."

4.14 The Tag Set

The tag sets discussed in the preceding section are summarized in Table 4.4 below.

Table 4.4: A tentative POS tag set for the Amharic language

No.	TAG	DESCRIPTION
1	N	Nouns including all pronouns, invariant for number, gender and case except for verbal nouns and such nouns formed using the prefix balä(e.g. lɔ̀j "chiled", lɔ̀j-očč " Children", ɔ̀ssu "He" ,dägɔ̀nät "kindness")
2	NV	Verbal nouns (e.g. mäblat "Eating", mäťäťäť " Drinking."
3	NB	Noun formed by prefixing the prefix balä to nouns (e.g. balä-bäg "The

		Sheep owner", balä-suq "Shop keeper", balä-lɔ̈bs "owner of the clothing")
4	NP	A word with a preposition not separated from a noun (e.g. bämäkina "By car", sɔ̈lähägär "a bout a country")
5	NC	A noun suffixed with a conjunction, i.e. a word with noun not separated from a conjunction (e.g. lomina bɔ̈rtukan "Lemon and orange", zäyts "how about oil")
6	V	Verb in any form except auxiliary verbs, compound verbs and other forms of the auxiliary and compound verbs (e.g. gäddälä "He killed", gäddlo "after he killed", gäddäläč "she killed")
7	AUX	Auxiliary verbs and all their other forms. This does not include compounds of allä, adärrägä, assäňä and all their other forms (e.g. alä "He, It present" ,näbbär "He, It was", näw "It is", näč "She is",načäw "They are")
8	VCO	Compound verbs, i.e. compounds of allä, adärrägä, assäňä, and all their other forms (e.g bɔ̈q allä "He appears", zɔ̈m assäňto "In a way of making them silent", bɔ̈dɔ̈g adärrägä "He take it up.")
9	VP	Any verb (main or auxiliary) headed by a preposition. The preposition not separated from the verb (e.g. sɔ̈lämäťťa "Since he came", kähedä "If he went")
10	VC	Any verb suffixed or prefixed (i.e. headed) by a conjunction. That is, a word with the conjunction not separated from the verb (e.g. mäťťana "He come and" , sɔ̈mäťťa "When he comes", sɔ̈zänb "when it rains", ɔ̈sktɔ̈ċärs "Until you finish")
11	J	An adjective which is preceded by neither prepositions nor conjunctions (e.g. däg "Kind", kɔ̈fuňa "Dangerously", tɔ̈llɔ̈k "Big")
12	JC	An adjective not separated from a conjunction (e.g. dägɔ̈na yäwah "Kind and Innocent", ťɔ̈qurna näččɔ̈ "Black and white")
13	JNU	A numeral that function as an adjective (e.g hulät bɔ̈rrɔ̈ċɔ̈qqo " two glasses", ammɔ̈st guräno "five Sheds")

14	JPN	A preposition not separated from a noun but that function as an adjective (e.g. yät'äla bɔ̀rrɔ̀ċɔ̀qqo "A glass for "tella" ", yäčayna sahɔ̀n "A china made plate")
15	JP	A word with a preposition not separated from the adjective. That is, the adjective is headed by a preposition (e.g. bädähɔ̀na "In a fine way")
16	PREP	A preposition that appear being not attached with other words (e.g. kä "From", lä "To", sɔ̀lä "For Sb/Sth.", ɔ̀ndä "Like")
17	ADV	An adverb (e.g. tolo "In a hurry", tɔ̀lantɔ̀nna "Yesterday", zare "Today", hulgɔ̀ze "Always")
18	ADVC	An adverb which has a conjunction suffixed to it (e.g. ahunɔ̀m "Even now")
19	C	Coordinating conjunctions that appear being not attached with other words (e.g. nägär gɔ̀n "However", wäyɔ̀ss "Or")
20	REL	A word which is a relative clause (e.g. yätäsäräqäbät "one who is stolen", yäqomut "those that stand")
21	ITJ	Interjections (e.g. goš! "Wonderful", wa! "Take care! Be careful! Watch out!")
22	ORD	Ordinal number (e.g. ammɔ̀stäňa "The fifth" ,assɔ̀räňa "Tenth")
23	CRD	Cardinal number (e.g. ammɔ̀st "Five", assɔ̀r "Ten")
24	PUNC	Punctuation (e.g. ÷, :, !,?)
25	UNC	Unrecognized word, i.e. a word not found in the lexicon of the tagger

The total number of word class tags in this study is 24, plus one tag for all punctuations.

4.15 Conclusion

This chapter (also the next chapter) is central to this study and has discussed the tags drawn for Amharic language based on the analyses made in chapter three. These tags are not exhaustive as many more tags can be drawn (e.g. tags for abbreviations and contracted forms). But these tags were enough for they included the main categories that should be addressed.

These tags are useful for knowledge acquisitions, i.e. to represent the structural or linguistic information contained in the NL considered, and they are useful to generate the statistics required on lexical and transitional probabilities discussed in chapter two and five.

The next chapter discusses issues on POS algorithms, interface design, creation of database (the knowledge base), and the actual experiment conducted.

CHAPTER FIVE

Part of Speech Algorithm and the Experiment

5.1 Introduction

The previous Chapter presented the parts of speech tags designed to serve as a tool for the preparation of the lexical and transitional probability matrices in this chapter.

This chapter discusses the actual and major tasks involved in the process of automated POS tagging. Among the tasks discussed in this chapter are preparation of the knowledge base, design of the database and the interface and the development of the prototype.

Discussions in some detail are also made on the POS tagging algorithms, the experiments conducted and the results achieved on the small sample corpus used for the experiment.

The chapter begins with a discussion on the sample text used for the experiment and the manual tagging process.

5.2 The Sample Corpus and The Manual Tagging Process

For the purpose of the experiment conducted, a small sample of a narrative text in Amharic was obtained from modern Amharic grammar book by Getahun (1990). This sample was selected since it was prepared with the aim of testing students' knowledge of Amharic grammar and, thus, it is comprehensive and has also a direct relevance to this study. This

original text with its phonetic transcription (Leslau,1973)[23] are found in the appendix by the name **AmharicText** and **LatAmharicText,** respectively. The study was limited to such small sample for no text annotated with parts of speech was available and it was also laborious, expensive and time consuming to manually tag a large corpus. In addition, preparation of the **LexicoProb** and **TransProb** for a large corpus requires expense and time apart from the fact that it is arduous. The corpus **LatAmharicText**, with the inclusion of few statements and words, was then manually annotated with parts of speech tags by three independent annotators.

As there were no standards to measure what the three annotators annotate, a majority vote of the three annotators was taken as a base for the experiment made in this research. Where there were some problems, they were resolved by consensus. The final manually tagged corpus is found in the appendix by the name **ManuTaggedText**. It is this manually tagged text that was used to get the training and test set used for the experiment.

5.3 Preparing the Sample Data for the Experiment

This corpus, **ManTaggedText**, was then divided into two sets, the **training set** and the **testing set**. The training set was used to estimate the lexical and transitional probabilities required for the knowledge base. This set, the training set, consists of 90 percent (261 words, excluding punctuation marks) of the total data, **ManuTaggedText,** to be tagged.

The test set used in the experiment consists of 10 percent of the total data to be tagged. Since the performance of the algorithm was evaluated using cross validation techniques, the test set and the training set for this study were obtained as follows

[23] Minor Changes are made in this transcription for convenience.

The test set was obtained by repeatedly removing different parts of the corpus, **ManuTaggedText,** until it constitutes a test set of 10 percent of the data to be tagged. The remaining corpus constitutes the training set. A Random sampling was used to obtain the **Test Set** from **ManuTaggedText** simply by assigning numbers to each sentence and then drawing a lot.

The training and test set obtained in this manner are found in the appendix by the name **TrainigSet** and **TestSet,** respectively. As a preparation for the experiment, the punctuation marks **::,?** and ፧ that appear in **AmharicText** are coded by #, ^ and ~ respectively in the **ManuTaggedText** as well the **TrainigSet** and **TestSet.**

For the purpose of this research, only **the percentage of correct tag assignment** (although there are other measures like the **complexity of a text)** was used to measure the performance of the tagger.

5.4 Evaluation Procedures

The following were the procedures followed during the experiment to evaluate the performance of the tagger.

1. The tagger was first trained on the training set obtained earlier, **TrainingSet**. The Vitrbi and the sentence extraction algorithms (where necessary) were then run on the training set to see how well the tagger performs on the training set, **TrainingSet**.

2. The result obtained on the training set was then evaluated by comparing it with the manually tagged corpus used as a training set, **TrainingSet**. Aiming to improve the tagger accuracy, causes of error were identified and corrected and step one was repeated again. This processes of correcting errors and repeating step one was done until the result obtained was found to be satisfactory, i.e. tag assignments of the manually and automatically tagged corpora were more or less the same.

3. The tagger was then tested on the test set, **TestSet**, obtained earlier but with its untagged version. This test was essential for it helps to see the generality of the technique used. That is, it helps to see how well the algorithms and hence the tagger performs on the new or unseen data.

4. The output of the tagger in step 3 was then evaluated by comparing it with the manually tagged test set, **TestSet.**

5. Step 4 (and also step 2 and 3 if found necessary) was repeated by identifying and correcting errors until the test set tagged manually and automatically were fond to be almost the same.

5.5 Data Preparation and Component Building

This section discusses the actual data preparation processes carried out to create the knowledge base for the tagger and the components built to hold the knowledge.

5.5.1 The Lexicon

Based on the analysis of the **TrainingSet**, the lexicon designed in figure 5.1 was prepared. This lexicon serves as a base for preparing the matrix of lexical probabilities required in section 5.5.2.

Figure 5.1: Lexicon of the summary of word counts in the corpus

	N	NV	NP ...	V	AUX	JNU...	REL...	CRD...	TOTAL
and	0	0	0	0	0	3	0	1	4
guränno	3	0	0	0	0	0	0	0	3
bäg	7	0	0	0	0	0	0	0	7
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
šäťo	0	0	0	1	0	0	0	0	1
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
näbbäru	.	0	0...	0	1	0	0	0	1
TOTAL	80	3	14...	30	19	11...	4...	3....	261

The lexicon in figure 5.1 is part of the entry of the entire lexicon; named here on **Lexicon,** design ed for this study. From the **l**exicon above or the entire lexicon designed, we see that the horizontal total provides information on the use of words in the training corpus, **Training Set**. For instance, there were seven uses of the word **bäg** "sheep" and three uses of the word **and** "one". The vertical total provides information on the number of words in each part of speech category. Thus, categorizing the words into the different parts of speech, for instance, there are 80 nouns, 30 verbs, 19 auxiliary verbs and so on in the **TrainingSet.**

The **Lexicon** also shows that the corpus used for the training, **TrainingSet**, consists of 261 words, excluding punctuation marks. In this study only 23 (excluding the UNC and PUNC) word categories listed in Table 4.4 of chapter four were used in preparing the lexicon in figure 5.1. Dotes in the figure indicate the presence of words and categories that are not displayed but that are in the **Lexicon.**

5.5.2 The Lexical Probabilities

The lexicon presented in figure 5.1 was then used to approximate the lexical probabilities of each word for each of the 23 categories. The entire lexical probabilities thus obtained using the information in figure 5.1 is known by the name **LexicoProb** here on. A sample extracted from the sample **LexicoProb** attached in the appendix is presented in Figure 5.2 below.

Figure 5.2 : The Lexical probabilities matrix

p (and\JUN) = 0.636	P (yätäsäräqäbät\REL) = 0.025
p (and\CRD) = 0.333	P (šät' o\V)=0.033
p (guränno\N)=0.038	p (yägojam\NP)=0.071
P (bäg\N)=0.088	p (wädä\PREP)=0.444
P (leba\N)=0.013	p (bät' am\ADV)=0.026

The probability values in Figure 5.2 (and hence in the **Lexicoprob**) were computed using the information in Figure 5.1 and the formula

$$P(W_i \backslash C_i) = \frac{\text{number of times } W_i \text{ appears in category } C_i}{\text{total number of words with category } C_i}$$ as discussed in equation (18) of chapter 2

In other words, lexical probabilities in Figure 5.2 and hence in the **LexicoProb** were estimated simply by counting the number of occurrences of each word by category. For

instance, the probability of bäg "sheep" to be used as a noun in the **TrainingSet**, i.e. p(bäg\N), was computed as follows.

Since for **bäg** "sheep", W_i= **bäg** and C_i=**N**, the formula above becomes

p(bäg \N) = number of times **bäg** appears in category **N** .....(1)

total number of words with category **N**

From figure 5.1, the numerator is 3 while the denominator is 80. Substituting these values in (1) above will give 0.038 as an approximate value of p (bäg \N).

5.5.3 The Transitional Probabilities

The transitional probabilities to be used as a knowledge base for this study were obtained using the information in Figure 5.3.

Figure 5.3: Transitional (or bigram) probabilities generated from the corpus named TrainingSet

Category	Number of times the category appears in the text	Pair	Count of the number of times the pairs occurs in the corpus	Bigram	Bigram estimate
ϕ	23	ϕ, N	5	P (N\ϕ)	0.217
ϕ	23	ϕ, NC	1	P (NC\ϕ)	0.043
ϕ	23	ϕ, NP	3	P (NP\ϕ)	0.130
ϕ	23	ϕ, V	0	P (V\ϕ)	0
N	80	N, NP	2	P (NP\N)	0.025
N	80	N, N	6	P (N\N)	0.2
N	80	N, NC	2	P (NC\N)	0.25
N	80	N, NB	0	P (NB\N)	0
N	80	N, V	17	P (V\N)	0.213

N	80	N, VC	5	P (VC\N)	0.063
N	80	N, J	5	P (J\N)	0.063
NV	3	NV, N	0	P (N\NV)	0
V	30	V, N	4	P (N\V)	0.133
V	30	V, NV	1	P (NV\V)	0.033
V	30	V, V	2	P (V\V)	0.067
V	30	V, AUX	2	P (AUX\V)	0.067
V	30	V, C	2	P (C\V)	0.67
J	16	J, N	9	P (N\J)	0.563

Figure 5.3 gives only some bigram frequencies computed using the **TraingSet**, the sample corpus used as a training set. The values for the bigram in figure 5.3 are obtained using the formula

$$P(C_i = \alpha \backslash C_{i-1} = \beta) \cong \frac{\text{count of the number of times } \alpha \text{ and } \beta \text{ occur together}}{\text{number of times } \beta \text{ occurs in the corpus}}$$ as discussed in equation 16 of chapter two

Where (α and β are parts of speech).

For instance, the probability of a verb to follow a noun, i.e. p (V\N), was computed as follows.

Since for p (V\N), α= V and β=N, the formula above becomes

$$p(V \backslash N) \cong \frac{\text{count of the number of times } \mathbf{V} \text{ and } \mathbf{N} \text{ occur together in the corpus}}{\text{number of times } \mathbf{N} \text{ occurs in the corpus}} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

From Figure 5.3, it is clear that the numerator is 17 while the denominator is 80. Substituting these values in (2) gives 0.213 as an approximate vale for p(V\N). If the full information in figure 5.3 is prepared, it helps to get the transitional probability values required for the knowledge base. A sampleof the transitional probabilities is found by the name **TransProb** in the appendix. Please note in this table that, where the transition probabilities are 0.0001, the

actual probability values were zero. The value 0.0001 was used in place to account the problem of sparse data[24].

5.5.4 Database Design

To store the lexical and transitional probabilities in **LexcoProb** and **TransProb**, a database with four tables was constructed. Figure 5.4 shows the database schema designed for the Amharic part of speech tagger developed as a prototype.

Figure 5.4 Lexical and Transitional matrices database schema: primary fields are underlined

WCODE

Wcode	Word

CCODE

Ccode	Category

LEXICOPROB

Wcode	f1	f2	f3	.....	f21	f22	f23

TRANSPROB

Ccode	f1	F2	f3	.....	f21	f22	f23	f0

Note that the **LexicoProb** and **TransProb** actually prepared for the study are the last two tables respectively, but filled with data obtained from the Training Set, **TrainingSet**. In this

[24] The actual probabilities in such cases are zero for we used not a large corpus (Allen, 1995: 194)

figure **Wcode** is a code for each of the distinct words in the **TrainingSet** and **Ccode** is a code for each of the 23 tags in Table 4.4 of chapter 4, excluding UNC and PUNC, and including the pseudocategory. The table WCODE and CCODE field with data are founed in the appendix.

A database designated by the name **Sampledb** was then created to store the information in the four tables using Micro Soft Access. The structure of the four tables is shown in Figur 5.5 below.

Figure 5.5: Part of speech tagger table designs

Table name: TransProb			
Attribute	Data Type	Length/format	Description
Ccode	Integer	3	
f1	Number	Double, 3 decimal	
f2	Number	Double, 3 decimal	
f3	Number	Double, 3 decimal	
. . .	. . .	. . .	. . .
f21	Number	Double, 3 decimal	
f22	Number	Double, 3 decimal	
f0	Number	Double, 3 decimal	For the pseudocode

Table name: LexicoProb		
Attribute	Data Type	Length/format
Wcode	Integer	3
f1	Number	Double, 3 decimal
f2	Number	Double, 3 decimal
f3	Number	Double, 3 decimal
. .	. . .	. . .
f21	Number	Double,3 decimal
f22	Number	Double,3 decimal
f0	Number	Double,3 decimal

Table name: Wordcode		
Attribute	Data type	Length
Wcode	Integer	3
Word	Text	20

Table name: Category code		
Attribute	Data type	Length
Ccode	Integer	3
Category	Text	4

5.6 Part of Speech Algorithms

This section presents the Viterbi and other algorithms used to develop the Amharic part of speech tagger. Figure 5.1 is the Viterbi algorithm used in this study. A detailed discussion on how this algorithm works is found in (Allen, 1995). Below Figure 5.6 are discussions on codes written, see appendix, for the Viterbi and sentence extraction algorithm. A module is also discussed for the part that actually performs the POS tagging.

Figure 5.6 : The Viterbi algorithm, extracted from (Allen, 1995)

Given word sequence $w_1,\ldots w_T$, lexical categories $L_1,\ldots,L_N$, lexical probabilities PROB ($W_t \backslash L_i$), and bigram probabilities PROB (L_i , L_j), find the most likely sequence of lexical categories $C_1,\ldots,C_T$ for the word sequence·

Initialization Step

For i = 1 to N do

SEQSCORE (i, 1)= PROB ($W_1 \backslash L_i$) * PROB ($L_i \backslash \phi$)

BACKPTR (i, 1)=0

Iteration Step

For t =2 to T

For i =1 to N

SEQSCORE (i, t)=MAX$_{j=1,N}$ (SEQSCORE (j, t-l)*

PROB ($L_i \backslash L_j$))*PROB ($W_t \backslash L_i$)

BACKPTR (i, t)=index of j that give the max above

Sequence Identification Step

C (T) = i that maximizes SEQSCORE (i, T)

For i =T- 1 to 1 do

C (i)=BACKPTR(C (i+1), i +1)

5.6.1 The Sentence Extraction Algorithm

To extract the first sentence from a file

1. Initialize a variable, inputstring, to hold the whole content of the file.
2. Initialize a variable, SentenceCounter, to hold location.
3. Assign the whole content of the file to the variable initialized, i.e. inputstring, using the built in string manipulation function.
4. Starting from the first character in the inputstring, extract all characters up to and including the hash mark (#) from the inputString

5. Assign what you extract in step 4 to a variable, inputsentence, which holds only the sentences extracted. Not at this stage one sentence is extracted.
6. Update the SentenceCounter by 1
7. Assign the sentence extracted in step 5 to an input box, untagged sentence text box, to display the first sentence extracted
8. Disable start.
9. Enable tag, if you want to tag the sentence extracted and displayed in step 7

To extract the second sentence from the same file

1. Press the continue button
2. Empty the content of the untagged sentence text box in step 7, above.
3. Empty the tagged sentence text box if the sentence displayed in the untagged sentence text box is tagged and displayed in the tagged sentence box
4. Deactivate the tag button
5. Check if all sentence in the file are extracted
6. If not then
    - 6.1 Assign the variable inputsentence with all characters between the hash mark (#)

      and the sentence counter
    - 6.2 Update the sentence counter by 1
    - 6.3 Assign thesecond sentence stored in inputsentence to the untagged sentence text box, to display the second sentence extracted.
    - 6.4 Enable tag button
7. If yes, display a massage, file is exhausted

These steps, i.e. 1-7, will be done iteratevely until all sentences in the corpus are exhausted.

5.6.2 The Word Extraction Algorithm

When a start button is pressed, a function that uses the built in string manipulation is used to extract the first sentence from the input text. Then the following procedures will be executed to identify all words of the sentence

1. Initialize a variable to hold the individual word
2. Extract one character at a time
3. Check if the character is a word delimiter
4. If the character is a word delimiter, then put the word in the array declared to hold
   i. words of the sentence extracted
5. Else concatenate the character to the variable that hold the word
6. If there is more data to process, go to step 2

5.6.3 The Viterbi Algorithm

When the tag button is pressed, the viterbi algorithm is executed in three steps, initialization, iteration and sequence identification steps·

5.6.3.1 The Initialization Step

The initialization step will use the following steps for each word identified using the word extraction algorithm discussed above.

1. The next word identified will be searched in the **wcode** table
2. If the word is found, then its code will be put in a variable, wordcode
3. Then for each category the following steps will be done

    3.1 Checking will be made whether the word exists in the LexicoProb

    3.2 If the word is found in the category, then the probability of the word with that category code will be put in a variable, p1.

    3.3 Using the category code checking is then made from TransProb table

    3.4 Using the category code, checking is then made from TransProb table and the corresponding probability value will be put in another variable, p2

    3.5 Then the product of the two variables, p1 and p2 will be stored in its respective row of the array SEQSCORE and the first column

    3.6 The first column and the respective raw of the BACKPTR, an arry, will also be set to zero

    3.7 If there is another category in the array the process starts again from step 3.1 , else it will proceed to the iteration step

4 If the word is not found in the category table then the next word will be read and the procedure will start again from step 1

5.6.3.2 The Iteration Step

The iteration step follows the following procedures.

1. Initialization a variable, T, with value 2
2. Starting from the next word, i.e. next to the word identified in the initialization step, to the last word the following steps will be executed iteratively.
   - 2.1 The next word will be read from the word array
   - 2.2 The existence of the word in Wcode table will be checked
   - **2.3** If it exists in the Wcode table, then its code will be read into a variable, wordcode, and the following steps will be done **for each Category**
     - **2.3.1** Again for each category the following steps will be executed iteratively
       - **2.3.1.1** The category will be checked in Transprob table
       - **2.3.1.2** If it is found then the probability found in that column and raw will be assigned to a Variabel, pl
       - **2.3.1.3** If the category is the first category, then calculate max value and assign the location where the max value is obtained accordingly.
       - **2.3.1.4** Else calculate max value and compare with the max value computed in 2.3.1.3
       - **2.3.1.5** Exchange the value if it is less and also change the location to the current location where the max value is obtained
       - **2.3.1.6** If there is another category start again from 2.3.1.1
       - **2.3.1.7** Searching is made to check the existence of the word code in the LexicoProb table and assign the probability, value

that it finds from the respective row and column to a variable, p2

2.3.1.8 Assign the product of max value with the value of p2 obtained above to the corresponding row and column cell of the array SEQSCORE

2.3.1.9 Assign the location of the max value in the corresponding raw and column of the BACKPTR array

2.3.2 if there is another category continue from step 2.3

2.3.3 Increment the value of T initialized in step 1 above

2.3.4 If there is another word to process start again from step2.1 above

2.3.5 Assign to a variable, last index, the value one minus the number of words

5.6.3.3 The Sequence Identification Step

The sequence identification step follows the following steps

1. Assign the value in the first row and last index column to a variable called max
2. Assign location one to another variable named loc.
3. Starting from the next category to the last category it will execute the following steps iteratively

3.1 Assign the value in the raw of the current category column last index to a variable max1.

3.2 if the value of max1 is greater than the value of max compared in step 1 of this phase, above, then change the value of max with max1 and assign the index of the category to the variable loc.

3.3 if there is another category start again from step 3 of this phase

3.4 in another array called C it does the following procedures

3.4.1 assign the loc. value of lastword +1 found above at the place of last index in the array C

3.4.2 for each word string from lastword-1 up to the first, assign the corresponding location value from BACKPTR array to a corresponding place in array C

5.6.4 Tagging Procedures

Finally, the tag-text function will execute to produce words with their appropriate tags

1. Initialize a variable, i, that holds the current word category location in the array C to 1
2. Assign the next word from the words array to a variable
3. Check if the word is found in Wcode table
4. If the word is found then check the corresponding category location code in category code table to a variable category 1.
5. Concatenate the word and its category along with a separator ("\") into the text box (output box) that will hold the output file
6. Increment the value of i,
7. If the word is not found then
8. Chick if it is a punctuation mark. If so concatenate the word along with the code for punctuation (PUNC) separating them with "\"
9. Else concatenate the word along with the code for unrecognized (UNC) separating them by "\"

The source code for the Amharic part of speech tagger is found in the appendix by the name **SourceCode**.

5.7 The Interface Designed

A sample of the interface designed for the part of speech tagger application program is found in the appendix by the name **Interface**. When the Amharic part of speech tagger application program run, the **Amharic Parts of Speech Tagger** window will be displayed. This is in fact the main interface that will be displayed when the application program runs for the first time and it persists until the program is exited. This window has four buttons at the top, just below the menu bar. Below is a brief description of each of these four buttons

The **Tag From File** button

Clicking this button displays the **Tag From File** dialog box. This dialog box allows a user to tag a text written and saved in a file.

1. The **Drive**, **Director** and **File** boxes in the dialog box allow the user to specify the path and file name of the file to be tagged.
2. Pressing the **Tag** button tags each word in the specified file starting from the beginning of file (BOF) to the end of file (EOF) and displays the tagged text in the output box, labeled Tagged Sentence.
3. Pressing the **Save** button allows to save the tagged text in a file by the name and path specified.
4. The **progress bar** indicate the progress of tagging while the tagger tags sentences in the file specified
5. The **Back** button returns the user to the main interface

The **Tag Sentence-by-Sentence** button

Clicking this button displays the **Tag from File Sentence by Sentence** dialog box. This dialog box allows the user to tag sentences in a specified file starting from the BOF to the EOF, but displaying one sentence at a time.

1. The **Drive, Directory** and **File List** boxes works as described earlier
2. Clicking the **Start** button, activates the continue button and display the first sentence extracted from the file in the **Untagged Sentence** Box (USB).
3. Pressing Tag, tags the sentence in the USB and display the tagged sentence in the Tagged Sentence box (TSB) and also writes the tagged sentence by opening a file
4. Pressing continue, clears the content of the USB and TSB and then displays the next sentence extracted from the file.
5. Clicking the **Tag** button again does what is described in step 3 above but this time appending the tagged sentence in the file opened in step 3.
6. Step 3 and 4 will be repeated until all sentence in the file are exhausted.
7. Pressing the **Save** button allows the content of the file opened in step 3 to be saved by the name and the path specified.
8. The **Back** button returns to the main screen from anywhere.
9. The **Progress Bar** functions as described earlier

The tag **Typed Sentence** Button

Clicking this button displays the **Tag Typed Sentence** dialog box, which allows to type a sentence directly into the input box and tag it. The buttons in this dialog box works as follows.

1. The input Box labeled **Sentence to Tag** is where the user types the sentence (or word) to be tagged.
2. The output Box labeled Tagged Sentence is where the **Tagged Sentence** (or word) is displayed.
3. Pressing the **Tag** button tags the sentence (or word) in the input box and displays the tagged sentence (or word) in the output box
4. The **Refresh** button, clears both the input and output boxes.
5. The **Back** button and the **progress bar** function as explained before.

The **Exit** button

Clicking this button closes and exists the **Part of Speech Tagging** application program

5.8 The Experiment

As outlined in the evaluation procedure the experiment for this study was carried out in two phases, experiment on the **training set** and **test set**.

5.8.1 Experiment On The Training Set

In this phase the tagger was first trained on the whole corpus, i.e. **TraingSet,** and was then run on the same data, i.e. **TraingSet.** Assuming that errors obtained in this phase are more of human made errors (errors in manually tagging the **ManuTaggedText** and preparing **LexicoProb** and **TransProb**), the **ManuTaggedText**, **LexicoProb** and **TransProb** were reviewed making corrections where necessary. The tagger was then retrained and the test redone again on the **TraingSet**. The final result obtained before and after such corrections were made were then reported under results of the experiment, section 5.9.

5.8.2 Experiment on The Test Set

In this phase the model of the language in 5.8.1 above was used to test the remaining 10% of the corpus left for the testing purpose, i.e. **TestSet**. The final result achieved on testing the tagger on the unseen part was then reported under results of the experiment.

5.9 Result of the Experiment

5.9.1 Result on the Training Set

The result obtained when the tagger was trained and run on the same data, **TraingSet**, is shown in table 5.1.

Table 5.1 Tagging result before making no error correction

Data	No of words	No of errors	Accuracy
TraingSet	261	32	88%

As one would expect the accuracy achieved should be high when a tagger is trained and tested on the same data. But, due to human made errors the accuracy was not as high as it was expected

The final accuracy obtained after human made errors are identified and corrected is shown in Table 5.2 below.

Table 5.2 Final tagging result on the training set after human made errors are corrected.

Data	No of words	No of errors	Accuracy
TraingSet	261	7	97%

As can be seen from fig 5.2 the result achieved after correcting human made errors indicates a better, in fact a high, accuracy.

5.9.2 Result On The Test Set

The test on the unseen part of the training corpus provides the result shown in table 5.3.

Table 5.3: Tagging result on the test data, i.e. **TestSet**

Data	No of words	No of errors	Accuracy
TestSet	29	3	90%

The result obtained when testing the tagger on the test set is approximately 90%. This experiment indicates that the accuracy level achieved using the small amount of training set is rather accaptable. Thus, the tagger developed seemed acceptable with such accuracy assuming the **ManuTaggedText** as the whole of the world knowledge.

5.10 Discussion

5.10.1 Error Analysis

During the experiment the following causes of errors were identified

1. Human made errors during the manual tagging process were identified to be one cause for wrong tag assignments. The words **lämägɔzat** "to buy" and **bätam** "very" are examples of this case. These words were originally assigned the tag **V** and **J** by the annotators, respectively. Later they were identified as words that should be tagged **VP** and **ADV**, respectively.

2. Some times a word assigned a wrong tag might raise the error rate significantly. **lämägɔzat** "to buy", which is a verb with a preposition not separated from it, is an example of this case. This word and all the nine words after it up to the hash mark (#) were assigned the tag N by the tagger during the first experiment. This results in 9 words, including **lämägɔzat,** to be wrongly tagged. Removing this word and repeating the experiment results in all the nine words to be assigned with their correct tages.

3. Human made errors made while preparing and storing the lexical and transitional probabilities stored in the database are other causes ofwrong tag assignment. **lämägɔzat** and **nät** "she is" were two examples of this case. These words were wrongly tagged for their probability value, p **(lämägɔzat\category)**, and p **(nät\category),** for each of the 23 categories was wrongly recorded to zero. Correcting these errors, and replacing the word **lämägɔzat** in the **TrainingSet** and doing the experiment again results in all the nine words including lämägɔ**zat** and **näw** to be assigned with their correct or appropriate tags.

4. Spelling the same word differently in the database, **wcode** and **TrainingSet,** as for instance **bɔ̀rɛ̀ɔ̀qqo** and **bɔ̀rɛ̀ɔ̀ggo** to mean **"glass"** was another cause of error**.** This accounted for six errors encountered in the experiment. Such errors were easily corrected.

5.Thirteen errors during the experiment on the training set were accounted due to compound words that are multi words. These errors were corrected by joining such compound words using hyphen both in the TrainingSet and the database table, Wcod**e.**

6. Some errors persist throughout the experiments (e.g. and "one")

5.10.2 Dealing With Wrongly Marked Up Words

The following were some of the ways used in the experiment to deal with wrongly marked up words

1. Review of the manually tagged corpus for human made errors and correcting them
2. Review of the large statistics used as a database (or knowledge based), which otherwise seriously affect the performance and hence the accuracy of the tagger, and correcting errors.
3. Avoid or correct words that are assigned wrong tags and that raise the error rate significantly
4. Avoid spelling the same word differently in the **TrainingSet** and the table named **Wcode**
5. Preprocess multi word compound words both in the **TrainingSet** and the **Wcode**

5.11 Conclusion

In this chapter, detailed discussions were made on the data prepared for the knowledge base, the database designed and the algorithms used.

Based on the algorithms, a program was written in Visual Basic and a prototype was developed with an interface easy to be used by any individual.

An experiment was also conducted using the prototype and the small sample corpus selected for the experiment. The results achieved and the discussions made out of the experiment are also reported in this chapter.

From what has been discussed in this chapter, the prototype developed has high accuracy, 97 percent for the training set and approximately 90 percent for the test set.

The next Chapter closes this thesis work providing conclusions and recommendations.

CHAPTER SIX
CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

In this thesis, I described how to develop an Automatic part of speech tagger for Amharic language using the Viterbi algorithm, and developed a prototype.

The thesis begins with a brief discussion on the role NLP plays in enhancing computers capability to process NL. POS tagging is one approach to understand NL. Being a central issue in my attempt to develop the Amharic POS tgger, POS tagging is a task which assigns a unique syntactical category or part of speech tag to sentences that are presented as a linear string of words. In this study important concepts and terms in relation to POS tagging have been made clear from the outset. Attempt was made to illustrate application areas where the outputs of a POS tagger are useful. The different Approaches to automated part of speech tagging have also been described in some details. Rule based and stochastic approaches, which are the major approaches to POS tagging, were briefly reviewed and the relative advantages of each approach have been included in the discussion.

The stochastic HMM approach was adopted in this research to develop the Amharic tagger. Reasons for such preference and, the step-by-step procedures followed to prepare the lexical and transitional probability matrices are discussed. The probability concepts and the mathematics related to this approach were also discussed in some detail.

Literature in the area of Amharic word classes was reviewed and discussed, as the knowledge of the grammar of the language is useful to design the POS tags and tag set required for the tagger.

The thesis then dwells wholly in chapter four, the major and central part of the thesis, drawing the POS tags using the properties of the language studied and presented in chapter 3. Almost all tags derived for this study were applied more widely. That is, the part of speech tags do not indicate such features as gender, number, person, tense, case, polarity, definiteness and so on. The tags developed indicate only the POS category in which the words are categorized. In cases when it is not possible to categorize words in a particular POS category, special tags were introduced.

Some of the major problems that the researcher faced in the process of drawing the POS tags, which are linguistic labels, and the steps taken to deal with the problems were also presented.

Steps in preparing the actual transitional and lexical probability matrices and the computations made to get such probabilities were presented in full details. The lexical and transitional probabilities obtained were presented in two tables, which form the knowledge base of the tagger, **LexicoProb** and **TransProb**. A database with four tables was then designed to hold the information in **LexicoProb** and **TransProb**.

The thesis then presented the algorithms and modules required by the tagger to access the knowledge base and annotate incoming sentences with appropriate parts of speech. For this purpose an interface, which allows a user to tag texts in three different ways, was created and a prototype was developed using Visual Basic.

Experiments were conducted in two phases, one on the training set and the other on the test set, obtained from a small sample using the cross validation method. Evaluation of the tagger performance was made based on the evaluation procedures outlined in the thesis. In the study only one parameter, the percentage of correct tag assignment, was used to measure the performance of the tagger.

The results achieved based on the small sample were high, 97% on the training set and approximately 90% on the test set. Before achieving such accuracy, the experiment was repeatedly done on both the test set and the training set but identifying errors and making corrections. The Viterbi algorithm was used in this study with no modification. All errors identified were due to human made errors rather than the algorithms used. Finally, the experiments conducted, the results achieved and a discussion on the possible causes of errors were presented with their solutions before winding up the thesis.

Although the accuracy of the tagger developed in this study is somewhat acceptable it may not have an immediate practical application for the tagger was not trained on large quantities of data.

In this study, the tagger developed was not trained on such huge data for various reasons. The reasons for not being able to train the tagger on such data are:

1. Lack of large corpora in the language annotated with POS tags.
2. Manually tagging and generating such quantities of data is very laborious, expensive and time consuming.

3. Financial constraint to carry out such huge research
4. Scope of the thesis, which is limited to demonstrate the potential of stochastic HMM approach to develop an automatic POS tagger to Amharic language.

Apart from such limitations, the research has indicated the possibility to construct or develop an automatic part of speech tagger for Amharic language. That is, it appears to be feasible to develop an efficient tagger for the Amharic language provided that enough training data is available.

Lastly, it is hoped that this thesis has described a new approach to study the Amharic language, which in future answers the growing desire for annotated corpora by researches addressing different issues in Amharic language.

It is also hoped that this first work in POS tagging will encourage Ethiopian students and researchers to take part in POS tagging which ultimately lend to a higher level and more demanding research endeavors such as parsing and machine translations, which all are tasks of NLP in general and NLU in particular.

6.2 Recommendations

There are a number of exciting future research directions in which similar studies (to the present) could be pursued. These are presented below as recommendations.

1. Launching big project to develop an efficient automatic POS tagger for Amharic language, which in its kind will be similar to taggers developed for others languages, like English and French.
2. Organizing a NLP team, which will be responsible to carry out such huge project in particular, and researches in NLP in general. As NLP requires expertise from different fields of study, the team for such project should consist of at least linguists, programmers and information professionals.
3. Experts and researchers in the area of Amharic language should be encouraged to manually annotate large quantities of data in the language to help researchers who wish to address a number of linguistic phenomena and also train taggers and parsers.
4. The school of information studies (SISA) should think of also offering such courses as syntax and semantics, which are useful to understand NL. Offering such courses might create interest among students and encourage them to be involved in research related to NLP in general and NLU in particular.
5. The two-departments, **SISA** and the **Department of Linguistics,** should work in close collaboration, especially in conducting research in the area of Natural Language Processing.

6.3 Future Research Areas

This research has many shortcomings. These are gaps that future researchers should address to come up with an efficient POS tagger for Amharic language. Thus, the following are suggested as possible research areas.

1. Replicate this work using a large corpus and incorporating such attributes as case, number, gender, person, tense, definiteness and so on for the tags used in the different part of speech categories.

2. Conduct similar researches in other local languages by adopting the procedures followed in this study.

3. Develop taggers for Amharic and other local languages using other approaches (e.g. rule-based approach) and compare the results obtained using the stochastic approach, the approach followed in this thesis.
4. Syntactic parsing, semantic parsing and machine translations are other possible future research areas worth conducting as a continuation of a full ledged Amharic tagger
6. Replicate this work using **trigram transational probabilities** rather than bigram probabilities and compare the two for higher accuracy.
7. Conduct research on tagging Amharic texts and identify the problems associated with it which, I think, will provide an exhaustive tag set list for the language which will be useful for researchers that require annotated corpora.

Bibilography

Abiyot Bayou (2000). Developing Automatic Word Parser for Amharic Verbs and Their Derivation, Master Thesis at The School of Information Studies for Africa.

Allen, Jemes (1995) Natural Language Understanding, 2nd ed.; The Benjamin / Cummings Publishing Company, Inc.; California

Amsalu Aklilu (1979) Amharic- English Dictionary; Kuraz publishing Agency; Addis Ababa

Babiner, Lawrence; Biing- Hwang Juang (1993) Fundamentals of Speech Recognition, Prentice Hall PTR, New Jersey

Baye Yimam (1986) yäamarðň säwäsäw ; E.M.P.D.A, Addis Ababa

Bender, M.L. (1974) Amharic verb Types from Text and Lexicon; Folia Oriental 15, (23-46)

Bender, M.; Freguson, C. (1976). Introduction, Language in Ethiopia; edited by Bender, M. and Frguson, C.[et al]; Oxfor:London

Bender, M.L and Hailu Fulas (1978). Amharic Verb Morphology. Michigan State University.

Brants, Thorsten (1997) Internal and External Tagsets in Part-of- Speech Tagging. University at Dessaarlandes Computational Lingustics: Saarbrucken, Germany in proceeding of Eurospeech 97, hodes,Greece

Brants, Thorsten (2000) TnT –A Statistical part-of-Speech Tagger. In Processing of the six Applied Natural Language Processing Conference ANLP-2000, April 29-may 3,2000, scattle, WA.

Brill, Eric (1993) A Corpus-Based Approach to Language Learning, Presented to the Faculties of the University of Pennsylvania in partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy,

Brill, Eric (no year) Unsupervised Learning of Disambignation Rules for part of Speech Taggig.

Chanod Jean-pierre; Pasi Tapanainen (no year) Tagging French-Comparing a Statistical and Constraint-based Method. At:- http://www.xerox.fr/grenoble/mltt/home.html

Cotterell, F.B.(1964) " Amharic word classes" in Journal of Ethiopian Studies 4:2, (33-48)

Culicover, Peter W. (1976) Syntax. Academics Press, Inc, New York

Cutting, Doug; Julian Kupliec; Jan Pedeersen; Sibun Penelopoe (no year) A Practical Part-of Speech Tagger, Xerox Palo Alto Research Center, Palo Alto, CA, USA

Dawkins, C.H(1969) The Fundamentals of Amharic , A.A Sudan interior mission.

Dereje Teferi (1999) Optical Character Recognition of Type Written Amharic Text. Master Thesis at The School of Information Studies for Africa, Addis Ababa

Erjavec Tomaz; Ide Nancy; Dan Jufis (1998) Development and Assessment of Common Lexic Specifications for six Central and Eastern Euro Languages. At:- http://www.ach.org/abstracts/1998/ab510.htm

Eynde, Frank V; jakub Zavrel ; Walter Daelemans (no year) Part of Speech Tagging and Lemmatisation for Spoken Dutch copus

Feldman, Susan (1999) NLP Meets the Jabberwocky: Natural Language Processing in Information Retrieval. At:- http://WWW.onlineinc.com/onlinemag/OL1999/fieldmans.htm/

Getahun Amare (1989) Zämänäwi yäamarðñ Säwäsäw bäqäläl aqäräräb; commercial printing Press Addis Ababa.

Girmay B. (1992) Word formation in Amharic. Journal of Ethiopian Language and Literature. No.2, 50-74

Guilder, Linda V.(1995) Automated Part of Speech Tagging: A Brief Overview. At:-
Vanguill@gusun.georgetown.edu

Harris, Mary D. (1985) Introduction to Natural Language Processing; Reston: Virginia

Hornby, A.S (1995) Oxford Advnced Learners Dictionary; Oxford: London University Press, Oxford.

Hirut Woldemariam (1989) "An Auto segmental Approach To Geez-Based Amharic Plural Nouns" in Institute of Language Studies Addis Ababa University: Addis Ababa 69-71

Kim, Youngin; Barbara Norgard (1998) Adding Natural Language Processing Techinques to The Entery Vocabulary Model Bulding Process. At:-
http://www.sims.berkely.edu/reserch/metadata/nlptech.html

Leslau, Wolf (1965) An Amharic Text Book of Every day Usage University of California, LosAngles.

Leslau, Wolf (1967) Amharic Text Book, Otto Harrassowitz, Wiesbaden, Belgium

Leslau, Wolf (1973) English –Amharic context Dictionary otto Harrassowitz, wiesaden, Belgium.

Mersehazen woldeqirqos (1934) yäamarðň säwäsäw, Birhanena Selam Printing Press, Addis Ababa.

Mao, Yonghong (1997) Natural Language Processing Model (Part of Speech Tagging and Sentences Parsing) Laboratory Manual. At:-
http://www. csic.cornell.edu/201/natural language

____,_____ no year Corpus Annotation. At:-
http://www.comp.lancs.ac.Uk/ucrel/annotation-htm/

_____, _____BNC2 POS-tagging Manual, Guidelines to word class tagging. At:-
http://www.comp.lancs.ac.uk/ucrel/bnc2/bnc2guide.htm

Nega Alemayehu (1999) Development of Steming Algorithm for Amharic Texts Retrieval.PHD Thesis at University of Shifeld.

Olsson , Fredrik (1998) Tagging and Morphological processing in the Sevensk System . At:- http://stp.ling.uu.se/N fredriko/exjobb.htm/

Pereira, Fernando (no year) Sentence Modling and Parsing. At:- http://cslu.cse.ogi.edu/HLT survey/ch3nodes.html

Salton.G. (1983) Introduction to Modern information Retrieval, McGraw-hill Company, New York

Stocbcwell, Robert P. (1997) Foundations of Syntactic Theory.Prentile-Hall inc,. Englewood Cliffs, New Jersey

Tufis, Dan; Oliver mason (no year) Tagging Romanian texts: a case study for QTA, a Language

Wedekind (1997), which Form, Predict all Other Best? Varition on the Amharic Verb "Theme".Journal of Ethiopian Studies 25, pp. 73-92

Wojeski, William (1999) Natural Language Processing in Information Retrieval: A case for Knowledge Representation Through Conceptual Graphs. At:- http://www.medir.ohsu.edu/~wojeskiw/minf514plaper.htm

Worku Alemu (1997) The Application of OCR Techniques to Amharic Script. Master Thesis at The School of Information Studies for Africa.

Appendix

Due to the requrment imposed by the Postgraguate School to limit a Masters thesis to 150 pages, the appendices mentionedin this thesis could not be included here. The Appendix for this thesis is compiled in the form of a booklet and is available in the **Bibiliograpic Lab of the School of Informaion Studies for Africa [SISA].** Readers of this thesis are higly recommended to use this thesis together with the appendix. Only the autput of the Amharic tagger developed, which is a tagged text, is attached here together with the interface desigined for the tagger.

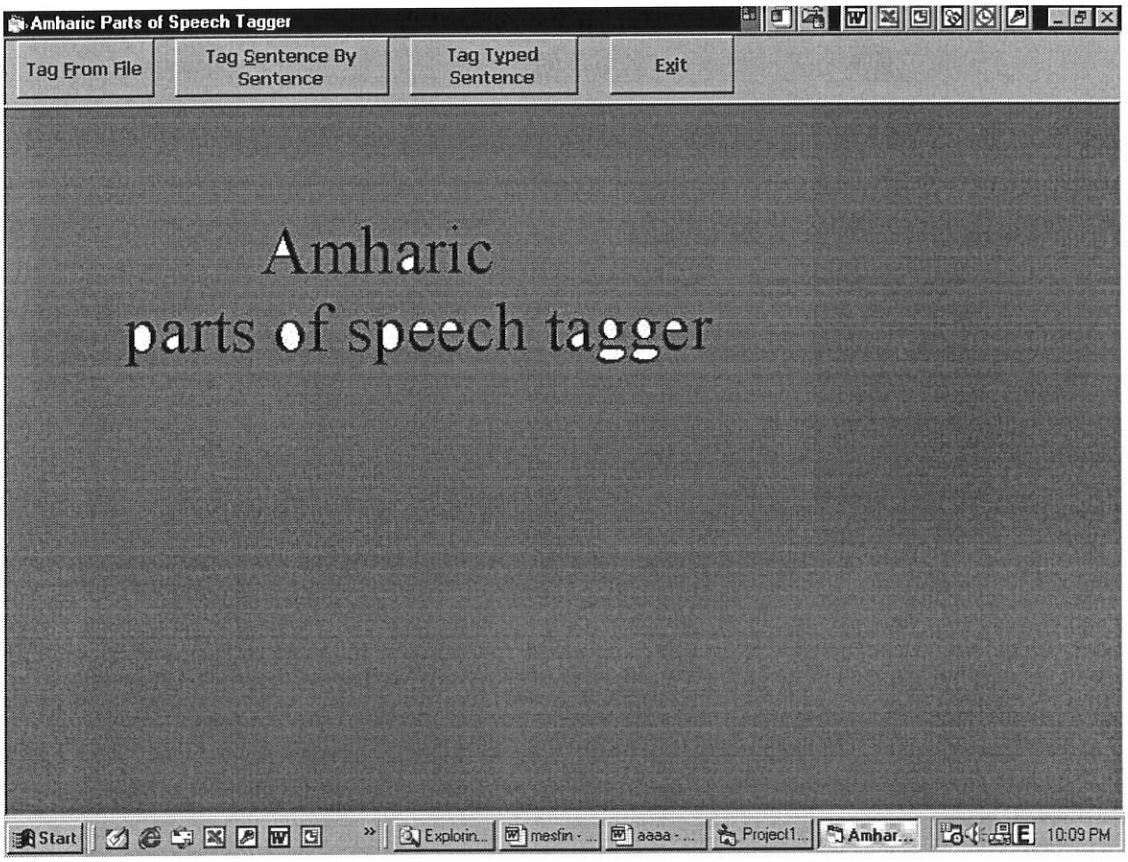
An Amahric text tagged using the Aharic POS tagger

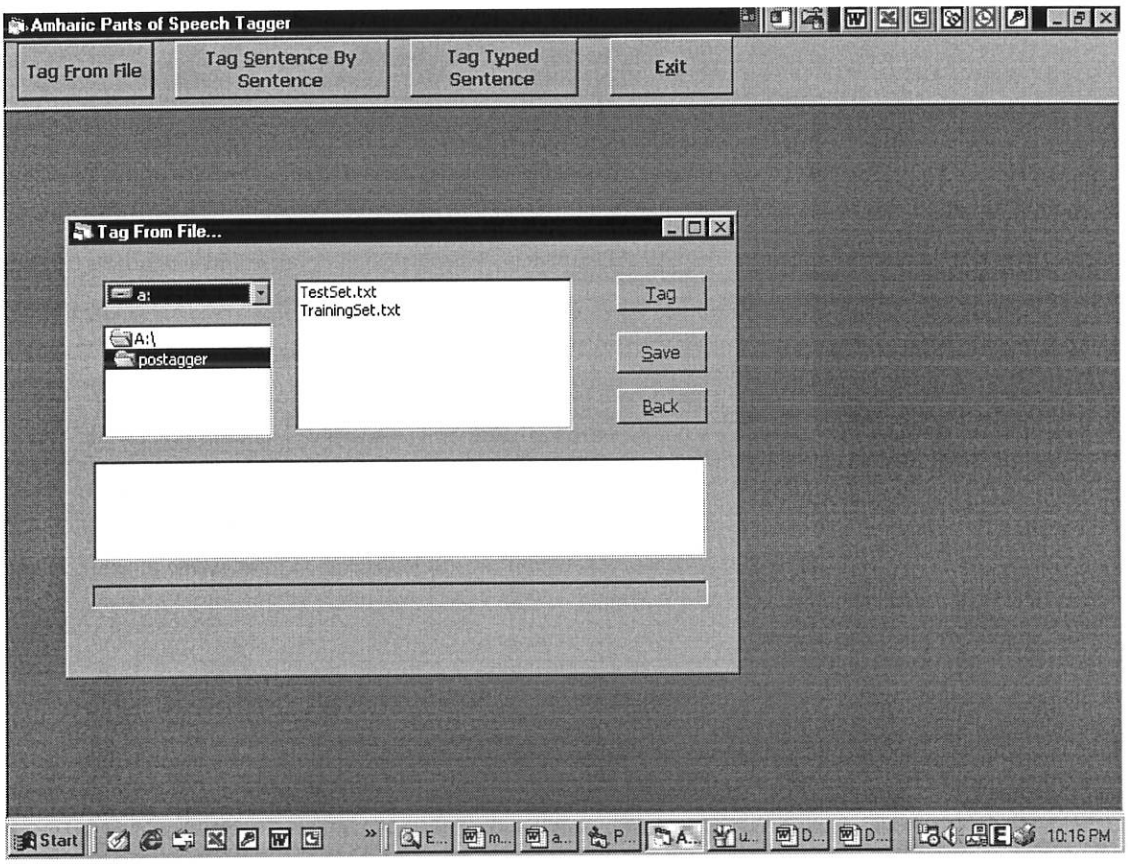
And\JNU guränno\N bäg\N leba\N yäsäräqäbät\REL and\JNU yägojam\NP gäbäre\N amm?st\JNU ?g?r\N gešowocun?nna\NC hulätt\JNU kuntal\N s?nde\N šäto\V batäraqqämäw\VP gänzäb\N wädä\PREP gäbäyä\N hedo\V amm?st\JNU yär?bbi\J bägocc?n\N lämägzat\VP fäll?go\V käsäfäru\NP bätam\ADV ?ruq\J wädähonä\VP t?ll?q\J gäbäyä\N guzowun\V jämmära\V #\PUNC kähulätt\NP qän\N guzo\N bähwala\NP ?gäbäyaw\NP bota\N därräsä\V #\PNUC ?gäbäyaw\NP bota\N yädäräsäw\VP bätam\ADV bätwat\ADV s?länäbbär\VP m?n?m\N säw\N alnäbäräm\AUX #\PUNC wäfram\J yät?t\JPN gabiwun\N täkänanbo\V bägäbäyaw\NP dar\N ballä\VP yäd?ngäy\JPN kab\N lay\PREP quc-alä\VCO #\PUNC käzziyam\C mistu\N wäy?zäro\N almaz\N käzägajäc?llät\VP and\JNU agälg?l\N yäs?nde\NP dabbo\N andun\CRD awätanna\VC gämäss-addrgo\VCO qur?sun\N mäblat\NV jämmära\V #\PUNC y?hun-?nji\C ayäru\N ?ndä\PREP t?q?mt\J wurc\N bätam\ADV qäzqaza\J s?länäbbär\VP dabbowun\N alämto\V mäwat\NV alcaläm\V #\PUNC

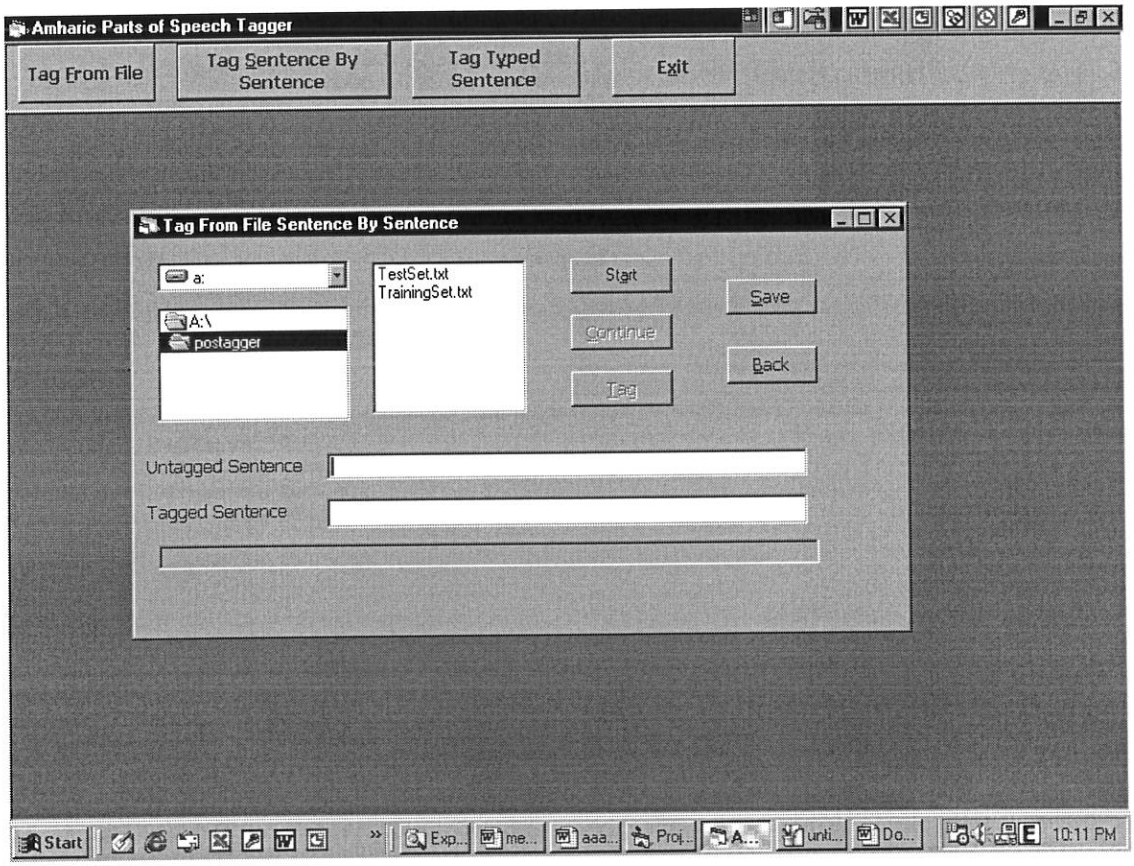
Lähulätt\NP säat\N yah?l\N täqämto\V käqoyyä\VP bähwala\NP t?qit\ADV säwwoc\N yätäläyayyu\J šäqätocc?n\N ?yyäyazu\VP käarattum\NP aq?tacawoc\N b?q-b?q\ADV malät\PREP jämmäru\V #\PUNC yämišätäw\REL aynät?m\N yann?nu\J yah?l\N ayyäle\J honä\AUX #\PUNC gäbärew\N hulätänna\ORD gize\N wädä\PREP bäg-tära\N sigwaz\VC fit-läfitu\ADV yägännaw?n\VP šäqät\N waga\N y?täyqal\V #\PUNC qän?s?nna\VC ?nnäzih?n\N yätälla\JPN b?rcqqowoc\N š?t?l?n\V #\PUNC ?yyalä\VP sikärakkär\VC ~\PUNC ?ne\N tänkarra\J yäcayna\JPN b?rcqqowoc\N allun?na\VC yänen\JPN b?rcqqowoc\N g?za\V ?yyalä\VP kägon\NP bäkkul\PREP yalläw\VP näggade\N y?tärawal\V #\PUNC bämäcrräša\ADV hulätt\JNU yäplastik\JPN kubbayawoc\N b?ca\ADV gäzto\V wädä\PREP bäg-tära\N head\V #\PUNC bäg-tära\N ?ndädärräsäm\VC yätäläyayya\J qäläm?nna\NC m?tan\N yällacäw\VP bätam\ADV b?zu\J bägoc\N ayyäna\VC b?zatacäw?n\N ?yyadänäqä\VP ass?r\JNU guränno\N y?honallu\AUX #\PUNC ay\ITJ haya\JNU guränno\N y?honallu\AUX ?yyalä\VP kärasu\NP gar\PREP sinägäggär\VC qo?yto\V wädä\PREP andit\JNU bäg\N täga-b?lo\VCO bämäyaz\NV bäläy?c\NB bäg\N ?\PUNC b?lo\AUX täyyäqä\V #PUNC bäläbetum\NB yäne\NP nat\AUX aläw\AUX #\PUNC s?nt\ADV näc\AUX ?\PUNC aläw\AUX gäbärew\N ato\N kassa\N #\PUNC and-mäto\JNU b?rr\N alä\AUX bäläbet\NB #PUNC gäbärew\N z?m-b?lo\VCO lihed\VC sil\AUX na\V y?ci\N t?qän?salläc\V b?lo\AUX andit\CRD t?qur\J bäg\N asäyyäw\V #\PUNC

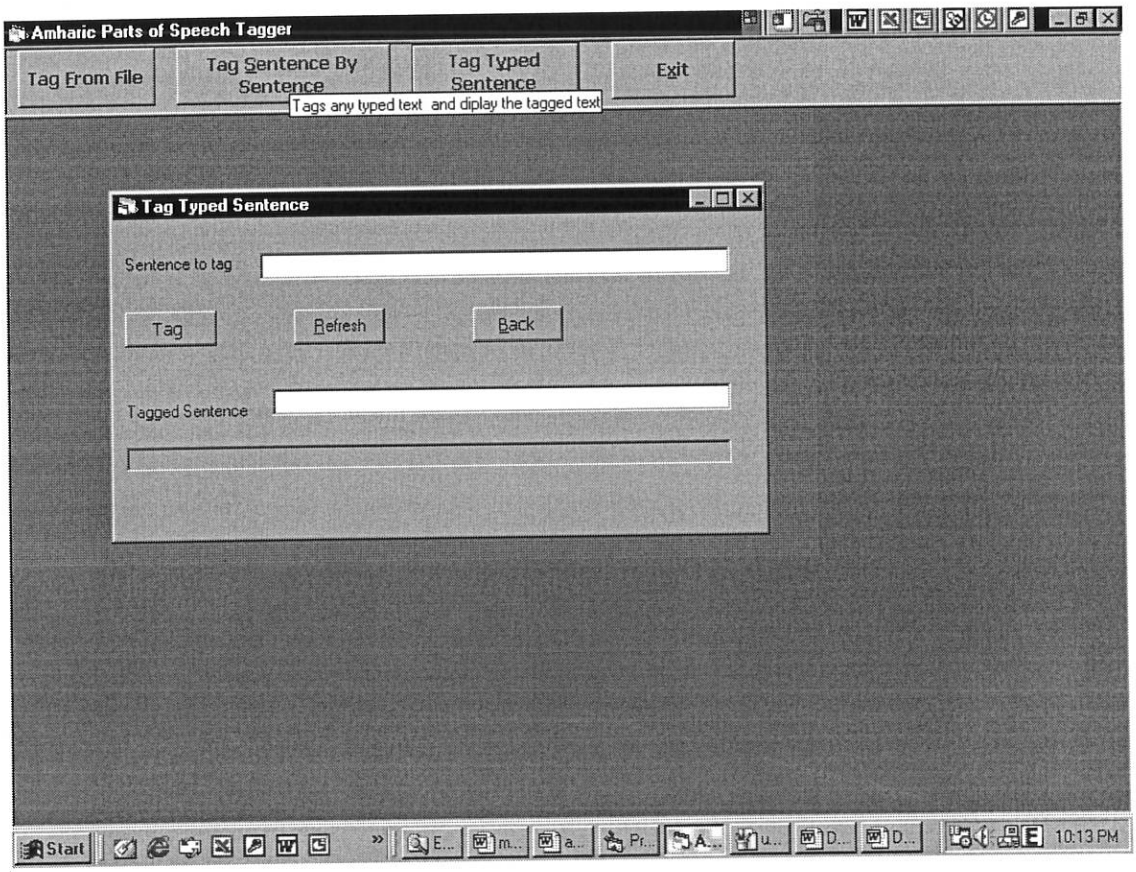
Gäbärew\N ahun?m\ADV z?m-alä\VCO #\PUNC yac?nawam\NC t?ru\J näc\AUX b?lo\AUX lela\ADV qayy\J bäg\N asäyyäw\V #\PUNC gäbärew\N ?ne\N yard\J bäg\N alfall?g?m\V b?lot\AUX wädämiqät?läw\VP alläfä\V #\PUNC käzziyam\C bätämäsasay\JP huneta\N y?h\N bäg\N s?nt\ADV näw\NX ?\PUNC ?näzzih\N bägoc\N s?nt\ADV näcäw\AUX ?\PUNC ?näzziya\N bägocs\NC ?\PUNC ?yyalä\VP sizor\VC qoyy?to\V and\CRD bota\N sidärs\VC däng?to\V qomä\V #\PUNC gud\ITJ !\PUNC alä\AUX #\PUNC läka\ITJ !\PUNC fitläfitu\N bäqaca\JPN gämäd\N täs?räw\V yäqomut\REL yätäsäräqut\REL yäsu\N bägoc\N näbäru\AUX #\PUNC

The Interface desigined for the Amharic part of speech tagger




Amharic Parts of Speech Tagger
Tag From File
Tag Sentence By Sentence
Tag Typed Sentence
Exit
Tag From File...
a:
A:\
postagger
TestSet.txt
TrainingSet.txt
Tag
Save
Back
Start
10:16 PM


Amharic Parts of Speech Tagger
Tag From File
Tag Sentence By Sentence
Tag Typed Sentence
Exit
Tag From File Sentence By Sentence
a:
A:\
postagger
TestSet.txt
TrainingSet.txt
Start
Continue
Tag
Save
Back
Untagged Sentence
Tagged Sentence
Start
10:11 PM


Amharic Parts of Speech Tagger
Tag From File
Tag Sentence By Sentence
Tag Typed Sentence
Exit
Tags any typed text and diplay the tagged text
Tag Typed Sentence
Sentence to tag
Tag
Refresh
Back
Tagged Sentence
Start
10:13 PM

Declaration

This thesis is my original work and has not been submitted as a partial requirement for a degree in any university

Mesfin Getachew
June 2001

The thesis has been submitted for examination with our approval as university advisors.

Ato Tesfaye Biru

Dr. Abebe Gebretsadik