ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

DESIGN AND DEVELOPMENT OF AUTOMATIC MORPHOLOGICAL SYNTHESIZER FOR AMHARIC PERFECTIVE VERB FORMS

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTERS OF SCIENCE IN INFORMATION SCIENCE

BY

KIBUR LISANU WUDINEH

JUNE 2002

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

DESIGN AND DEVELOPMENT OF AUTOMATIC MORPHOLOGICAL SYNTHESIZER FOR AMHARIC PERFECTIVE VERB FORMS

BY

KIBUR LISANU WUDINEH

Signature of the Board of Examiners for Approval

_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

DEDICATION

This thesis is respectfully dedicated to my late father Lisanu Wudineh. Though you are no longer with us in flesh, it is my deep belief that your spirit will grace me at this successful moment of my life.

ACKNOWLEDGMENT

My first, and most earnest, acknowledgment must go to my advisors Dr. Yonas Admasu, W/o Woinshet Abdella and Ato Solomon Berhanu for their invaluable advice, patience and constructive criticisms, without which this thesis would not have become a reality.

My heartfelt thanks go to Ato Tesfaye Birru, Ato Daniel Aberra, Ato Workshet Lamenew, Ato Million Meshesha, Ato Mesfin Getachew, Ato Sebsibe H/Mariam, and Ato Ermias Abebe who were always ready to answer to my queries in the entire process of researching and writing this thesis.

Far too many people to mention individually have assisted in so many ways during my work. They all have my sincere gratitude. In particular, I would like to thank the late Getu Assefa, Tesfaye Simachew, Haileyesus Aragay, Nakachew Mekonnen, all instructors of Unity College and Paulos Gemechu, Samuel Eyasu, Lemma Negussie, Fikru Amanuel, Abeje G/Medhin of the Department of Mathematics, Yohanes Seifu, Yeworqzaf Haile, classmates, instructors, friends, and family members.

A penultimate thank-you goes to all my beloved friends and all the staff of SISA, who helped me reach this moment with more ease than I could have anticipated. Because they were always there when I needed them most, and never once complaining about how frequently I visit, they deserve far more credit than I can ever give them.

My final, and most heartfelt, thanks must go to all those who stood by side during the misfortunes and hardships I faced.

ABBREVIATIONS AND SYMBOLS USED

Symbols

[]	What is inside is a root
' '	What is inside is a word.
/ /	What is inside is a morpheme
( )	What is inside is a gloss
+	Morpheme break
0	null character

Abbreviations

NL	Natural Language
NLP	Natural Language Processing
NLU	Natural Language Understanding
CDs	Compact Discs
ECSA	Ethiopian Central Statistical Authority
POS	Part-of-speech
TLM	Two Level Model
RMS	Root Mean Square
1psc	First person singular common (can be applied both for feminine and masculine)
1ppc	First person plural common
2psm	Second person singular masculine
2psf	Second person singular feminine
2ppc	Second person plural common
3psm	Third person singular masculine
3psf	Third person singular feminine
3ppc	Third person plural common

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Natural Language processing plays a significant role in increasing computers' capability to understand natural languages. Morphological synthesis is one aspect of the task of understanding natural language, the language by which most human knowledge is recorded.

Morphological synthesis or generation is a process of returning one or more surface forms from a sequence of underlying (lexical) forms. Morphological synthesis systems are used as components in many applications, including machine translation, spell-check, speech recognition, dictionary (lexicon) compilation, POS tagging, morphological analysis, conversational systems, automatic sentence construction and many others.

Today, synthesizers of different kinds have been developed for languages that have relatively wider use internationally. The same cannot be said for Amharic, the working language of the Federal Government of Ethiopia, and one of the major languages of the country (Bender, 1976).

This study is, thus, an attempt to develop a prototype automatic morphological synthesizer for Amharic, specifically for perfective verb forms.

In this study, algorithms that take into consideration the morphological properties of Amharic are developed from scratch and applied, as there are no previous such attempts.

The study adopts the combination of rule-based and neural network approaches to design and develop a prototype, referred as *AmharicMorphologicalSynthesizer*. The rule-based approach generates all the roots successfully where as the neural network predicts the type of roots in the test data set with an accuracy of 81.48%. For a new case consisting of 14 roots the neural network identifies type A perfective verb forms with an accuracy of 80%, type B perfective verb forms with accuracy of 25% and that of type C perfective verb forms with an accuracy of 100%.

The thesis, in short, describes processes of automated morphological synthesis from manually synthesizing words to developing a prototype and conducting an experiment with it. The result obtained using the small manually constructed root table will encourage the undertaking of further research in the area, especially with the aim of developing a full-fledged Amharic morphological synthesizer.

CHAPTER ONE

INTRODUCTION

1.1 Background

Language is one of the fundamental aspects of human behavior and it constitutes a crucial component of our lives. In its written form it serves as a means of recording information and knowledge on a long term-basis and transmitting what it records from one generation to the next. In its spoken form it serves as a means of coordinating our day-to-day life with others (Allen, 1996).

Linguistics can be defined most simply as the study of languages, particularly, natural languages. Natural language is a set of symbols and conventions used by human beings for communication purposes (Gazdar, 1996). The academic discipline that studies computer processing of natural language (NL) is known as natural language processing (NLP) or computational linguistics (Ibid.). Grishman (1984) states that NLP involves the development of computational models of a language, and based on this developing computer programs that can analyze NL and act appropriately on the information contained in the text or information.

A large part of the information stored in bibliographic retrieval systems consists of natural language data, and many users would prefer, given the choice, to approach retrieval systems by using natural language formulations of their information needs (Salton, 1983). Salton (1989) also argues that the use of natural language search statements could raise the effectiveness as well as the efficiency of the retrieval operations by making possible the formulations of precise requests that correctly reflect user needs and simplify the user-system interactions.

To make use of natural language search statements, intensive research should be conducted at the different levels of language processing. There are different levels of natural language processing: phonological (i.e. sounds or combinations of sounds), morphological (processing of individual word forms), lexical (deals with the procedures operating on full words), syntactic (is designed to group the words of a sentence into structural units), semantic (adds contextual knowledge to the purely syntactic process in order to restructure the text into units that represent the actual meaning of a text), and pragmatic (uses additional information about the social environment in which a given document exists) (Salton, 1983). This study is based on the morphological level of natural language processing. Thus, NLP demands deep Natural Language Understanding (NLU) and modeling 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.

However, NLP is an extremely complicated task. This complication emanates from the fact that natural language involves a large number of classes and relationships whose existence is not transparent from the surface structure of the natural language (Mao, 1997). In connection with this, Mesfin (2001) also states that the assignment of words, for example Amharic words, to their appropriate classes and also identifying the relation that they have with other words in a sentence, paragraph or document is difficult. Such complication becomes even worse for highly inflected Semitic language such as Amharic (Girmay, 1992). In such languages, different word forms can be generated from a single basic unit such as the root. The following example shows just a few of the many word forms that can be obtained from a single perfective root (see section 3.3 of chapter 3 for detailed discussion) *[sbr]* and a subject marker suffix 3psm (third person singular masculine).

No.	Phonetic Representation	Amharic Equivalent	Gloss
1	alsäbbäräm	አልሰበረም	He did not break
2	Säbbärä	ሰበረ	He broke (something)
3	säbabbärä	ሰባበረ	He broke (something) over and over
4	assäbbärä	አሰበረ	He helped someone break something
5	assabärä	አሳበረ	He helped someone break something
6	täsabbärä	ተሰበረ	He broke with someone
7	täsäbabbärä	ተሰባበረ	He was broken over and over
8	assäbabbärä	አሰባበረ	He made some people break each other
9	käsäbbärä	ከሰበረ	If he broke (something)
10	käsäbabbärä	ከሰባበረ	If he broke (something) over and over
11	läsäbbärä	ለሰበረ	For he broke (something)
12	alsäbabbäräm	አልሰባበረም	He did not break again and again
13	täsäbbärä	ተሰበረ	He is broken
14	kätäsäbbärä	ከተሰበረ	If he is broken

Table 1.1 Some generations of the root *[sbr]* and a subject marker suffix 3psm (ä).

For computers to understand Natural Languages, they should be made to handle such 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 this regard, works in computational linguistics or NLP systems tried to develop a system for processing NL at different levels of complexity to have a general NLU (Natural Language Understanding) system (Allen, 1996). There are, for instance, systems developed for processing NL at phoneme, word, sentence, and pragmatic levels. These systems are developed in such a way that the output of a lower system can serve as an input for the next higher level. For instance, the output of a morphological synthesizer that works at word level could serve as an input for syntactic and semantic parsers that work at sentence level (Uibo, 2001). The

morphological synthesizer in this case will enable one to generate the surface form (e.g. spies) from its constituent distinct parts called morphemes (e.g. /spy / and /s/). That means /spy/ **+ /**s/**= /**spies/. Thus, the morphological synthesizer will accept /spy/ and /s/ as an input to generate the surface form /spies/. Similarly, the Amharic verb **/**gäddälä/ (he killed) is derived from /gäddäl-/ (the notion of killing) and (ä)- 3psm subject maker (i.e. gäddäl + ä = gäddälä).

Morphological synthesizers have vital role in NLP systems. They are used to generate surface word forms, which are the ones that are found in everyday communication, from lexical components that could be stored separately in different databases (lexicons). Such systems are used as a subcomponent of NLP in applications like machine translation, dictionary (lexicon) development, and spelling and grammar checking (Harris, 1985). Thus, it is the purpose of this study to explore the possibility of developing an automatic morphological synthesizer useful for generating Amharic words.

1.2 Applications of Natural Language Processing

NLP can be applied in a number of areas. According to Grishman (1984), 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
- Information retrieval

Other applications of NLP include term and name identification, word sense disambiguation, morphological analysis, morphological synthesis, unknown word processing, natural language generation, data mining and entity extraction and parts of speech tagging (Mao, 1997).

1.3 Statement of the Problem and Its Justification

Access to information has become a critical factor to the success of all sectors of a given society. Individual members of the society need information of one kind or another for their daily activities, their decision-making process and for social communication, interaction and mobility. At the institutional level, managers, policy makers, planners, scientists, technicians, and the different arms of the government machinery, etc need information in the planning and execution of their programs. At both levels of need (that is, at the individual and institutional levels), access to the right information at the right time is a pre-requisite for functional efficiency (Birungi, 1995).

Particularly in developing countries, information is long believed to be a corner stone of rapid social and economical development (UNESCO, 1976). As Kuznets (1966) has also remarked, the economic development of any country depends upon the effective utilization of its store of information on development activities.

The unfortunate situation, however, is that, most often developing countries have no systematic programs for the collection, analysis and dissemination of available

information to the potential users. The individual and institutional decision-makers, scientists and researchers remain unaware of the existence of relevant information. As a result, the quality of their decision may be poor (Birungi, 1995). Today, a large amount of information is available on the Internet and CDs. Developing countries like Ethiopia can facilitate their overdue dream of development by making this store of knowledge accessible to their citizens. One of the barriers to this is the absence of online machine translation systems that can translate texts from a foreign language to a local one; say, from English to Amharic.

The 1994 nationwide census of Ethiopia shows that out of 36,626,387 people aged ten years and above 76.6% (or 28,045,040 persons) fall under the category of illiterate. Of those categorized as literate, nearly half of (i.e. 4,417,005 people) are 6 graders or below (ECSA[1], 1998). Taking into account the fluency of English at that level, the problem of getting pertinent information through the Internet can clearly be observed. Thus, the existence of machine translation systems that require morphological synthesizers as a component are of paramount importance for the delivery of electronic resources (such as Internet and CDs) to the population at large in their mother tongues.

Machine translation is only one of the applications of natural language processing. Parsing, tagging, morphological analysis and morphological synthesis are, for instance, other applications of NLP (Grishman, 1984). Concerning Amharic only a limited number of studies have been conducted in the area of computational linguistics. In fact, most of the studies mainly focus on the area of character recognition (Worku, 1997; Ermias, 1998; Dereje, 1999; Millon, 2000) and text retrieval (Birru, 1992; Nega, 1999; Saba, 2001).

[1] ECSA is an acronym for Ethiopian Central Statistical Authority

As far as I know, only two studies have been conducted in the area of NLP for Amharic, Abiyot (2000) and Mesfin (2001). Abiyot's research attempts to address the need for having a word parser for Amharic language, which its absence, as clearly stated by Abiyot, will make NLP difficult for Amharic. Mesfin's work, on the other hand, focuses on experimenting with the application of stochastic method to Amharic part of speech (POS) tagging. His work lacks some word-level processes, such as distinguishing numbers (plural or singular) and sex (masculine or feminine).

Like word parsing (morphological analysis) and POS tagging, morphological synthesis is also another task that should be addressed in the area of NLP of the Amharic language. The absence of morphological synthesis systems will have an effect on researches in machine translation, spell-check, speech recognition, dictionary (lexicon) compilation, POS tagging, morphological analysis, conversation systems, automatic sentence construction, etc. (Grishman, 1984). Studies in morphological synthesis will also contribute a lot in the effort to increase computer processing of the Amharic language (Abiyot, 2000). But, to the best of my knowledge, no research has been undertaken so far in the area of automatic morphological synthesis for Amharic. The absence of morphological synthesis systems limits the effort of making computers work comfortably with the Amharic language. For instance, further and higher forms of studies such as parsing (syntactic and semantic) and machine translation that help to quickly and easily retrieve information in the language may be difficult without having morphological synthesis systems first. Moreover, the lack of a morphological synthesis system hinders the undertaking of works related to dictionary compilation, spell-check, and speech recognition and generation in Amharic (Abiyot, 2000; Mesfin, 2001).

Therefore, it is worth conducting research and also to develop an automatic morphological synthesizer for Amharic based on the property of Amharic perfective verb forms. Amharic perfective verb forms are taken for this study because they are the basis for the derivation of all Amharic verb forms and their subsequent derivations (Wedekind, 1996). Thus, algorithms developed for Amharic perfective verb forms can easily be modified for all the other verb forms and their derivations, which constitute 75% to 80% of the entries of Amharic dictionaries (Leslau, 1973).

1.4 Objectives of the Study

1.4.1 General Objective

The general objective of this study is to investigate a development option for an automatic (computer-based) word synthesizer for Amharic perfective verb forms.

1.4.2 Specific Objectives

With the aim of achieving the above general objective, the study has attempted to address the following specific objectives:

1. Study the morphological property of Amharic in general and that of perfective verb forms in particular to identify properties useful for automatic morphological synthesis;
2. Study the type of lexicons required for morphological synthesis, and design the lexicons accordingly;
3. Review the various techniques (or approaches) suggested for the development of an automstic morphological synthesizer;
4. Prepare training and running data sets to be used in the experiment;
5. Develop and test a prototype of an Amharic morphological synthesizer, which will implement the findings of the study;

6. Forward recommendations for further study.

1.5 METHODS

For the successful completion of this study, the following methods have been used.

1.5.1 Literature Review and discussion with experts

A number of resources including books, research reports, journal articles, manuals, and other published and unpublished documents (including those from the Internet) have been used for the following purposes:

- To comprehend the morphological structure of Amharic words in general and that of perfective verb forms in particular;
- To study techniques or approaches in morphological synthesis and to adopt one that is found appropriate to the current research work;
- To examine and select the appropriate machine learning algorithm;
- to know how to develop corpus data for morphological synthesis research work.

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

1.5.2 Lexicon and algorithm development

A databse labeled as ***MorphoSynthesisLexicon*** is developed as a knowledge base. This database consists of 7 tables. The tables are RootTable, TemplateTable, SuffixTable, VowelChangeTable, ConsonantChangeTable, PalatalizationTable, and TranslationTable.

Algorithms are designed from scratch as there are no previously designed algorithms for this purpose based on the morphological properties of the language to generate perfective verb forms from an input root and suffix (subject marker or object marker).

The neural network (see section 2.3.1.1 of chapter two) requires a large data to function properly. Since such a data is not available for Amharic, data consisting of 273 roots with their type (see section 3.3 of chapter three) for training (NuralRootTable5) and 14 roots without their type for running (Runningfacts) have been prepared. Microsoft Excel 3.0 has been used for the data preparation.

1.5.3 Development of the prototype morphological synthesizer

In order to exploit the benefits of rule-based and neural network approaches, a hybrid approach is adopted in the design of the prototype morphological synthesizer, referred as *AmharicMorphologicalSynthesizer*, in this study.

A combination of two rule-based approaches - CV-based (see section 2.3.2.1 of chapter two) and Two-Level Model (TLM) (see section 2.3.2.2 of chapter two) of morphology are selected based on the morphological properties of Amharic and used in this study. The CV-Based approach is used to form stems from consonantal roots and templates. TLM is then used to handle the phonological and morphophonemic processes (such as assimilation) involved in word formation.

Different lexicons and algorithms required for the development of an automatic morphological synthesizer following these two approaches are also designed from scratch, as there are no previously developed Amharic morphological synthesis algorithms. In the development, effort is made to consider the phonological and

morphological properties of Amharic words. A prototype morphological synthesizer for Amharic perfective verb forms is also developed to test the accuracy of the algorithms. The prototype synthesizer is developed using Microsoft Visual Basic version 6.0. Visual Basic is chosen for its user-friendly functionalities that enable easier implementation of the algorithm.

The neural network is used to predict the type of the roots not incorporated in the database. Once the type of the root is predicted, all the possible templates are retrieved from the template table of the *MorphoSynthesisLexicon* database and the process of stem and word formation follows. For this purpose, the neural network software *BrainMaker* is used and it displays a promising result. The reason of selecting *BrainMaker* software is availability.

1.5.4 Testing Techniques (or Procedures)

The test has been conducted in two phases. The first phase was aimed at evaluating the performance of the rule-based approach by selecting the roots *sbr* of type A, *flg* of type B, *mrk* of type C, *lkk* of type B and C, *lqq* of type A and C, *lsn* of type A and B, and *hlf* of type A. The above roots have been selected by domain experts (Linguists) based on the representativeness of their type. The errors encountered during experimentation have been corrected and the experiment done iteratively until the result is found to be satisfactory.

The second phase of the experiment was conducted to see the predictive power of the neural network. For this purpose, a table (*NuralRootTable5*-discussed in section 4.9 of chapter four*)* consisting of 273 roots (10% of which constitutes the test data set) with their type is prepared for training and 14 roots without their type to be used as running

fact is prepared and stored in a table (*Runningfacts*- discussed in section 4.10 of chapter four*)*.

The *NuralRootTable5* table was used to train *BrainMaker.* Based on the training, a number of models have been built. The data in the running facts table (*Runningfacts)* have been submitted to these models in order to select the best model. Accordingly, the results from the selected model have been reported in section 5.7 of chapter five.

1.6 Application of the results and beneficiaries

Morphological synthesis systems are useful in many areas of NLP for Amharic. Thus, the beneficiaries of this study include researchers who are, or wants to be, involved in increasing the capability of computer processing in Amharic. The study could also be used:

- As a base for building a full-fledged morphological system for Amharic;
- As a component for the development of higher forms of NLP systems such as machine translation, spell-check, speech recognition, automatic dictionary (lexicon) compilation, POS tagging, morphological analysis, automatic sentence construction, etc.;
- For Amharic language teaching and learning;
- To define the structure of Amharic perfective verb forms;
- To build a morphological dictionary for Amharic perfective verb forms;

1.7 Limitations of the study

As described earlier, the synthesizer developed is a combination of rule based and neural network. The neural network should have been trained and tested several times on the manually prepared roots and their type table to see its actual performance. Unfortunately this was not done due, mainly, to time constraint and the unavailability

of sufficient roots and their type lexicon.

1.8 Scope

The scope of the thesis is limited to demonstrating the potential of a hybrid (Rule-Based and Neural Network) approach to develop an Automatic Synthesizer morphological for Amharic triliteral[2] perfective verb forms. The study excludes beneficative and affirmative suffixes and considers only subject and object marker suffixes.

1.9 Organization of the Thesis

This section describes the organization of the rest of the thesis. Chapter two discusses what morphological synthesis is, approaches to the development of word synthesis and issues related to word synthesis. Chapter three describes the Amharic language and the word formation processes in the perfective verb forms of the language. In chapter four the features of Amharic perfective verb forms that are used in developing the prototype synthesizer are discussed. This chapter also describes the design of the lexicons that are used as data providers to the synthesizer. The algorithms designed and the experiments conducted are discussed in chapter five. The conclusions and recommendations based on the findings of the study are presented in chapter six. A bibliography, to be used for further reading, has been following this chapter. The full set of Amharic *fidel* (alphabet) and sample output of the prototype synthesizer that are used to test its accuracy are attached in Appendix 1 and Appendix2 respectively.

[2] See section 3.3 of Chapter 3 for detailed discussion

CHAPTER TWO

MORPHOLOGICAL SYNTHESIS

2.1 Introduction

As described in the first chapter, the principal objective of this study is to design and implement a prototype morphological synthesizer for Amharic perfective verb forms. Morphological synthesis or generation is a process of returning one or more surface forms from a sequence of morpheme glosses.[3]

In this chapter, basic concepts of morphology and different approaches to morphological synthesis are discussed. Section 2.2 discusses basic terminologies of morphology while section 2.3 discusses the different approaches of morphological synthesis that have close relevancy to this study.

2.2 Basic Terminologies in Morphology

As Aronoff (1976) puts it, morphology deals with the internal make-up of words. In formal language[4], words are just arbitrary strings denoting constants or variables. Nobody would care about the morphology of formal languages. In natural languages the picture is very different. Every human language contains some hundred thousands of words. At the same time new words are continuously integrated while others drift out of use. These large numbers of words are produced from a limited collection of smaller units called morphemes (Trost, 2000). The task of morphology is thus to identify and describe the mechanisms behind this process.

[3] A paraphrase or synonyms used in a dictionary entry to provide an explanation of the sense of a word or phrase related to the headword.
[4] Formal language is any set of string like 'zzmy'. 14

The basic building blocks in morphology are morphemes. Morphemes are defined as the smallest unit in language to which a meaning may be assigned or, alternatively, as the minimal unit of grammatical analysis.

Realizing morphemes as part of a word is called morph.[5] Often, there is a one-to-one relation between morpheme and morph. The morpheme */door/*, for instance, is realized as the morph */door/*. On the other hand the morpheme */take/* is realized as the morphemes */take/* and */took/*. In such a case we speak of allomorphs.[6] Plural in English is usually expressed by the morph */s/*. There are exceptions though: in *oxen* plural is expressed through the morph */en/*, in */men/* by stem[7] vowel alternation.

The form of a morph may be free or bound. A free morph is termed 'free' because it occurs relatively freely within other words or morphemes. In other words one can say that a free morph may form a word on its own, e.g., */door/*. We call such words monomorphemic because they consist of a single morph. Bound morphs, on the other hand, occur only in combination with other forms. The majority of affixes[8] are bound morphs. For example, the word */dogs/* consist of the free morph */dog/* and the bound morph */-s/* which is an affix. Words may also consist of free morphs only, for example the morpheme */woodwork/* consists of the free morphemes */wood/* and */work/,* or bound morphs only, for example the morpheme */aggression/* consists of the bound morphemes */aggress/* and */ion/*.

[5] See section 2.2.1 for detailed discussion on Morph
[6] The different forms (pronunciations) of a single morpheme
[7] A stem is a bound form of a lexical item which typically consists of a root to which one or more morphological formatives have been added and which serves as the immediate base for the formation of some further form or set of forms.
[8] Affix is a bound morpheme, which attaches to a base (root or stem) and it could be prefix, suffix, infix or circumfix. See section 2.2.1 for detailed discussion.

As described by Trost (2000), every language typically contains some ten thousand morphs. The magnitude is significantly below the number of words. Strict rules govern the combination of these morphs to form words. The combination of morphs to give meaningful words is the concern of morphological synthesis or generation. This way of structuring words in a language makes the cognitive load of remembering so many words much easier.

2.2.1 Constitutes of a morph

Morphs in a language are composed of affixes and stems. An affix is a bound morpheme that attaches to a base (root or stem). Affixes can be prefixes, suffixes, circumfixes and infixes. A prefix is an affix that is attached infront of a base. In English, */re-/, /en-/, /in-/,* as in *reemploy, endanger, inaccessible,* are examples of prefixes. The hyphen (-) indicates the position of attachment. In Amharic, /as-/, /lä-/, /sɨlä-/, as in /as-marrä/ (he made someone be forgiven), /la-mäťťa/ (for the one who comes) and /sɨlä-bälla/, (for he ate) are examples of prefixes. A suffix, on the other hand, is an affix that is attached after a base. The plural markers */-s/* and */-oč/* of English and Amharic, respectively, are examples of suffixes, as in *plants and säwoč* (men). A circumfix is the combination of prefix and suffix that together express some feature. In Amharic the combination of the prefix *t* and *alläš* in *t*-säbr-i-alläš (you will break it) is an example of circumfix. An infix is an affix where the placement is defined in terms of some phonological condition(s). For example, in the Amharic word säbabbärä (he broke something into pieces) the vowel *a* just after the consonant b is an infix that is obtained through reduplication of säbbärä (he broke).

A morphology that uses the three types of affixes-prefix, suffix and circumfix-is called concatenatative morphology whereas the morphology that also uses infix is said to be nonconcatenatative morphology (Antworth, 1991).

Affixation and compounding are alike. Their major dissimilarity lies in the nature of morphs they combine (Trost, 2000). Affixation attaches a bound morph onto a free morph whereas compounding combine two freely standing morph to form another word forms. 'Bedroom' and 'homework' of English and *timhirt-bet* ('lesson-house', which means school), *mad-bet* ('meal-house', which means 'Kithchen') and *bet-ä-kiristian* (house-of-Christian, which means 'church') of Amharic are examples of compounds.

Reduplication is a border case of affixation. The form of the affix is a function of the stem to which it is attached, i.e., it copies (some portion of) the stem. Reduplication may be complete or partial. In the latter case it may be prefixal, infixal or suffixal. Infixal reduplications are the ones common in Amharic (Trost, 2000). Reduplication is used in Amharic to show an action done repeatedly.

Example: -

/säbbärä/ (he broke) /säbabbärä / (He broke something into pieces)

2.2.2 The Structure of Words: Morphotactics

Morphs must somehow be put together to form words. A word grammar determines the way this has to be done. This part of morphology is called morphotactics. Usually there are language-specific word grammars that help determine how morphs are put together. These word grammars put constraints on morph patterning. For example, the English word pseudohospitalization is formed from /pseudo-/, /hospital/, /-ize/ and /-ation/. But these morphemes can be concatenated randomly as follows if such word

grammars don't restrict their formation:

*[9]hospitalationizepseudo

*pseudoizehospitalation

*pseudohospitalationize

In Amharic, too, from morphemes /as-/, /bälla/ (he eats), and /-at/ different word forms can be formed, as in /bällasat/, and /asatbälla/, but the grammatically correct one is /asbällat/ ('he forced her to eat' or he had her/it eaten). Therefore, a system for morphological synthesis needs to have a component that determines the well formedness of different word forms.

Constraints on affixes are made in generating an acceptable word. Some affixes, for instance, are attached to specific word categories only. These constraints can be syntactic, phonological, semantic or purely lexical in nature (Trost, 2000). A semantic restriction on the English adjectival prefix */un-/* prevents its attachment to an adjective that already has a negative meaning (e.g. Unhappy, but not *unsad). Examples of phonological restriction are also found in Amharic too. The plural marker */-očč/* is attached to nouns with consonantal endings, while */-woč/* is for vowel endings as in *lam +očč → lamočč* ('cows') and *bärre +woč → bärrewoč* ('oxen').

Morphotactics is responsible for governing the rules for the combination of morphs into larger entities. But phonological rules may apply and change the shape of morphs. Morphophonology, a discipline that merges morphology and phonology, deal with these changes and their underlying reasons.

It is common to see phonological influences when morphs concatenate to form words. In many cases this concatenation process will induce some phonological change in the

[9] *Shows ill formed construction.

vicinity of the morph boundary. Assimilation is an example. It is a process where the two segments at a morph boundary influence each other, resulting in some feature change that makes them more similar. Take, for example, the English prefix /in–/ where the /n/ changes to /m/ before labials.[10]

in + mature → immature ***(m is the labial considered in mature)***

in+ probable → improbable ***(p is the labial considered in probable)***

The same effect is also observed in Amharic; when a vowel is attached to a dental consonant that dental is changed to a palatal (discussed in section 3.8.2 of chapter three).

Example: -

wädd + e → wädĩjje (my dear one) (in this case the dental d is changed into the palatal j due to the occurrence of the vowel e)

To sum up, a system for morphological synthesis should consider the morphotactic, morphophonological and phonological features in order to generate linguistically acceptable word forms.

2.2.3 Types of Morphology

There are two productive ways to form words from morphemes: inflection and derivation (Katamba, 1993). So, we have correspondingly inflectional and derivational morphology.

Inflection morphology deals with the combination of a word with a grammatical morpheme, usually resulting in a word of the same class as the original stem, and serving some syntactic function, for example plurals of nouns. They do not change the

[10] A consonant produced with one or both lips, or a vowel for which the lips are rounded

part-of-speech category but the grammatical function (also called morphosyntactic[11] information). The different forms of a word are produced by inflection. In English, the word 'work' is a verb, and inflectional forms like *'works',* 'working', and *'*worked' are produced by adding the 3rd person singular maker /-s/, the present continuous marker /-ing/ and the perfective /-ed/ respectively. These four word forms of 'work', i.e.'work', 'works', 'working', and 'worked' are all verbs and there is no change in the part-of-speech category due to the affixation.

On the other hand, derivational morphology creates new words (i.e., words with a different part-of-speech category) by adding a bound morpheme to a stem. Derivation can be applied recursively, i.e., words that are already the product of one derivation process can undergo the process again. The following is an example from English:

large (adj.) ➔ enlarge (en- + large) (v) ➔ enlargement (enlarge + -ment) (noun)

2.2.4 Prosodic Morphology and Nonconcatenative Morphology

Traditional morpheme theory is ideal for the description of word-building processes whereby morphemes are concatenated (i.e. are attached one after the other). This theory is not at all well suited to the task of describing nonconcatenative morphological processes involving, for example, infixation or the internal modification of the root. So, although it has been recognized for a long time that, in Semitic languages, the root, usually consisting of three consonants (e.g. *sbr* 'break'), serves as the skeleton to which flesh is added in the process of word-formation, before the advent of prosodic morphology, there was no theoretically effective way of describing this method of word-formation.

[11] Morphosyntactic: any morphologically distinguished class of words that plays a part in syntax.

McCarthy (1981) initiated prosodic morphology. He noted the similarity in the behavior of vowels introduced into consonantal roots by morphological processes in Arabic on the one hand, and that of phonological prosodies, such as tone spreading, on the other. He hypothesized that the verb in Arabic has elements arranged on three independent tiers at the underlying level of representation in the lexicon, the three tiers being the root tier (also called the consonantal tier), the skeletal tier and the vocalic melody tier. Below is an example in Amharic:

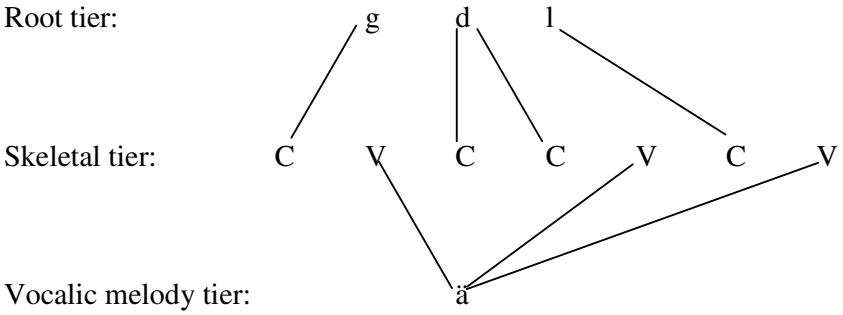


Figure 2.1 CV-based generation of the verb gäddälä

The above three independent tiers give the Amharic word gäddälä (He killed).

These three tiers are linked together by association lines. The three universal conventions for making associations between the prosodic templates and the autosogementalized strings are as follows (McCarthy, 1981):

i. The meaning of a verbal lexeme is signaled at the root-tier by the consonantal segments. Usually a verb has three consonants in its underived lexical entry in the lexicon. Thus the root *gdl* in Amharic represents the lexeme *kill*, which is realized by a variety of word-forms.

ii. The skeletal tier (which is also called the CV-tier) is like a potter's template. So, it is also called the prosodic template tier. It provides a canonical shape that is associated with a particular meaning or grammatical function. The template CVCCVCV, for instance, carries the

grammatical meaning perfective. Hence *gäddälä* means 'he killed'.

iii. The vocalic (vowel) melody tier provides information analogous to that carried in English by inflectional affixes like tense, aspect, number or derivational affixes. For instance in the above Amharic perfective verb *gäddälä,* the vocalic pattern ää indicates that the tense of the verb is past. The last vowel ä indicates that the person is second person masculine and the number is singular, as we observe in the case of English.

2.2.5 Computational Morphology

Morphology deals with the internal structure of words. Computational morphology, on the other hand, is intended to handle the task of morphology automatically with the use of computers and computational methods. Generally, the tasks involved in computational morphology can be grouped into two parts:

a) Word-form synthesis and analysis;
b) Parts-of-speech (POS)-or inflectional-category determination.

The following section discusses these tasks in detail.

2.2.5.1 Word-Form Synthesis and Analysis

Synthesis or generation is the processes of producing word forms from their constituent morphemes, whereas analysis or recognition does the reverse process, which is tokenizing word forms into their ingredient morphemes. A word form generator would accept as input a lexical form (such as /cry/ + /s/) and returns the surface form /cries/. On the other hand, the recognizer would accept as input a surface form such as /cries/ and returns an underlying form divided into morphemes, namely, /cry/ + /s/. These processes demand identification of word form components (for example stems and suffixes) and taking account of the regular phonological or orthographical alternations due to morphological, and morphophonological processes

involved.

2.2.5.2 POS or Inflectional Category Determination

Computational morphological systems operate as a morphological front end of syntactic parsers. Such syntactic parsers need not only tokenized word forms but also determine the POS or inflectional category of the entire word. The POS tag or inflectional category of words is often taken from a morpheme that serves as a "head of a word[12]" for that particular word/word form. For example, the English word 'sadness' comes from the following morphemes: the adjective stem /sad/ and the noun marker suffix /-ness/. Of these morphemes, the noun-marker suffix /-ness/ is a head of the word /sadness/, and thus, the entire word is a noun.

2.2.6 Morphological synthesis (generation)

2.2.6.1 Knowledge Required by a morphological synthesizer

The word formation process of a language should be represented correctly for the synthesizer in order to carry out the synthesis process effectively. Even though some basic knowledge is common for most synthesizers, the detailed knowledge required can vary from language to language. Pullman *et. al.* (1988) notes that the following are the three main types of knowledge that need to be represented for synthesizers.

i. Knowledge about the properties of the stored base forms of words:

ii. Knowledge about spelling or phonological changes upon affixation; and

iii. Knowledge about the syntactic or semantic properties of affixation (that is, inflectional and derivational morphology).

Moreover, the synthesizer should have knowledge of the syntactic or semantic properties of the vowel patterns of the stem, and particularly for Semitic words. This is because the vowel patterns of the stem in Semitic languages can determine the

[12] For a detailed discussion on "head of a word" see William (1981).

inflectional and derivational category of a word. For example the two Amharic words 'sɨbäri' ስበሪ (break!) (2psf[13])) and 'sɨbbari' ስባሪ (a fragment) are derived from the same root 'sbr' ስብር (break) have different meanings due to the different vowel patterns they possess.

2.2.6.2 Inputs and Outputs of Morphological Synthesis

Any synthesis task is governed by the final goal it seeks to achieve, which will determine the success or failure of synthesis. The output of a morphological synthesis of a language for NLP and linguistic can be different as they have different goals to achieve.

The study of NL for NLP involves identifying computational characteristics of the language relevant to the application at hand, and it is constrained by limitations of computational effectiveness and available technologies. The study of NL for linguistic purposes is usually motivated by the problem of language acquisition and identification of universal property of a language. Analysis of NL for linguistic purposes involves characterizing observable properties of the language and description of an entire language.

The morphological synthesizers developed so far function generally in two different ways. The first one is generating well-formed words from a sequence of morphemes (/spy/+/s/=/spies/) like that of Englex,[14] and the other one is generating as many different word forms as possible from a given stem and an affix or from a given root and an affix like that of MORPH[15]. This study will consider the possibility of generating all possible words from an input root and suffix. For instance, if the inputs

[13] 2psf stands for Second Person Singular Feminine.
[14] A computational Morphology of English.
[15] A morphological synthesizer for Germany.

to the system are the root name *[sbr]* and the suffix subject marker *3psm*, some of the words generated by the synthesizer are given hereunder. This table is given in chapter one as Table 1.1 and it is repeated here for the purpose of this discussion.

No.	Phonetic Representation	Amharic Equivalent	Gloss
1	alsäbbäräm	አልሰበረም	He did not break
2	säbbärä	ሰበረ	He broke (something)
3	säbabbärä	ሰባበረ	He broke (something) over and over
4	assäbbärä	አሰበረ	He made something broken
5	assabärä	አሳበረ	He helped someone break something
6	täsabbärä	ተሳበረ	He broke with someone
7	täsäbabbärä	ተሰባበረ	He was broken over and over
8	assäbabbärä	አሰባበረ	He helped someone break something
9	käsäbbärä	ከሰበረ	If he broke (something)
10	käsäbabbärä	ከሰባበረ	If he broke (something) over and over
11	läsäbbärä	ስለሰበረ	For he broke (something)
12	alsäbabbäräm	አልሰባበረም	He did not break again and again
13	täsäbbärä	ተሰበረ	He is broken
14	kätäsäbbärä	ከተሰበረ	If he is broken

Table 2.1 Sample generations of the morphological synthesizer developed.

2.3 Approaches to Morphological Synthesis

As discussed in Kazakov and Munandhar (2000), the different approaches to morphology are categorized as corpus based and rule-based.

2.3.1 Corpus-based Approaches

Corpus-based approaches do not strictly follow explicit theory of linguistics (Kazakov and Munandhar, 2000). The approaches are completely based on test corpora, which constitute the input data. Approaches in this category use some algorithms to learn, say about the word formation process of a language from a given corpus and perform the synthesis based on this knowledge. Moreover, the employed algorithms are subject to modification and further fine-tuning during the operation (Kazakov and Munandhar, 2000).

Corpus-based approaches are further divided into *supervised* and *unsupervised* based on the type of test corpora they use. Unsupervised approaches use heuristics or probability information generated from the test corpora to generate the morphological synthesis system (Kazakov and Munandhar, 2000). In this approach, no sample outputs are given. Kazakov and Munandhar (2000) argue that this approach reduce the cost of browsing annotated corpora.

Supervised approach, on the other hand, requires annotated text corpora. In this case a teach input is provided, which tells the system the outputs required for a given input. This study uses this approach to predict the type of the root not available in the *RootTable*[16] table. To implement this, the neural network software *BrainMaker* designed using the back-propagation algorithm is used. The annotated corpus in this study is the *NuralRootTable5*[17] table, compiled as part of this research from various sources given in the introduction part of chapter three, which consists of 273 roots with their type.

Similar researches conducted on the application of neural networks in morphological synthesis is sought for and cannot be found. The following section gives an overview of artificial neural networks.

2.3.1.1 Artificial Neural Network (ANN)

Artificial Neural Network (ANN) is a system loosely modeled on the human brain. The field goes by many names, such as connectionism; parallel distributed processing, neuro-computing, natural intelligent systems, machine learning algorithms, and artificial neural networks. It is an attempt to simulate within specialized hardware or sophisticated software, the multiple layers of simple processing elements called

[16] Discussed in detail in section 4.2 of chapter 4
[17] Discussed in detail in section 4.10 of chapter 4

neurons. Each neuron is linked to certain of its neighbors with varying coefficients of connectivity that represent the strengths of these connections. Learning is accomplished by adjusting these strengths to cause the overall network to output appropriate results (Anderson, 1992; McNeil, 1992).

Layers

Biologically, neural networks are constructed in a three dimensional way from microscopic components. These neurons seem capable of nearly unrestricted interconnections. This is not true in any man-made network. Artificial neural networks are the simple clustering of the primitive artificial neurons. This clustering occurs by creating layers, which are then connected to one another. How these layers connect may also vary. Basically, all artificial neural networks have a similar structure of topology. Some of the neurons interface the real world to receive its inputs (input layer) and other neurons provide the real world with the network's outputs (output layer). All the rest of the neurons are hidden from view (hidden layer).

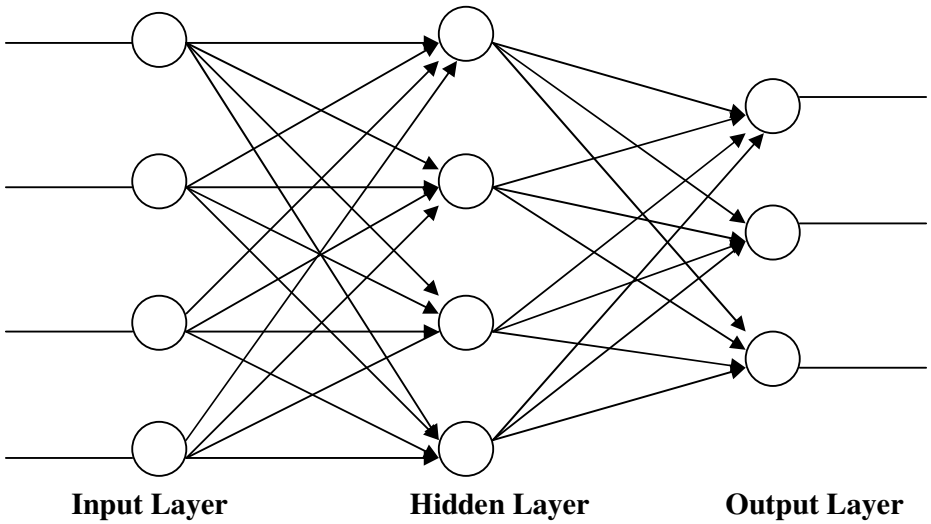


Figure 2.2 A Sample Neural Network

As the figure above shows, the neurons are grouped into layers. The input layer consists of neurons that receive input from the external environment. The hidden layer

after accepting inputs from the input layer it automatically assign weights based on the neural transfer function selected (Sigmoid function). The output layer consists of neurons that communicate the output of the system to the user or external environment.

Learning

The brain basically learns from experience. Neural networks are sometimes called machine learning algorithms, because of its connection weights (training) causes the network to learn the solution to a problem. The strength of connection between the neuron is stored as a weight-value for the specific connection. The system learns new knowledge by adjusting these connection weights.

The learning ability of a neural network is determined by its architecture and by the algorithmic method chosen for training (Taylor, 1995).

The training method usually consists of one of the following.

1. Unsupervised learning: The hidden neurons must find a way to organize themselves without help from the outside. In this approach, the network is provided with inputs but not with desired outputs. The system must then decide what features it will use to group the input data. This is often referred to as self-organization or adaption. This is learning by doing.
2. Supervised learning: In this case both the inputs and the outputs are provided. The network then processes the inputs and compares its resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights which control the network. This process occurs over and over as the weights are

continually tweaked. The set of data which enables the training is called the training set. During the training of a network the sane set of data is processed many times as the connection weights are over refined.

Application areas of neural networks

ANNs can be used for many purposes, such as

- Optimization,
- Control,
- Vision and image processing,
- Speech recognition and synthesis
- Natural language processing,
- Modeling human cognition, etc.

Basically, most applications of neural network fall into the following five categories (Anderson, 1992; McNeil, 1992):

1. Prediction: Uses input values to predict some output. E.g. predict whether, identify people with cancer risk.
2. Classification: Use input values to determine the classification. E.g. is the input the letter A, is the blob of the video data a plane and what kind of plane is it.
3. Data association: Like classification but it also recognizes data that contains error. E.g. not only identify the characters that were scanned but identify when the scanner is not working properly.
4. Data conceptualization: Analyze the inputs so that grouping relationships can be inferred. E.g. extract from a database the names of those most likely to by a particular product.
5. Data filtering: Smooth an input signal. E.g. take the noise out of a telephone signal.

2.3.2 Rule-based approaches

Rule-based approaches are based on a theory of morphology laid down by experts. This group of methods enables one to incorporate sophisticated linguistic theory, such as generative phonology, into computational morphology processes. Because of their reliance on linguistic theories, systems developed using such approaches are often efficient and produce better quality outputs (Karttunen, 1994). Moreover, Karlsson and Karttunen (2000) indicated that rule based approaches are tested quite for a long period time now, and there are a number of systems developed using this approach both for commercial and research purposes.

As described earlier, most of the morphological generators and recognizers are done using the rule-based approach. The reason being that morphological systems developed using rule-based approaches have the following advantages over those developed using corpus-based approaches. Such advantages include the following as indicated in Karlsson & Karttunen (2000):

Data-compaction: Morphological systems developed using rule-based approaches require less storage than morphological systems developed using corpus-based approaches.

High speed: Morphological systems developed using rule-based approaches are faster than those developed using the corpus-approaches.

Better Efficiency: Morphological systems developed using rule-based approaches are reported to have better accuracy than those developed using corpus-based approaches.

Better adaptability: Morphological systems developed using rule-based approaches are easier and more straightforward to twist or modify for the purpose of correcting

errors.

The methodology used in this study is the combination of rule-based and ANNs. The reason behind is to benefit from the above virtues of rule-based approaches and the predictive capability of ANNs.

There are a number of rule-based approaches for morphological synthesis. Among them are two-level model (TLM) of morphology, CV-Based approach, bottom up approach, top-to-bottom approach, and Unification based word grammar. This section reviews the first two approaches that have close relevancy to this study.

2.3.2.1 CV-Based Approach or Root-and-template morphology

In his application of autosegmental theory of morphology to Arabic, McCarthy (1981) proposes that a word can be regarded as having separate autosegmental tiers for vowels (that marks information about voice and aspect) and consonant melodies (also called root, and which conveys the basic semantic meaning) that are linked to a core template comprising C-V (also called *Binyan*) elements through universal and language-specific conventions. These association conventions are constrained by the same set of well-formedness conditions independently motivated for tone, vowel harmony and other prosodic features.

Amharic is a Semitic language whose morphology is quite similar to that of Arabic (Trost, 2000). Thus, the points raised by McCarthy in relation to Arabic morphology seem valid for Amharic.

The type of morphology characterizing Semitic languages is commonly known as nonconcatenative morphology. Nonconcatenative morphology differs from its concatenative counterpart in that, apart from prefixation or suffixation, it has a

morphology pervaded by a wide variety of purely morphological alternations internal to the stem. For example, infixing different vocalic patterns can produce different word forms. The following example could help to easily illustrate the aforesaid discussion.

Root	template	stem	Gloss	Class
sbr	$C_1äC_2C_2äC_3$	säbbär	he broke	Verb
	$C_1äC_2aC_3$	säbar	something broken	Noun
	$mäC_1C_2äC_3$	mäsbä	to break	Noun

Figure 2.3 CV-based formations of stems and words

The CV-based approach is used in this study for stem formation. As can be shown above, the stems are represented as templates. Inserting characters of a root to their appropriate templates form stems. This template matching handles morphological and phonological activities internal to the stem components such as assimilation, root reduction and extension, involved in stem formation.

2.3.2.2 Two-Level Model of Morphology (TLM)

A major breakthrough in the field of morphology came in 1983 when Kimmo Koskenniemi, a Finnish computer scientist, produced his dissertation *Two-level morphology: A general computational model for word-form recognition and generation* (Koskenniemi, 1983). Koskenniemi's model of two-level morphology was based on the traditional distinction that linguists made between morphotactics and morphophonemics, which accounts for alternate forms or spellings of morphemes according to the phonological context in which they occur. For example, the word säbɨrre (I having broken) is formed from the stem /säbr/ and the morpheme /e/.

Koskenniemi's model is two-level in the sense that a word is represented as a direct, letter-for-letter correspondence between its lexical or underlying form and its surface form. An example showing the generation of the word 'chased' is given in two-level representation as follows. In this case, + is a morpheme boundary symbol and 0 is a null character.

Lexical form	c	h	a	s	e	+	e	d
Surface form	c	h	a	s	0	0	e	d

Table 2.2 The generation of the word chased using the two-level model

Shortly after Koskenniemi's dissertation appeared, Lauri Karttunen and others produced a LISP implementation of Koskenniemi's two-level model and dubbed it KIMMO (Karttunen, 1983). The main components of the KIMMO parser are shown in Figure 2.4. It had two analytical components: the rules component and the lexical component, or lexicon. First, the rules component consisted of two-level rules that accounted for regular phonological or orthographic alternations, such as /chase/ versus /chas/. Second, the lexicon listed all morphemes (stems and affixes) in their lexical form and specified morphotactic constraints. For example, the lexicon would have included lexical entries for the verb stem /chase/ and the suffix /-ed/, and would have specified their relative order. Using these data components were two processing functions, the generator and the recognizer. The generator would accept as input a lexical form such as /spy/+/s/ and return the surface form /spies/. The recognizer would accept as input a surface form such as /spies/ and return an underlying form divided into morphemes, namely /spy/+/s/, plus a gloss string such as N+PLURAL.

PC-KIMMO is a new implementation for microcomputers of a program dubbed KIMMO after its inventor Kimmo Koskenniemi (see Koskenniemi 1983). It is of

interest to computational linguists, descriptive linguists, and those developing natural language processing systems. The program is designed to generate (produce) and/or recognize (parse) words using a two-level model of word structure in which a word is represented as a correspondence between its lexical level form and its surface level form.

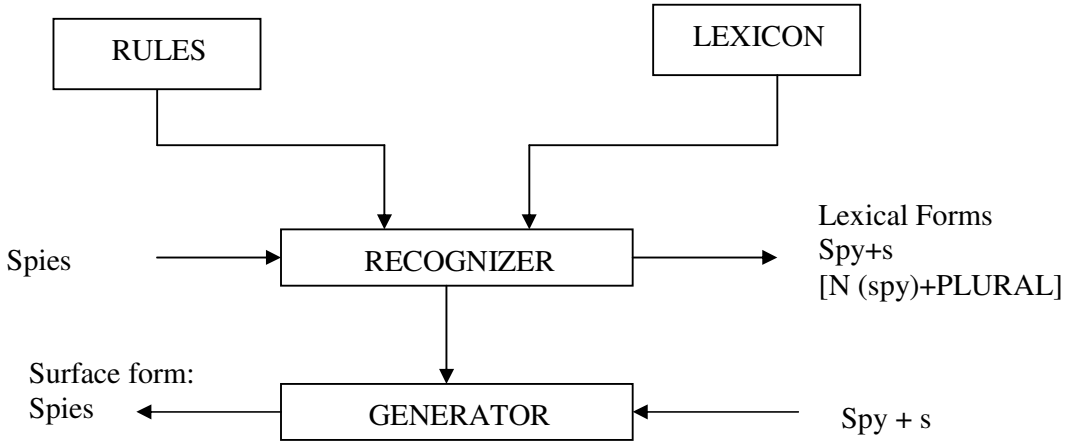


Figure 2.4 Main components of Karttunen's KIMMO Parser

TLM of morphology is used in this study during word formation process through suffixation. The correspondence between the underlying form and its lexical equivalent is represented in the tables *VowelChangeTable*, *ConsonantChangeTable*, and *PalatalizationTable* of the *MorphoSynthesisLexicon* database. Sample records in the above tables can be obtained in chapter four.

2.4 Summary

Various computational morphology concepts are discussed in this chapter. The discussions included in this chapter on neural networks and rule-based approaches will be applied in chapter five to develop the morphological synthesizer. The next chapter will discuss Amharic verb formation.

CHAPTER THREE

VERB FORMATION IN AMHARIC

3.1 Introduction

This chapter discusses the verb formation processes in Amharic, particularly the perfective verb forms. As indicated in the first chapter, the study focuses on perfective verb forms that are the basis for the derivation of all Amharic verb forms and their subsequent derivations (Wedekind, 1996). To emphasize the importance of verbs in Amharic, Wedekind (1996) says that the verb is the language.

The analysis and discussions in this chapter are based on the data and analysis from Abiyot (2000), Baye (1986[18]), Getahun (1990), Hirut (1998), Dawkins (1969), Bender and Hailu Fullass (1982) (henceforth BF), Mullen (1986), Alemayehu (1987), Habte Mariam (1994), Girmay (1992) and Leslau (1967). Further details on the subject can be obtained from these sources.

3.2 The Amharic writing system

In this study, phonetic symbols are used to represent the Amharic alphabet assuming that it is possible to develop a program that converts the phonetic symbols to their equivalent Amharic texts written using the Amharic alphabet. A list of the Amharic alphabet (fidel) adopted from Leslau and used in this study is found in appendix 1. A detailed discussion of the Amharic writing systems is found in Abiyot (2000).

[18] The year in Baye is given in Ethiopian calender 35

3.3 Amharic verb classes

BF (1978) classify Amharic verbs based on the following six criteria:

i. Consonantal skeleton (number of radicals);
   - Monoliteral (consisting of only one consonant);
   - Biliteral (consisting of only two consonants);
   - Triliterals (consisting of only three consonants);
   - Quadriliteral (consisting of only four consonants);
   - Quiniliteral (consisting of only five consonants).

ii. Pattern of gemination of consonants;
   - Type A: penultimate (second from the last) consonant that geminates in perfect only;
   - Type B: are verbs whose penultimate consonant geminates throughout the conjugation;
   - Type C: are verbs whose penultimate consonant geminates in perfect and imperfect only;

iii. Occurrence of vowels other than the normal ä;

iv. Identical consonants in sequence;

v. Presence of w, y, h (in the verbal root);

vi. Occurrences of initial 'a' or 't' in base forms.

Cohen (1978), as cited by Habte Mariam (1994), classifies simple (non-derived) verbs on the basis of their conjugational[19] pattern as their phonetic content into the following major and minor divisions:

The major group includes:

i. Triliterals: type A,

ii. Triliterals: type B,

[19] Conjugation means to give various inflectional endings of a verb, i.e. voice, mood, tense, number and person.

iii. Quadrilaterals

iv. Quniliterrals and sexiliterals,

The minor group includes:

i. Verbs with their second and third radicals identical,

ii. Verbs with initial /a/, and

iii. Verbs with /o/ or /e/ after their first consonant.

Dawkins (1969) divides Amharic verbs into three major types, with several sub-groups within each type, and each subgroup being brought about by the special phonetic composition of its root. Type A verbs are triliterals with their penultimate consonant geminated only in the perfect, while type B verbs consist of triliterals that geminate their penultimate consonant through the conjugation. Type C verbs have three consonants and geminate their penultimate consonant in the perfect as well as in the imperfect.

Observing the proliferation of classification, especially in BF (1978) and in Dawkins (1969) to a lesser degree (if we take into account their subclassifications), Habte Mariam (1994) proposes the following four classes on the basis of grammatical considerations alone, without taking into account the phonetic content of the verbal stems. He remarks that such minor classes as those that are brought about by the phonetic content of the stems do belong to one or the other of the four classes and their minor anomalies should be accounted for in terms of phonological processes.

Type A consists of triliteral simple stems that spread the vocalic melody of the conjugation to a maximum of two places and that geminate the penultimate consonant only in the perfect. A common example of type A verbs is *säbbär-ä* (he broke something).

Type B triliteral stems are characterized by the spreading of the vocalic melody to a maximum of two places and by the gemination of the penultimate consonant throughout the whole conjugation. An example of type B verbs is *fälläg-ä* (he searched for something).

Type C triliteral stems, as exemplified by *marräk-ä* (he captured [prisoner of war]), are characterized by the vowel *a* after the first radical and by the gemination of the penultimate consonant in the perfect as well as in the imperfect.

Type D verbs are simple quadriliteral and derived stems. They are characterized by the spreading of the vowel of the conjugation to a maximum of three places and by the gemination of the penultimate radical in the perfect as well as the imperfect.

The classification given by Habte Mariam is adopted for this study, however ignoring what he has classified as type D. Classifying Amharic words into three as type A, B, and C is also supported by Getahun (1990). These types of verbs are all triliteral. Baye (1999) argues that all Amharic words can be reduced and extended to three radicals. Researcher's survey of the Amharic-English dictionary by Amsalu (1979) also shows that 75-80% of the Amharic verbs are triliterals.

3.4 Amharic Verbal Stem Forms.

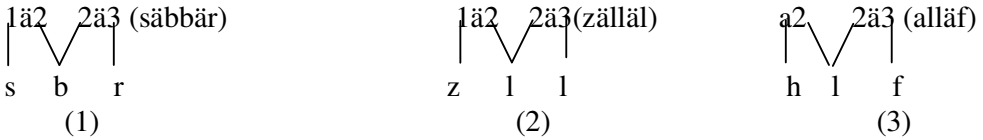
There is disparity on the classification of verbal stems in Amharic. Mullen (1986) and BF (1982) classify Amharic verbal stems into six: perfective, imperfective, gerundive, jussive, infinitive and agentive. Habte Mariam (1994) classifies them into four-

perfective, imperfective, gerundive and jussive, whereas Baye (1999) and Getahun (1990) classify Amharic verbal stems into three: perfective, imperfective and jussive. This study focuses only on the perfective verb form because it is believed to be the source of all verbal variations (tense, aspect, mood) and all the other verbal stems can be predicted from it (Wedekind, 1996).

3.5 Formation of Perfective Triradical Basic Stems in Amharic

To produce triliteral basic stems in the perfect for type A verbs, insert the root into the template as 1ä22ä3 or 1ä22ä2 or a22ä3ä, etc., as in the example below.

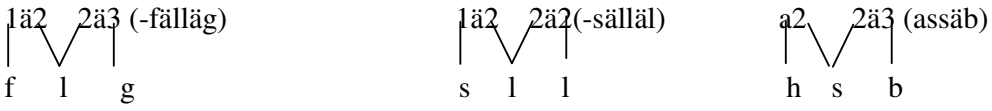
Examples:



In example (3) above /h/ is changed into /a/.

To produce triliteral basic stems in the perfect for type B verbs, insert the characters of the root into the template 1ä22ä3 or 1ä22ä2 or a22ä3ä, etc., as in the example below.

Examples:



To produce triliteral basic stems in the perfect for type C verbs insert the characters of the root into the template 1a22ä3 or 1a22ä2, etc., as in the example below.

Examples:

3.6 Formation of Perfective Triradical Derived Stems in Amharic

BF (1978) state that the main derivations in Amharic basic verbs with some indication of their syntactico-semantic functions involve **prefixing** and **internal changes**.

3.6.1 Prefixing

1. Prefix /a-/ (transitivizer)

Prefixing /a/ to an intransitive verb makes the verb transitive. The following example shows the aforesaid process.

Example; i. a. Kebede wịha t'ät't'-a

Kebede- water drink- (3psm)

(Kebede drank water)

b. Kebede saru-n wịha a-t'ät't'-a

Kebede- grass-the water made drink- (3sm)

(Kebede watered the grass)

The verb t'ät't'-a is intransitive whereas the verb a-t'ät't'-a is transitive.

ii. a. Abebe mịsa bälla

(Abebe ate launch)

b. Abebe lịj-u-n mịsa a- bälla

(Abebe fed his/the child lunch.)

The verb bälla is intransitive whereas the verb a- bälla is transitive.

2. Prefix /as-/ (causative)

When a verb is prefixed with 'as-' it indicates that the subject causes the action expressed by the underlying verb.

as – säbbärä . "he made somebody break something"

as – fällägä . "he made somebody search

something/someone”

3. Prefix /tä -/ (passive marker)

The prefix /tä/ changes the verb to passive.

For example; i. a. säbbärä “he broke ”

b. tä – säbbärä “it got broken”

ii. a. gäddälä “ he killed”

b. tä – gäddälä “ he was killed"

3. Prefix /al-/ (Negative marker)

The Amharic prefix ‘al-’ is said to be a negative marker prefix and the suffix that always come with this prefix is ‘-m’

al-säbbärä-m (He didnot break)

3.6.2 Internal changes

The internal changes of derived verbs occur in two ways. The first one involves the insertion of vowel /-a-/ after the first radical of the root. It always occurs with prefix /tä-/. It indicates reciprocity. Example:

tä- sabbäru “ they were broken.’’

tä- gaddälu “ they killed each other.”

If the subject is the cause for the reciprocal action, the stem will be prefixed with /a-/. For example;

a- sabbäru . ‘they made people break each other’

a- gaddälu ‘they made people kill each other’

The second one involves duplication of penultimate (the second from the last) radical followed by vowel /ä/. It expresses that something is done again and again or iteratively.

Example: tä- säbabbäru ‘they broke each other again and again’

tä- gädaddälu	‘they killed each other again and again’
a- säbabbäru	‘they made people break each other again and again’
a- gädaddälu	‘they made people/animals kill each other again and again’

Not all verbs undergo these derivational processes. Verbs take or fail to take the various derivations in idiosyncratic patterns according to semantic, syntactic, and phonological considerations. Therefore, in most of the cases, it is difficult to predict whether it is possible to apply a particular derivational process to a specific root. BF (1978) suggest the following criteria to be applied as a rule:

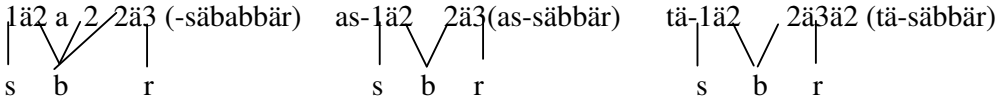
- A verb with no antepenultimate radical cannot undergo the first internal change.
- A verb with initial ‘a’ does not take the /a-/ prefix.

Leslau (1967) says that the Amharic verb has various stems derived from the basic stem. They are: the reduplicative stem and the stems formed with prefixed morphemes.

The reduplicative stem is characterized by the repetition of the second radical in triradicals (säbbär: säbabbär). The stems derived with prefixed morphemes are the tä-stem, as-stem, at-stem, a-stem, taš-stem, astä-stem, tästä-stem, an-stem, tän-stem, etc.

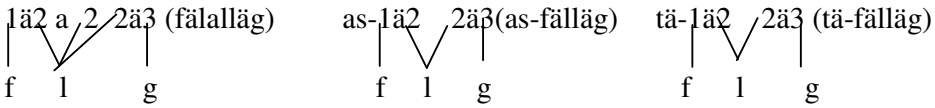
To produce a triliteral derived stem in the perfect for type A verbs insert the characters of the root into the template 1ä2a22ä3 or as-1ä22ä3 or tä-1ä22ä3, etc.

Examples:



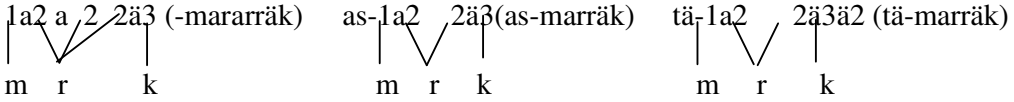
To produce a triliteral derived stems in the perfect for type B verbs insert the root in to the template 1ä2a22ä3 or as-1ä22ä3 or tä-1ä22ä3, etc.

Examples:



To produce a triliteral derived stems in the perfect for type C verbs insert the root in to the template 1ä2a22ä3 or as-1a22ä3 or tä-1a22ä3, etc.

Examples:



3.7 Inflectional Affixes of Amharic verb stems

One of the characteristics of Amharic word formation is that inflectional affixes cannot be added until stems are formed. As indicated above, inserting the root into the already established template forms basic as well as derived stems. For example, it is not correct to say ‘sbr- ä’. Phonological rules (such as palatalization) must apply only when fully inflected words are derived from verbal stems. The following is an example of consonant change through assimilation.

fälläg + k → fälläkk 'you wanted' (g changed into k).

The inflectional affixes of Amharic verb stems indicate features of person, number and gender. Amharic verbs exhibit two major types of inflectional affixes with their verb stems;

a) Subject markers
b) Object markers

3.7.1 Subject Feature Indicators

Subject feature indicators indicate the feature of the subject i.e. the features of the doer of the action. The subject feature indicator affixes are obligatory. All Amharic perfective verb forms cannot function without subject marker suffixes.

Example:

1. Y säw gäddäl-ä 'Y (masculine) killed a human being'
2. Y säw gäddäl–äčč 'Y (feminine) killed a human being'

The subject 'Y' in the first example has the feature value of person = third; number = singular; and gender = male. These feature values are the features of the suffix '– ä' of the verb 'gäddäl-ä'. In the second example the subject 'Y' has the feature value of person = third; number = singular; and gender = female, which are the features indicated by the suffix '–äčč' of the verb 'gäddäl–äčč'.

Amharic subject indicator affixes can be grouped into three paradigms, based on the stem to which they can be attached (Alemayehu, 1987). Except some affixes of the imperfective and jussive, all the remaining affixes are suffixes. The following table presents subject marker suffixes in the perfective stem.

Suffix	Features			Examples
	Person	Number	Gender	
-hu/ku	1	S	C[20]	säbbär-ku (I broke)
-n	1	Pl	C	säbbär-n (We broke)
-h/k	2	S	M	säbbär-k (You broke)
-š	2	S	F	säbbär-š (You broke)
-ačč‡hu	2	Pl	C	säbbär-ačč‡hu (You broke)

[20] C stands for both feminine (F) and masculine (M).

-ä	3	S	M	säbbär-ä (He broke)
-äčč	3	S	F	säbbär-äč (She broke)
-u	3	Pl	C	säbbär-u (They broke)

Table 3.1 Subject marker suffixes of the Perfective stem

3.7.2 Object marker Affixes

The object marker affixes are attached to a verb after the subject marker affixes are attached to the stem. All object indicators of Amharic verbs are suffixes.

For example:

Almaz Daba-n gäddäl-äčči-w 'Almaz killed Daba'

The subject 'Almaz' has the feature value of: person=3rd; number = singular; and gender = female; and the object 'Daba' has the features: person=3rd; number = singular; and gender = male. The suffix /-äčč/ indicates the features person=3rd; number = singular; and gender = female whereas the suffix /-w/ indicates the features person=3rd, number = singular, and gender = male in the verb 'gäddäl-äčči-w'.

The following table presents the object marker suffixes with their features.

Suffix	Features		
	Person	Number	Gender
-ñ	1	S	C
-h	2	S	M
-š	2	S	F
-wo(-wot)	2	Respect	C
-w;	3	S	M

-at	3	S	F
-ačč̈aw	3	Respect	C
-n	1	Pl	C
-ačč̣hu	2	Pl	C
-ačč̈aw	3	Pl	C

Table 3.2 Object Marker Suffixes

3.7.3 Combination of Suffixes and Suffix Orders of Amharic Verb

Verbs should have one of the obligatory suffixes and any one of the optional. The following table summarizes the possible obligatory and optional suffixes of Amharic perfective verb stems.

Number	***Subject Marker Suffixes (Obligatory)***	***Object Marker Suffixes (Optional)***
1	-hu/ku	-ñ
2	-n	-n
3	-h/k	-h/k
4	- š	-š
5	-ačč̣hu	-wo
6	-ä	-wot
7	-u	-ačč̣hu
8		-w
9		-at
10		-ačč̈aw

Table 3.3 Orders of suffixes to the Amharic Perfective Verb Stem

The order of suffixes in a word can affect the grammar and/or meaning of the word. For example two Amharic words: gäddäl-n- š 'We killed you (feminine) ' and gäddäl-š-n 'You (Feminine) killed us' have different meanings because of the order of the suffixes /-n/ and /-š/.

From the above discussion it is possible to see that, given a verb stem, the order of the suffixes to Amharic verbs can be expressed as:

X-S_1-S_2; Where:

X= A perfective Stem

S_1 = Subject marker suffix that occur with the perfective stem

S_2= Object marker suffix that occur with the perfective stem

For example the word: gäddäl-n-š can be written as

STEM = gäddäl

S_1 = Subject marker in the perfect=n

S_2= Object marker in the perfect=š

Mullen (1986) notes that the strict order of the affixation and the conjugation pattern of the Amharic verbs indicate that the word formation process in Amharic has three levels:

1. Creation of stems from roots. Inserting root consonants into a template, which indicates the non-concatenative part of the Amharic language, will give us the stem;
2. Produce fully inflected words (from stems) as well as derive new words from existing words, mainly by affixation of subject markers;
3. Cliticization (affixation of other inflectional affixes, such as object

markers, to the fully inflected word) and compounding.

Levels 2 and 3 indicate the concatenative nature of Amharic.

3.8 Phonological Properties

After the stem is formed through a non-concatenative process, the next procedure will be concatenation, i.e. follow levels 2 and 3 given above. When words are formed by concatenation, phonological changes that may alter the original sound can occur, mostly at the morpheme boundaries. To determine the correct forms of the word, phonological changes that have taken place during word formation should be considered carefully. As the objective of this study is on synthesizing words in the text, only processes that cause changes in a grapheme[21] are considered.

3.8.1 Vowel Changes

When two vowels come in sequence, change of vowel sounds can occur. The following table[22] summarizes the possible changes of vowel sounds that can occur in Amharic.

	a	e	i	o	u	ä
a	a	aye	ay	awo	aw	a
e	eya	eye	ey	ewo	ew	ä
i	iya	iye	i	iwo	iw	iya
o	owa	oye	oy	owo	ow	o
u	uwa	uye		uwo	uw	wa
ä	a	äye		o	äw	ä

Table 3.4 Vowel Changes

[21] The minimal distinct unit in the writing system of a language
[22] The table is obtained from BF (1978).

If the word ends with *a* or *ä* and the suffix starts with */a/,* the */a/* and */ä/* in the word will change into a as can be seen in the following example:

a + ačč*i*hu → ačč*i*hu

a + at → at

a + ačč*i*äw → ačč*i*äw

a + aläčč → aläčč

ä + ačč*i*hu → ačč*i*hu

ä + at →at

ä + ačäw →ačäw

What we saw above can also be illustrated in the following way.

Perfect stem + ä + at = Perfect stem + at

Perfect stem + ä + aččäw = Perfect stem + aččäw

Perfect stem + a + at =Perfect stem + at

In this case perfect stem + ä and perfect stem + a constitute the word.

The remaining rules apply similarly.

3.8.2 Palatalization

Palatalization causes change of sound that affects a grapheme when a dental consonant is followed by the vowels /e/ and /i/ ('palatalizers'). The following table shows the corresponding dental and palatal consonants in Amharic.

Dentals[23]	***Palatals***[24]
d	j
t	č

[23] Refers to a sound made by pressing the tip of the tongue against the teeth.
[24] The roof of the mouth consists of two distinct parts: the hard front part, called the palate, and the softer back part, called the velum. Amharic consonants that are articulated in the palatal area are called palatals.

t’	č’
z	ž
s	š
l	y
n	ñ

Table 3.5 Dental consonants with the corresponding palatal consonants

In Amharic, the common changes due to palatalization occur in the following ways.

Where d is a dental consonant and p is a palatal consonant

1. Rule one d+e→pe
2. Rule two d+ia→ pa
3. Rule three d+i→pi or d+i→p

The above three rules can be presented in a tabular form as palatalization due to /e/, palatalization due to /ia/, and palatalization due to /i/, respectively

Underlying form	Surface form
s + e	še
t+e	če
d+e	je
z+e	že
n+e	ñe
t’+e	č’e

Table 3.6 Palatalization due to ‘e’

Underlying Form	***Surface form***
s+ia	ša

t+ia	ča
d+ia	ja
z+ia	ža
n+ia	ña
t’+ia	č’a

Table 3.6 Palatalization due to ‘ia’

Underlying Form	*Surface form*
s+i	ši
t+i	či
d+i	ji
z+i	ži
n+i	ñi
t'+i	č'i

Table 3.7 Palatalization due to 'i'

The above changes occur at the morpheme boundaries

3.8.3 Consonant Changes

When two consonants come in sequence, change of consonant may occur. The following table summarizes the possible change of consonants that can occur in Amharic;

Underlying Form	*Surface form*
b+y	bi
d+y	di
g+k	kk
l+r	rr
l+y	li
m+y	mi
s+š	šš
q+k	kk

Table 3.8 Consonant Changes

The following are examples to visualize how the above rules can be used.

fälläg + k = fälläkk	'You wanted, wished'	(The rule is g + k= kk)
särräq + k = särräkk	'You stole'	(The rule is q + k= kk)
läbbäs + š = läbbäšš	'You dressed'	(The rule is s + š = š š)

3.9 Summary

This chapter has discussed about the verb formation processes in Amharic, especially about perfective verb forms. The next chapter will discuss the procedures and assumptions taken to design the lexicons used to synthesize a verb. The coming two chapters constitute the core of this study.

CHAPTER FOUR

DESIGN OF LEXICONS FOR AMHARIC VERB SYNTHESIZER

4.1 Introduction

This chapter discusses the design of lexicons essential for the prototype verb synthesizer developed for Amharic. The design process is based on the morphological properties of the language presented in chapter three and the assumptions and approaches discussed in chapter two.

The ***MorphoSynthesisLexicon*** database developed as a knowledge base consists of 7 tables. The tables are RootTable, TemplateTable, SuffixTable, VowelChangeTable, ConsonantChangeTable, PalatalizationTable, and TranslationTable. This chapter will discuss the design of each of the above tables respectively.

4.2 Design of the root lexicon (RootTable)

A number of computational linguists (Golding, 1985; Anteworth, 1995; Karttunen, 1983; Güngördü, 1995) suggest that highly inflected languages should not store numerous inflected forms of a word in the lexicons of natural language systems (NLS). Storing these forms drastically inflates the size of the lexicon. Therefore, the lexicons of NLS for such languages should have a list of root forms, together with rules to recognize and generate the inflected variant forms. This recommendation also applies to Amharic NLS since it is one of the highly inflected languages (Girmay, 1992).

As indicated in the previous chapter, most of the words in Amharic are either verbs or derived from verbs. The basic lexical element of Amharic verbs is the root form. The

root has one or more consonants. Stems are formed from the root by inserting vowel pattern. Thus, most of the words in the language can be reduced to consonant and vowel pattern. The vowel patterns are not the same for each root. The type of root determines the structure of the stem. Therefore, the type of the root should be known to determine the structure of the stem and other forms.

The design of the root lexicon is based on the classification of Amharic verbs as described in section 3.4 of chapter 3. Each root in the lexicon *RootTable* has the following fields:

- RootName: a string of up to four consonants
- RootType: the type of verb the root belongs to; which can be A or B or C

Based on this analysis a table is designed, which have the following structure.

Field Name	***Data Type***	***Description***
RootName	Text	A primary key with a maximum of 4 characters.
RootType	Text	A primary key with 1 character (A or B or C).

Table 4.1 Structure of the lexicon RootTable.

The relational schema of the *RootTable* table is given as follows.

RootTable

RootName	RootType

On words from this point, the fields underlined represent primary keys.

The following table depicts the *RootTable* table populated with sample data.

RootName	RootType
bkn	C
bnn	A
bnn	C
brk	C
brq	A
brq	C
brr	A
btn	B
bzt	C
čbt	B

Table 4.2 Some of the Records in the RootTable table

The *RootTable* table contains 145 records of which 59 are of type A, 50 are of type B and 36 are of type C. This table does not include all the triradical (triliterals) perfective roots.

4.3 Design of the template lexicon (TemplateTable)

As indicated in Chapter three, plugging each character of the root into an appropriate template forms stems (see section 3.5 and 3.6 for detailed discussion). The numbers in the templates stand for a consonant. Instead of having a separate table for the vocalic melody, vowels are inserted into the templates because it is found to be computational efficient. The prefixes, collected from sources described in section 3.1 of chapter three, are attached to the templates because:

- They are few in number (al, as, tä, kä, sla, sɨlä, sɨlas, sɨlätä) and, thus, no need of having another table;
- It will increase the efficiency of the system by relieving it from accessing another table.

The designed template table (TemplateTable) has two fields:

- TemplateName: string of up to 15 characters.
- RootType: the type of verb the template belongs; which can be A or B or C

Based on this analysis a table is designed, which has the following structure.

Field Name	*Data Type*	*Description*
TemplatetName	Text	A primary key with a maximum of 15 characters.
RootType	Text	A primary key with 1 character (A or B or C).

Table 4.3 Structure of the lexicon TemplateTable.

The relational schema of the *TemplateTable* table is given as follows.

TemplateTable

TemplateName	RootType

The following table depicts some records in the TemplateTable table.

TemplateName	RootType
1ä11a2	B
1a22ä2	C
1ä22ä2	A
1ä22ä2	B
1a22ä3	C
1ä22ä3	A
1ä22ä3	B
1ä2a22ä3	A
kä1ä2a22ä3	A

Table 4.4 Some of the Records in the TemplateTable table

4.4 Designing the Affix Database

The affixes carry different types of syntactical and semantic information, which help in interpreting different forms of a word. In Amharic once stems are formed from the

roots, words can be formed by affixation. The features of affixes considered in designing the synthesizer in this study are:

- Amharic perfective verb stems should have at least a subject marker suffix to function as a word;
- Object marker suffixes are optional. That is, they can be attached after subject marker suffixes but they cannot be affixed right after the stem. In Amharic, any subject marker suffix can be followed by any other object marker suffix;
- Affixes from the same category (i.e. subject marker suffixes or object maker suffixes) cannot come at the same time in a single verb.

4.4.1 Designing the suffix lexicon (SuffixTable)

Suffixes in the ***SuffixTable*** have a SuffixName, SuffixType, and SuffixCategory as a field, where;

- SuffixName is a string of one or more character(s).
- SuffixType is a string of not more than five characters that indicates the feature of the verb like 1psc, 1ppc, 2psm, 2psf, 2ppc, 3psm, 3psf, 3ppc etc.
- SuffixCategory is a single character that indicates the category the suffix belongs to; which can be S (subject marker suffix) or O (object marker suffix).

Based on this analysis a table is designed which have the following structure.

Field Name	***Data Type***	***Description***
SuffixName	Text	A primary key of one or more characters.
SuffixType	Text	A primary key of size 5 that indicates the features of a verb.
SuffixCategory	Text	A primary key of size 1 that indicates the category (O or S) the suffix belongs to.

Table 4.5 Structure of the lexicon SuffixTable.

The relational schema of the *SuffixTable* table is given as follows.

SuffixTable

SuffixName	SuffixType	SuffixCategory

The following table depicts some records in the SuffixTable table.

SuffixName	SuffixType	SuffixCategory
ä	3psm	S
ačäw	3ppc	O
äčč	3psf	S
aččïhu	2ppc	O
aččïhu	2ppc	S
at	3psf	O
h	2psm	S
hu	1psc	S
ïn	1ppc	S
k	2psm	S

Table 4.6 Some of the Records in the SuffixTable table

The SuffixTable table designed in this study contains all the available subject and object marker suffixes.

4.5 Design of the vowel change lexicon (VowelChangeTable)

For the verb generated to be meaningful, the system should handle vowel changes that occur when two vowels appear consecutively. The prototype synthesizer developed handles this problem by accessing VowelChangeTable table before producing the generated verb form.

The designed vowel change table (VowelChangeTable) has two fields:

- VowelPattern: string of two characters that are vowels.

- VowelReplacement: string of not more than five characters.

Based on this analysis a database is designed which have the following structure.

Field Name	*Data Type*	*Description*
VowelPattern	Text	A primary key with a maximum of 2 characters.
VowelReplacement	Text	A string with a maximum of 5 characters.

Table 4.7 Structure of the lexicon VowelChangeTable.

The relational schema of the *VowelChangeTable* table is given as follows.

VowelChangeTable

VowelPattern	VowelReplacement

The following table illustrates some records in the VowelChangeTable table.

VowelPattern	VowelReplacement
aa	a
aä	a
äa	a
ää	ä
ae	aye
äe	äye
ai	ay
ao	awo
äo	o
au	aw

Table 4.8 Some of the Records in the VowelChangeTable table

The VowelChngeTable table designed for this study entertains all the changes that may occur when two vowels come in succession.

4.6 Design of the consonant change lexicon (ConsonantChangeTable)

For the generated verb to be meaningful, the system should handle consonant changes that occur when two consonants appear consecutively. The prototype synthesizer developed handles this problem by accessing ConsonantChangeTable table before

producing the generated verb form.

The designed consonant change table (ConsonantChangeTable) has two fields:

- ConsonantPattern: string of two characters that are consonants.
- ConsonantReplacement: string of not more than three characters.

Based on this analysis a table is designed which have the following structure.

Field Name	*Data Type*	*Description*
ConsonantPattern	Text	A primary key of 2 characters.
ConsonantReplacement	Text	A string of 3 characters.

Table 4.9 Structure of the lexicon ConsonantChangeTable.

The relational schema of the *ConsonantChangeTable* table is given as follows.

ConsonantChangeTable

ConsonantPattern	ConsonantReplacement

The following table depicts some of the records in the ConsonantChangeTable table.

ConsonantPattern	ConsonantReplacement
by	bi
dy	di
gk	kk
lr	rr
ly	li
my	mi

Table 4.10 ConsonantChangeTable table populated with sample data.

The ConsonantChngeTable table designed for this study entertains all the changes that may occur when two consonants come in succession.

4.7 Design of the palatalization lexicon (PalatalizationTable)

A change occurs when a palatal consonant is followed by 'palatalizers' /e/ and /i/. This change should also be entertained before the synthesized verb is reported. Thus

the prototype verb synthesizer will access the PalatalizationTable to account for such possible changes that occur when a palatal consonant is followed by palatalizers.

The designed palatalization table (PalatalizationTable) has three fields:

- Dental: a string of one character.
- Palatal: a string of two characters.
- SurfaceForm: string of three characters.

Based on this analysis a database is designed which have the following structure.

Field Name	***Data Type***	***Description***
Dental	Text	A primary key with one character.
Palatal	Text	A primary key with a maximum of 2 characters.
SurfaceForm	Text	A maximum of 3 characters.

Table 4.11 Structure of the PalatalizationTable.

The relational schema of the *PalatalizationTable* table is given as follows.

PalatalizationTable

Denatal	Palatal	SurfaceForm

The following table depicts some records in the PalatalizationTable table.

Dental	Palatal	SurfaceForm
d	e	je
d	i	ji
d	ia	ja
n	e	ñe
n	i	ñi
n	ia	ña
s	e	še
s	i	ši
s	ia	ša
t	e	če

Table 4.12 Some of the Records in the PalatalizationTable table

The PalatalizationTable table designed for this study entertains all the changes that may occur when a dental consonat is followed by palatalizers.

4.8 Design of the translation lexicon (TranslationTable)

This table is required by the synthesizer to convert verbs in phonetic representation into the equivalent Amharic letters.

The designed translation table (TranslationTable) has two field.

- PhoneticPattern: a string of two characters that are vowels.
- AmharicEquivalent: a string of not more than five characters.

The designed translation table (TranslationTable) has the following structure.

Field Name	***Data Type***	***Description***
PhoneticPattern	Text	A primary key with 2 characters.
AmharicEquivalent	Text	A string of not more than 5 characters.

Table 4.13 Structure of the TranslationTable table.

The relational schema of the *TranslationTable* table is given as follows.

TranslationTable

PhoneticPattern	AmharicEquivalent

The following table depicts some records in the TranslationTable table.

LatinPattern	AmharicEquivalent
a	›
a	œ
b	w
ba	v
bä	u
be	u?
bi	u=
bo	x
bu	u<
č	‹

Table 4.14 Some of the Records in the TranslationTable

4.9 Design of the training root lexicon (NuralRootTable5)

The neural network software *BrainMaker* need to be trained so that a model with a better prediction capability would be developed. The training procedures used to develop a better model are discussed in section 5.7 of chapter 5. The table in which the neural network is trained, NuralRootTable5 prepared in Excel 3.0, has the following records in it.

Consonant1	VocalicMelody	Consonant2	Consonant3	RootType
K	ä	r	?	B
D	ä	d	b	B
G	ä	d	b	B
g	a	n	b	C
g	a	l	b	C
h	ä	s	b	A
k	ä	n	b	B
k	a	h	b	C
l	a	h	b	C

Table 4.15 Some of the Records in the NuralRootTable5

There are 273 records in the *NuralRootTable5* out of which 106 are of type A, 85 are of type B and 82 are of type C. This table is not complete. 10% (the default value) of all the records available in the NuralRootTable5 are taken by the neural network software as a test data set and the rest 90% as a training dataset.

4.10 Design of the running fact lexicon (Runningfact)

This table is used to test the predictive power of the developed model. A table consists of 14 roots with no root type is prepared in Excel 3.0 to measure the predictive power of the developed network (model). The following table shows some of the records in

the *Runningfact* table.

Consonant1	VocalicMelody	Consonant2	Consonant3
Z	ä	g	n
Z	ä	r	f
H	ä	f	s
L	ä	m	d
M	ä	z	z
M	ä	š	g

Table 4.16 Some of the Records in the Runnigfact

Out of the 14 records in *Runningfact* table are of type A, 4 are of type B and 5 are of type C.

4.11 Summary

In this chapter the properties of the language and features that are useful for implementing word synthesizers for Amharic perfective verbs have been discussed. Moreover, the necessary database tables for synthesizing a word have been analyzed and designed. The next chapter discusses the algorithm designed and the results of the experiment based on the design and the analysis presented in this chapter. The chapter also discusses the results of the experiment on the effectiveness of the algorithms.

CHAPTER FIVE
ALGORITHM DESIGN AND THE EXPERIMENT

5.1 Introduction

In the previous chapter an attempt was made to discuss the design of the database required by the word synthesizer. In this chapter an attempt is made to discuss the algorithms designed and implemented in this study. Also discussed are the prototype implementation of the algorithms designed, experiments conducted and results achieved.

In this study, the experiment is conducted into two phases. The first phase of the experiment indicates the algorithms, all developed from scratch for the purpose of this study, developed and the results obtained on the implementation of these algorithms. This phase is enclosed in sections 5.2 to 5.7. The second phase uncovers the experiment conducted using the neural network software BrainMaker. This phase of the experiment helps to predict the type of the root not available in the root table. It is discussed in section 5.7.

5.2 The word synthesis algorithms

A number of algorithms designed for word synthesis are available in the literature. However, most of these algorithms are designed for Western languages, such as English, which have relatively simple morphological processes as compared to Semitic languages like Amharic. Therefore, all the algorithms employed in this study are designed from scratch. The reason being that the researcher could not get access to similar systems developed using the methods, which are the combination of CV-based and TLM of morphology, adopted in this study.

To trigger the algorithms to function in order, a root is given as an input to the system. After a root is provided, three major processes are accomplished to generate all the possible words from the input. These three processes are shown as level 1, level 2 and level 3 in the overview diagram below.

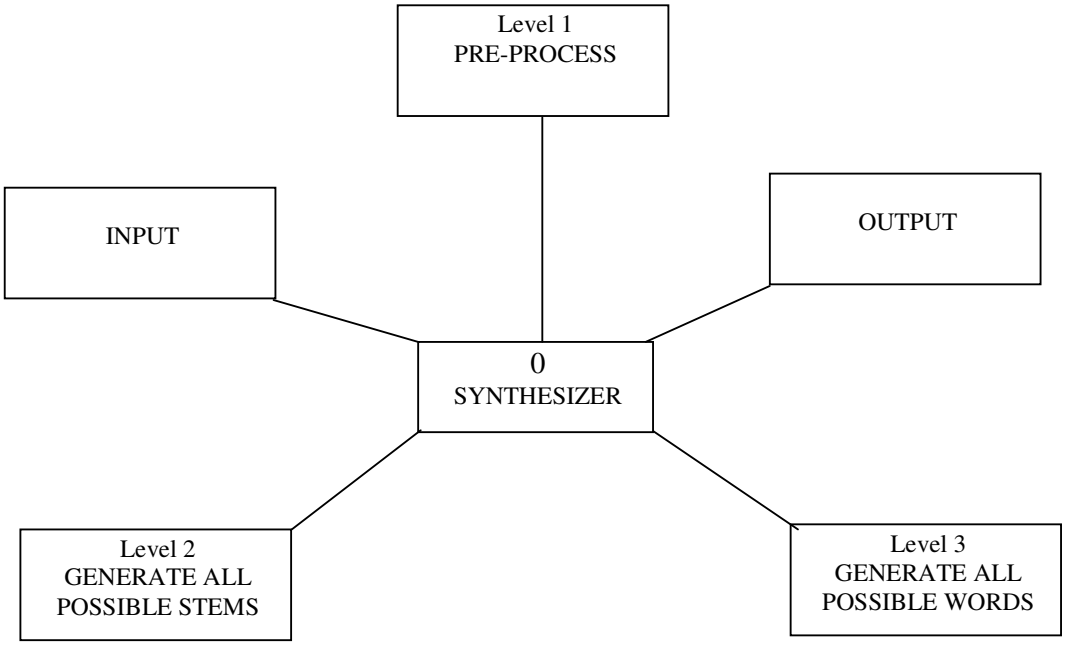
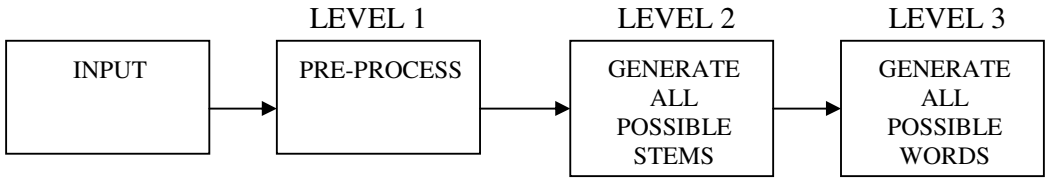


Figure 5.1 Overview Diagram of the Synthesizer

The above overview diagram can equivalently represented as follows.



After receiving the root as an input, the only input of the system to generate stems, the first process is pre-processing the input string. This process determines the type of the root in the *RootTable* table. The neural network is applied to predict the type of the root not in the *RootTable* table. The second process (Level 2) generates all the possible stems that can be formed from the root received as an input. The third process (Level 3), upon suffixation, synthesizes all the possible words from the stems found as a result of process two. Processes two and three work based on the

phonological processes identified in section 3.7 of chapter three. The processes in levels 2 and 3 are discussed in the following sections.

5.2.1 Stem formation

As indicated above, the first step of the synthesizer is pre-processing. In this stage, the type of the root is determined. The next step is generating all possible stems. This process, stem formation, begins by accepting all possible root types obtained from the root preprocessed at level 1. The algorithm below accepts the root types obtained and generate all possible stems.

Input: **Root name**
Stem Array: **An array where all generated stems will be stored**
Output: **All possible Stems**

```
Start
    Find the root name matching the root name given from the root table

    If root name found
            Get the root type for the root name
    Else
            Call function ClassifyRootType to get the root type

    Get all available templates for the root type from the TemplateTable

    For each template
        Set stem to template
        Set template number counter to the largest number on the template
        set root consonant counter to the count of different consonants of the root
        If the template number counter is equal to the root consonant counter
                Replace all occurrences of '1' on the stem with the first character of the root
                Replace all occurrences of '2' on the stem with the second character of the root
                If there is '3' on the stem
                        Replace all occurrences of '3' on the stem with the third character of the root
                Insert the stem to the Stem Array
    Return Stem Array
End
```

Figure 5.2 The Algorithm that generates all possible stems from a root

The above algorithm generates all the possible stems from an input root. For example if the input root is *hlf,* the algorithm generates the following stems.

1	alhälläf	አልኸለፍ
2	hälläf	ኸለፍ
3	hälalläf	ኸላለፍ
4	ashälläf	አስኸለፍ
5	ahhaläf	አሀለፍ
6	tähalläf	ተሀለፍ
7	tähälalläf	ተኸላለፍ
8	ahhälalläf	አኸላለፍ
9	kähälläf	ከኸለፍ
10	kähälalläf	ከኸላለፍ
11	slähälläf	ስለኸለፍ
12	alhälalläf	አልኸላለፍ
13	tähälläf	ተኸለፍ
14	slätähälläf	ስለተኸለፍ
15	slätähalläf	ስለተሀለፍ
16	slashälläf	ስላስኸለፍ
17	slahälläf	ስላኸለፍ
18	kätähälalläf	ከተኸላለፍ
19	kätähälläf	ከተኸለፍ
20	kashälalläf	ካስኸላለፍ
21	kashälläf	ካስኸለፍ

Table 5.1 Output of the algorithm given in figure 5.2

Based on the evaluation of domain experts, all the generated stems presented above are incorrect and therefore should be corrected. To correct the wrongly generated stems, the following algorithm is developed.

Input: **Stem array**
Output: **Modified Stem Array**

Start
For each stem from stem array
For each occurrences of 'o' within the stem
Check 'o' is between 'm' and 'k' or between 'k' and 'r'
Else
delete stem from the stem array
Replace all occurrences of 'h' within the stem by 'a'
Apply Vowel change Rule
Apply Consonant Change Rule
Apply Palatalization Rule
Return the Modified stem Array
End

Figure 5.3 The Algorithm that corrects the wrongly generated stems

The above algorithm corrects all the aforementioned incorrect stems generated from the root *hlf*. The result of the algorithm is given below.

1	alalläf	አላለፍ
2	alläf	አለፍ
3	alalläf	አላለፍ
4	asalläf	አሳለፍ
5	aläf	አለፍ
6	talläf	ታለፍ
7	talalläf	ታላለፍ
8	alalläf	አላለፍ
9	kalläf	ካለፍ
10	kalalläf	ካላለፍ
11	slalläf	ስላለፍ
12	alalalläf	አላላለፍ
13	talläf	ታለፍ
14	slätalläf	ስለታለፍ
15	slätalläf	ስለታለፍ
16	slasalläf	ስላሳለፍ
17	slalläf	ስላለፍ
18	kätalalläf	ከታላለፍ
19	kätalläf	ከታለፍ
20	kasalalläf	ካሳላለፍ
21	kasalläf	ካሳለፍ

Table 5.2 Output of the algorithm given in figure 5.3

All the stems shown in figure 5.3 are correct as per the evaluation of domain experts.

The algorithms presented in figures 5.2 and 5.3 work together to form correct stems of any input root.

5.2.2 Word formation

After the synthesizer gets the correct stems, it forms the words through suffixation (addition of suffixes) using the following algorithm.

Input: **StemArray (an array where all the correct stems are stored) and suffix**
Output: **all possible words**
WordArray: **an array where all generated words will be stored**

Start

Initialize a WordArray
For each stem from the Stem Array
Add a suffix to the stem and store the generated word to the WordArray

For each word from the WordArray

Apply Vowel change Rule
Apply Consonant Change Rule
Apply Palatalization Rule
Apply Negative Marker Rule

Return the WordArray

End

Figure 5.4 The Algorithm that generates all possible words from stems

If the vowel change rule is not used, then some of the words generated from a given root and suffix will not give meaning. The above can be exemplified using the words generated from root name=sbr; suffix subject marker=3psm (ä); suffix object marker=3psf (at).

1	alsäbbäräatm	አልሰበረአትም
2	alsäbbärätm	አልሰበረትም
3	säbbäräat	ሰበረአት
4	säbbärät	ሰበረት
5	säbabbäräat	ሰባበረአት
6	säbabbärät	ሰባበረት
7	assäbbäräat	አሰበረአት
8	assäbbärät	አሰበረት
9	assabäräat	አሳበረአት
10	assabärät	አሳበረት
11	täsabbäräat	ተሳበረአት
12	täsabbärät	ተሳበረት
13	täsäbabbäräat	ተሰባበረአት
14	täsäbabbärät	ተሰባበረት
15	assäbabbäräat	አሰባበረአት
16	assäbabbärät	አሰባበረት
17	käsäbbäräat	ከሰበረአት
18	käsäbbärät	ከሰበረት
19	käsäbabbäräat	ከሰባበረአት
20	käsäbabbärät	ከሰባበረት
21	släsäbbäräat	ስለሰበረአት

22	släsäbbärät	ስለሰበረት
23	alsäbabbäräatm	አልሰባበረአትም
24	alsäbabbärätm	አልሰባበረትም
25	täsäbbäräat	ተሰበረአት
26	täsäbbärät	ተሰበረት
27	slätäsäbbäräat	ስለተሰበረአት
28	slätäsäbbärät	ስለተሰበረት
29	slätäsabbäräat	ስለተሳበረአት
30	slätäsabbärät	ስለተሳበረት
31	slassäbbäräat	ስላሰበረአት
32	slassäbbärät	ስላሰበረት
33	slasäbbäräat	ስላሰበረአት
34	slasäbbärät	ስላሰበረት
35	kätäsäbabbäräat	ከተሰባበረአት
36	kätäsäbabbärät	ከተሰባበረት
37	kätäsäbbäräat	ከተሰበረአት
38	kätäsäbbärät	ከተሰበረት
39	kassäbabbäräat	ካሰባበረአት
40	kassäbabbärät	ካሰባበረት
41	kassäbbäräat	ካሰበረአት
42	kassäbbärät	ካሰበረት

Table 5.3 Output of the algorithm given in figure 5.4

According to the evaluation of domain experts, the odd numbered generated words are wrong. To correct them, the following algorithm is developed.

Input: **Word**
Output: **Modified word**

Start
While there are two consecutive vowels in the word
Replace them with the appropriate surface form from the VowelChangeTable
Return Word
End

Figure 5.5 The Algorithm that changes two consecutive vowels

The output of the algorithm given in figure 5.4 for the input root name = sbr; suffix subject marker =3psm and suffix object marker=3psf is given here below.

1	alsäbbäratm	አልሰበራትም
2	alsäbbärätm	አልሰበረትም
3	säbbärat	ሰበራት
4	säbbärät	ሰበረት
5	säbabbärat	ሰባበራት
6	säbabbärät	ሰባበረት
7	assäbbärat	አሰበራት
8	assäbbärät	አሰበረት
9	assabärat	አሳበራት
10	assabärät	አሳበረት
11	täsabbärat	ተሳበራት
12	täsabbärät	ተሳበረት
13	täsäbabbärat	ተሰባበራት
14	täsäbabbärät	ተሰባበረት
15	assäbabbärat	አሰባበራት
16	assäbabbärät	አሰባበረት
17	käsäbbärat	ከሰበራት
18	käsäbbärät	ከሰበረት
19	käsäbabbärat	ከሰባበራት
20	käsäbabbärät	ከሰባበረት
21	släsäbbärat	ስለሰበራት
22	släsäbbärät	ስለሰበረት
23	alsäbabbäratm	አልሰባበራትም
24	alsäbabbärätm	አልሰባበረትም
25	täsäbbärat	ተሰበራት
26	täsäbbärät	ተሰበረት
27	slätäsäbbärat	ስለተሰበራት
28	slätäsäbbärät	ስለተሰበረት
29	slätäsabbärat	ስለተሳበራት
30	slätäsabbärät	ስለተሳበረት
31	slassäbbärat	ስላሰበራት
32	slassäbbärät	ስላሰበረት
33	slasäbbärat	ስላሰበራት
34	slasäbbärät	ስላሰበረት
35	kätäsäbabbärat	ከተሰባበራት
36	kätäsäbabbärät	ከተሰባበረት
37	kätäsäbbärat	ከተሰበራት
38	kätäsäbbärät	ከተሰበረት
39	kassäbabbärat	ካሰባበራት
40	kassäbabbärät	ካሰባበረት
41	kassäbbärat	ካሰበራት
42	kassäbbärät	ካሰበረት

Table 5.4 Output of the algorithm given in figure 5.5

Using the algorithm in figure 5.5, the wrongly generated odd numbered words are corrected as per the evaluation of domain experts.

Like the vowel change rule, if the consonant change rule is not used, then some of the words generated from a given root and suffix will not give meaning. The input root name =flg and suffix subject marker = 2psm (h/k) is used here below to elaborate the above conjecture.

1	alfälläghm	አልፈለግህም
2	alfälläkm	አልፈለግክም
3	fälläghh	ፈለግህ
4	fälläkk	ፈለግክ
5	asfälläghh	አስፈለግህ
6	asfälläkk	አስፈለግክ
7	affaläghh	አፋለግህ
8	affaläkk	አፋለግክ
9	täfälläghh	ተፋለግህ
10	täfälläkk	ተፋለግክ
11	fälalläghh	ፈላለግህ
12	fälalläkk	ፈላለግክ
13	täfälalläghh	ተፈላለግህ
14	täfälalläkk	ተፈላለግክ
15	affälalläghh	አፈላለግህ
16	affälalläkk	አፈላለግክ
17	käfälläghh	ከፈለግህ
18	käfälläkk	ከፈለግክ
19	käfälalläghh	ከፈላለግህ
20	käfälalläkk	ከፈላለግክ
21	släfälläghh	ስለፈለግህ
22	släfälläkk	ስለፈለግክ
23	alfälalläghm	አልፈላለግህም
24	alfälalläkm	አልፈላለግክም
25	täfälläghh	ተፈለግህ
26	täfälläkk	ተፈለግክ
27	slätäfälläghh	ስለተፈለግህ
28	slätäfälläkk	ስለተፈለግክ
29	slätäfalläghh	ስለተፋለግህ
30	slätäfalläkk	ስለተፋለግክ
31	slasfälläghh	ስላስፈለግህ
32	slasfälläkk	ስላስፈለግክ
33	slafälläghh	ስላፈለግህ
34	slafälläkk	ስላፈለግክ
35	slafalläghh	ስላፋለግህ
36	slafalläkk	ስላፋለግክ
37	kätäfälalläghh	ከተፈላለግህ

38	kätäfälallägk	ከተፈላለግክ
39	kätäfällägh	ከተፈለግህ
40	kätäfällägk	ከተፈለግክ
41	kasfälallägh	ካስፈላለግህ
42	kasfälallägk	ካስፈላለግክ
43	kasfällägh	ካስፈለግህ
44	kasfällägk	ካስፈለግክ

Table 5.5 Output prior to the use of the algorithm given in figure 5.6

Even numbered words in the above table are wrongly generated due to the rule in Amharic that says the pattern */gk/* should be replaced by */kk/*. To account for such changes when two consonants come in succession, the following algorithm is developed.

Input: **Word**
Output: **Modified word**

Start
While there are two consecutive consonants in the word
Replace them with the appropriate form from the ConsonantChangeTable
Return Word
End

Figure 5.6 The Algorithm that changes two consecutive consonants

The following algorithm works in parallel with the above one to account for the changes due to palatalization.

Input: **Stem and Suffix**
Output: **Generated word**

Start
Lchar = the last character of the Stem
Fchar = the first character of the Suffix
Pattern = Lchar + Fchar
Set Word to Stem
Get the SurfaceForm for Pattern from the PalatalizationTable
If the SurfaceForm found
Word = stem without Lchar + SurfaceForm + Suffix without Fchar
Return Word
End

Figure 5.7 The Algorithm that checks palatalization

The output of the algorithms given in figures 5.6 and 5.7 for the input root name = flg suffix subject maker = 2psm is given here below.

1	alfälläghm	አልፈለግህም
2	alfälläkkm	አልፈለክም
3	fällägh	ፈለግህ
4	fälläkk	ፈለክ
5	asfällägh	አስፈለግህ
6	asfälläkk	አስፈለክ
7	affalägh	አፋለግህ
8	affaläkk	አፋለክ
9	täfallägh	ተፋለግህ
10	täfalläkk	ተፋለክ
11	fälallägh	ፈላለግህ
12	fälalläkk	ፈላለክ
13	täfälallägh	ተፈላለግህ
14	täfälalläkk	ተፈላለክ
15	affälallägh	አፈላለግህ
16	affälalläkk	አፈላለክ
17	käfällägh	ከፈለግህ
18	käfälläkk	ከፈለክ
19	käfälallägh	ከፈላለግህ
20	käfälalläkk	ከፈላለክ
21	släfällägh	ስለፈለግህ
22	släfälläkk	ስለፈለክ
23	alfälalläghm	አልፈላለግህም
24	alfälalläkkm	አልፈላለክም
25	täfällägh	ተፈለግህ
26	täfälläkk	ተፈለክ
27	slätäfällägh	ስለተፈለግህ
28	slätäfälläkk	ስለተፈለክ
29	slätäfallägh	ስለተፋለግህ
30	slätäfalläkk	ስለተፋለክ
31	slasfällägh	ስላስፈለግህ
32	slasfälläkk	ስላስፈለክ
33	slafällägh	ስላፈለግህ
34	slafälläkk	ስላፈለክ
35	slafallägh	ስላፋለግህ
36	slafalläkk	ስላፋለክ
37	kätäfälallägh	ከተፈላለግህ
38	ätäfälalläkk	ከተፈላለክ
39	kätäfällägh	ከተፈለግህ
40	kätäfälläkk	ከተፈለክ
41	kasfälallägh	ካስፈላለግህ
42	kasfälalläkk	ካስፈላለክ

43	kasfällägh	ካስፈለግህ
44	kasfälläkk	ካስፈለክ

Table 5.6 The output of the algorithms given in figures 5.6 and 5.7

The wrongly generated even numbered words are corrected using the implementation of algorithms 5.6 and 5.7 as per the evaluation of domain experts.

If the negative-marker rule is not used, some of the words generated from a given root and suffix will not give meaning. The input root name =skr and suffix subject marker = 3psm could be used to elaborate the above conjecture.

1	alsäkkärä	አልሰከረ
2	säkkärä	ሰከረ
3	säkakkärä	ሰካከረ
4	assäkkärä	አሰከረ
5	assakärä	አሳከረ
6	täsakkärä	ተሳከረ
7	täsäkakkärä	ተሰካከረ
8	assäkakkärä	አሰካከረ
9	käsäkkärä	ከሰከረ
10	käsäkakkärä	ከሰካከረ
11	släsäkkärä	ስለሰከረ
12	alsäkakkärä	አልሰካከረ
13	täsäkkärä	ተሰከረ
14	slätäsäkkärä	ስለተሰከረ
15	slätäsakkärä	ስለተሳከረ
16	slassäkkärä	ስላሰከረ
17	slasäkkärä	ስላሰከረ
18	kätäsäkakkärä	ከተሰካከረ
19	kätäsäkkärä	ከተሰከረ
20	kassäkakkärä	ካሰካከረ
21	kassäkkärä	ካሰከረ

Table 5.7 Output prior to the use of the algorithm given in figure 5.8

The words in number 1 and 12 of the above table do not convey meaning due to the failure of using the negative-marker rule. To accommodate the negative maker rule, the following algorithm is developed.

Input: **word**
Output: **modified word**

Start
If the first two characters of the word is equal to "al"
Add suffix 'm' at the end of the word
Return word
End

Figure 5.8 The Algorithm that applies negative-marker rule

The outputs of the algorithm given in figure 5.6 for the input root name = skr and suffix subject marker = 3psm is given here below.

1	alsäkkäräm	**አልሰከረም**
2	säkkärä	**ሰከረ**
3	säkakkärä	**ሰካከረ**
4	assäkkärä	**አሰከረ**
5	assakärä	**አሳከረ**
6	täsakkärä	**ተሳከረ**
7	täsäkakkärä	**ተሰካከረ**
8	assäkakkärä	**አሰካከረ**
9	käsäkkärä	**ከሰከረ**
10	käsäkakkärä	**ከሰካከረ**
11	släsäkkärä	**ስለሰከረ**
12	alsäkakkäräm	**አልሰካከረም**
13	täsäkkärä	**ተሰከረ**
14	slätäsäkkärä	**ስለተሰከረ**
15	slätäsakkärä	**ስለተሳከረ**
16	slassäkkärä	**ስላሰከረ**
17	slasäkkärä	**ስላሰከረ**
18	kätäsäkakkärä	**ከተሰካከረ**
19	kätäsäkkärä	**ከተሰከረ**
20	kassäkakkärä	**ካሰካከረ**
21	kassäkkärä	**ካሰከረ**

Table 5.8 Output after to the use of the algorithm given in figure 5.8

The wrongly generated words in numbers 1 and 12 are corrected through the implementation of the algorithm in figure 5.8.

5.3 The prototype

The algorithms presented earlier are coded using Microsoft Visual Basic 6.0. Figure 5.9 shows the main screen of AmharicMorphologicalSynthesizer.

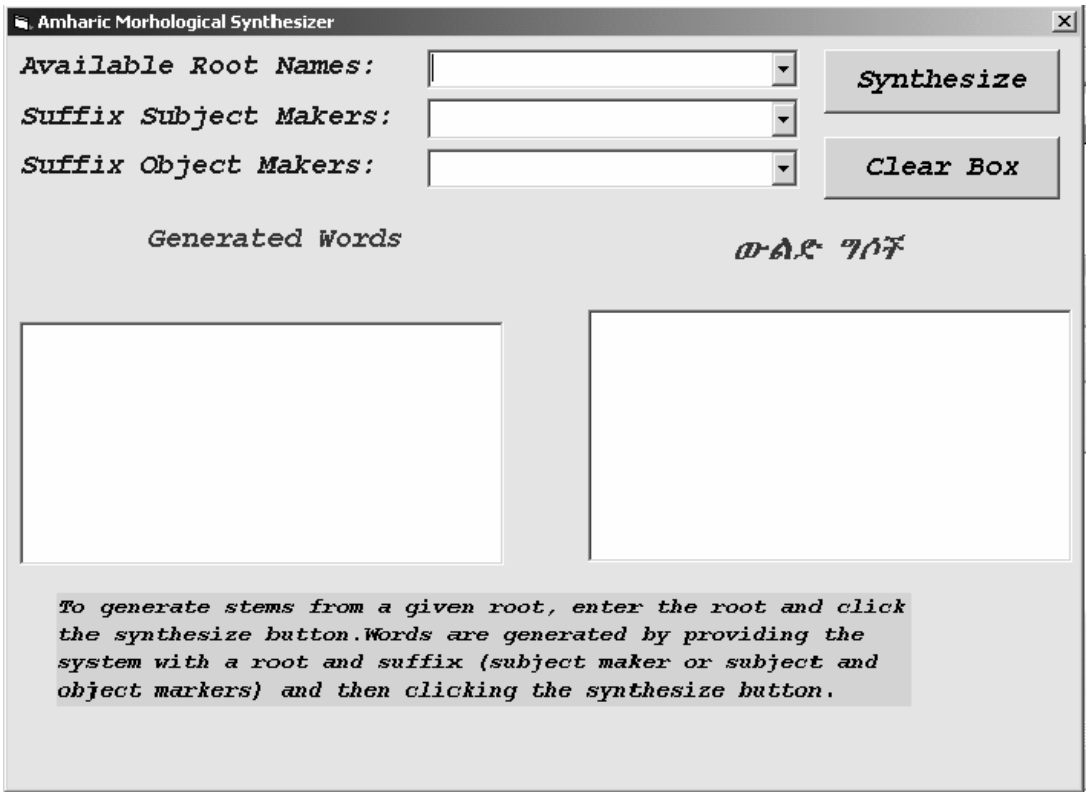


Figure 5.9 Main screen of the AmharicMorphologicalSynthesizer.

The required words are generated either by typing the appropriate root and suffixes directly on the spaces provided or selecting the appropriate root from the root combo box and the appropriate suffixe(s) either from the suffix subject marker or from both the suffix subject marker and the suffix object marker combo box(s)

When the synthesize button is pressed after providing all the necessary information, all the algorithms discussed above are applied and the words or stems generated are listed in the list box provided below *Generated Words* in phonetic symbols and the Amharic equivalent of each phonetic symbol in the space provided below **ውልድ**

ግሶች.

The prototype *AmharicMorphologicalSynthesizer* can be used to generate stems or words depending on the input. If the only input to the *AmharicMorphologicalSynthesizer* is root, stems will be generated as output. On the other hand, words are generated as output provided that the input to the system are root and suffix.

When the *Synthesize* button is pressed with out a root, an error message "You have to enter a root first" is displayed. If the suffix object marker is supplied without the suffix subject marker, an error message "You have to enter suffix subject maker first" is displayed

5.4 Testing Procedures

The following procedures were employed through the first phase of the experiment to evaluate the performance of the synthesizer.

1. To test the effectiveness of the algorithms developed, the roots *sbr* of type A, *flg* of type B, *mrk* of type C, *lkk* of type B and C, *lqq* of type A and C, *lsn* of type A and B, and *hlf* of type A were selected. The above roots have been selected by domain experts (linguists) based on the representativeness of their type. The representativeness of the selected roots is also supported by the sources given in the introduction section of chapter three. With these selected roots, the prototype verb synthesizer generates a total of 1428 words that is tasking on the part of the domain experts who evaluated the system.
2. Domain experts evaluated the words generated by the selected roots. With the aim of improving the accuracy of the *AmharicMorphologicalSynthesizer*, causes of errors 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. The accuracy of the system is calculated as the number of correctly generated words divided by the total number of words generated by the system multiplied by 100%.

Mathematically,

$$Accuracy = \frac{Number\_of\_correctly\_generated\_words}{Total\_number\_of\_words\_generated} * 100\,\%$$

5.5 Results of the experiment

The following table presents the results of the test based on the selected roots and suffixes by applying the above procedures.

Root Name	Root Type	Suffix Subject Maker	Suffix Object Maker	Number of Words Generated	Number of Correctly Generated words	Number of Wrongly Generated words	Accuracy
Sbr	A	3psm	No	21	21	0	100%
		3psf	No	21	21	0	100%
		2psm	No	42	42	0	100%
		2psf	No	21	21	0	100%
		1psc	No	42	42	0	100%
		3ppc	No	21	21	0	100%
		2ppc	No	21	21	0	100%
		1ppc	No	42	42	0	100%
		3psm	3psm	21	21	0	100%
		3psm	3psf	42	29	13	69.05%
		3psm	2psf	21	21	0	100%
		3psm	3ppc	42	42	0	100%
		2psf	3ppc	42	42	0	100%
Flg	B	3psm	No	22	22	0	100%
		3psf	No	22	22	0	100%
		2ppsm	No	44	44	0	100%
		2psf	No	22	22	0	100%
		1psc	No	44	44	0	100%
		3ppc	No	22	22	0	100%
		2ppc	No	22	22	0	100%
		1ppc	No	44	44	0	100%
		1ppc	2psf	22	22	0	100%
		2psf	3psm	22	22	0	100%
		3psm	No	19	15	4	78.95%
		3psf	No	19	14	5	73.68%
		2psm	No	38	22	16	57.90%
		2psf	No	19	13	6	69.42%

Root Name	Root Type	Suffix Subject Maker	Suffix Object Maker	Number of words Generated	Number of correctly Generated words	Number of wrongly Generated words	Accuracy
Mrk	C	1psc	No	38	24	14	63.16%
		3ppc	No	19	12	7	63.16%
		2ppc	No	19	10	9	52.63%
		1ppc	No	38	24	14	63.16%
		2psm	3ppc	38	28	10	73.68%
		1psc	2psf	19	13	6	68.42%
		1ppc	2ppc	19	13	6	68.42%
		3psm	3psm	38	19	19	50%
Lkk	B and C	2psf	No	4	4	0	100%
		2ppc	No	4	4	0	100%
		3psm	No	4	4	0	100%
		2psm	1psc	4	4	0	100%
		3psf	3ppc	8	8	0	100%
Lqq	A and C	2psf	No	10	10	0	100%
		2ppc	No	10	10	0	100%
		3psm	No	10	10	0	100%
		3psf	1psc	10	10	0	100%
		3psf	3ppc	20	20	0	100%
Lsn	A and B	3psf	No	43	43	0	100%
		2psm	No	86	86	0	100%
		2ppc	No	43	43	0	100%
		2psm	3ppc	86	86	0	100%
		2psm	1psc	43	43	0	100%
Hlf	A	2psm	No	42	42	0	100%
		3psf	No	21	21	0	100%
		2ppc	No	21	21	0	100%
		3ppc	1psc	21	21	0	100%
		2ppc	3ppc	42	42	0	100%

Table 5.9 A table that depicts the results of the test on the selected roots

5.6 Discussion of the results

5.6.1 Error Analysis

During the experiment the two linguists[25] who evaluate the outputs of the system almost in the same way identified the following errors. The errors are summarized as follows.

1. Two consonants should not come at the beginning of any Amharic word as in *slätäsäbbäräcc* (for she got broken) ***ስለተሰበረች***. The above error is not supported by Leslau (1973) on which much of the thesis was based. The suggestion given is to put ɨ in the middle of the two consonants. Thus the above word will be changed to sɨl*ätäsäbbäräcc*. To correct this error, the phonetic symbol ɨ was inserted in the appropriate templates. In the output, s?lasäbbärä is generated. All the errors in the roots *sbr* and *flg* are corrected by inserting the above ɨ in all the appropriate templates.

2. The second error is that the */c/* in *slätäsäbbäräcc* should be */č/*. This is reported to be a problem observed in some of the generations of an input root. Internally, */c/* is recognized as */č/*. Thus, this is not a problem, as it is understood correctly by the system.

3. Some of the templates in Type C verbs such as ***mrk*** were not correctly represented in the database. For instance, instead of putting the template as 1ä2a22ä3 it was represented as 1a2a22ä3 in *TemplateTable* of the

[25] The developed system is evaluated by Ato Daniel Aberra, an instructor in the Department of Linguistics at Addis Ababa University, and Dr. Yonas Admassu, an instructor in the Department of Literature at Addis Ababa University

MorphoSynthesisLexicon database. Due to the above misrepresentation the word märarräkäcc **መራረክች** is generated wrongly as mararräkäcc **ማራረክች(**which is meaningless). All the errors in type C root types such as *mrk* are fixed easily by modifying the appropriate templates.

4. **አልሰበርህም (**alsäbbärhm) (you did not break) and **አልሰበርክም** (alsäbbärkm) (you did not break) do have the same meaning. But the synthesizer generates them as different words. The results of suffixing the second person singular masculine (2psm) subject marker suffixes /k/ and /h/ to the same stem will generate words with the same meaning. In this case these 2psm suffixes ***k*** and ***h*** are considered to generate words with the same meaning, but in different dialects or regional variations. The same justification is given for the similar errors reported in the First Person Singular Constant (1psc) subject marker suffixes /hu/ and /ku/.

5. Some of the generated words were correct both grammatically and semantically, but not used in day-to-day conversation. An example of this is täsabbäru (they broke each other) **/ተሳበሩ/.**

6. Some of the verbs generated are grammatically correct but ambiguous in meaning. An example of this is täsabbäräw (he broke with him) **ተሳበረው.** But the linguists who evaluated the output of the system just argue unanimously that such words do have the potential to be added into the vocabulary of Amharic.

The errors in 1 and 3 above are corrected by making some modifications on the database except the second one that persisted throughout the experiment. The errors

reported in 4, 5 and 6 above are not actual errors in the perspective of this study since the principal objective of the study is not on semantics.

5.7 Root Type Prediction Using Neural networks

The second phase of the experiment uses the neural network software *BrainMaker* for the purpose of predicting the type of the root not available in the database. The flowchart of the training procedures used by Berhanu (1999) and also in this study is given below.

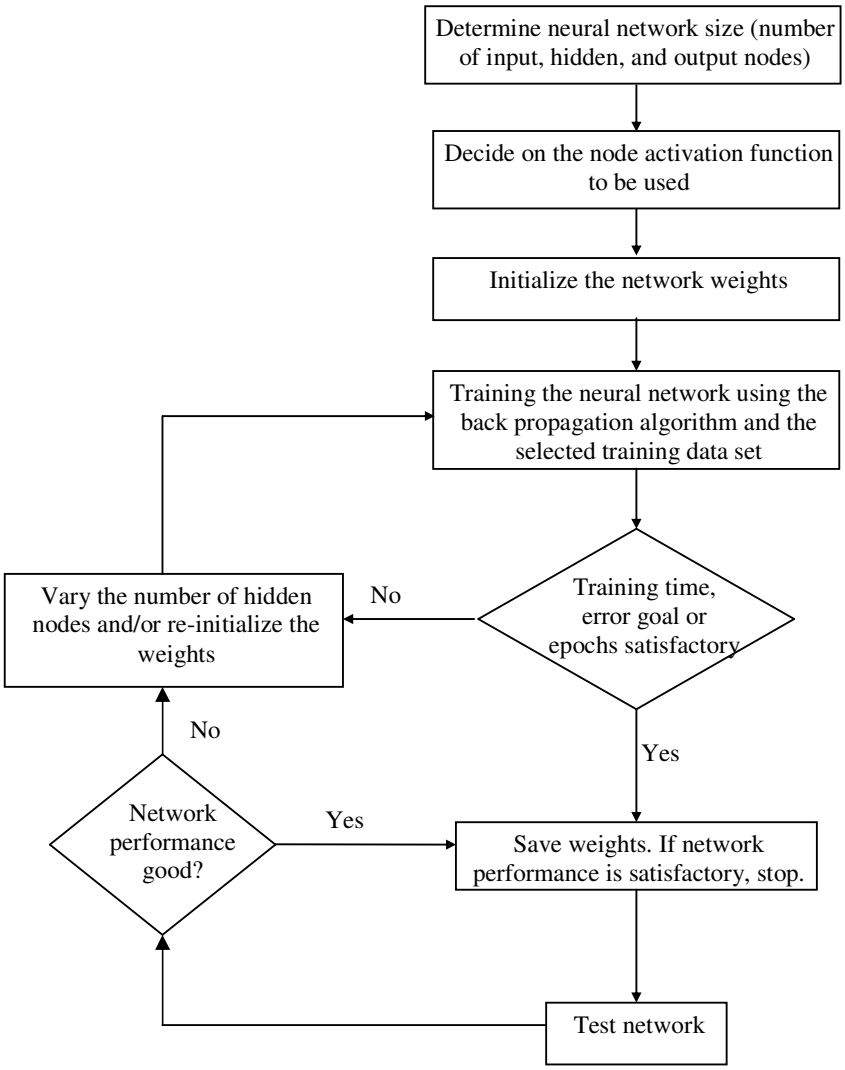


Figure 5.10 Neural Network Training Procedures

The numbers of input and output nodes were determined as 64 and 3 respectively. Initially, training was started based on the default network parameters. The default parameters for the neural network are: learning rate=1.0; neuron function= sigmoid; training tolerance=0.100; testing tolerance=0.400; training noise=0.00; number of hidden layers=1; number of hidden nodes= 64; network weights=small randomly chosen numbers, and number of test data= 10% of the data set and gave the following network progress.

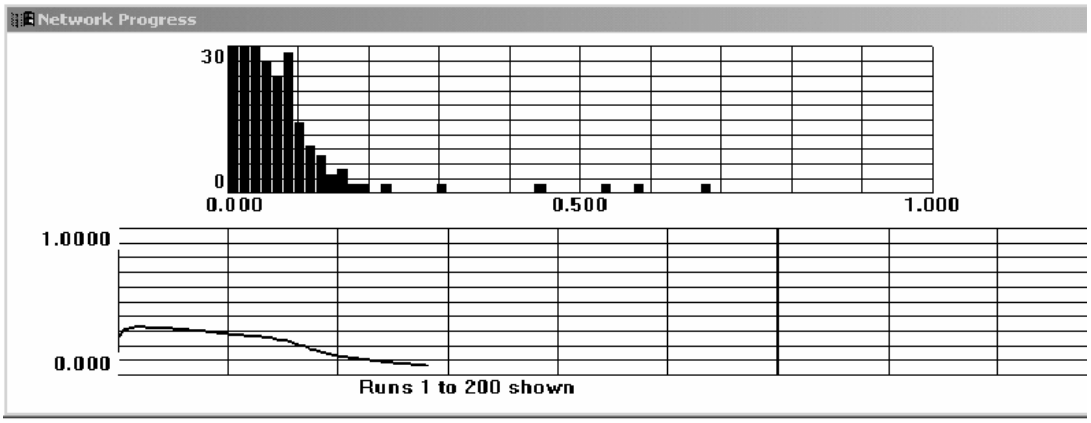


Figure 5.11 Progress of the neural network on the training data set using the default values.

The above model constructed using the default values predicts the type of 21 of the 27 roots correctly and 6 of the 27 roots wrongly. Thus, the model works with an accuracy of 77.78%. The above network has an average error and root mean square (RMS) error of 0.1378 and 0.3346 respectively.

After iterative testing of the input of the neural network by changing network parameters, about 12 models are selected with better predictive power. Then these 12 models are further tested using the running fact. The model having good prediction capability is then selected. The model had the following network parameter changes over the default settings: Learning rate= 0.400; Connection noise=0.2002;

With the above changes, the network progresses as follows.

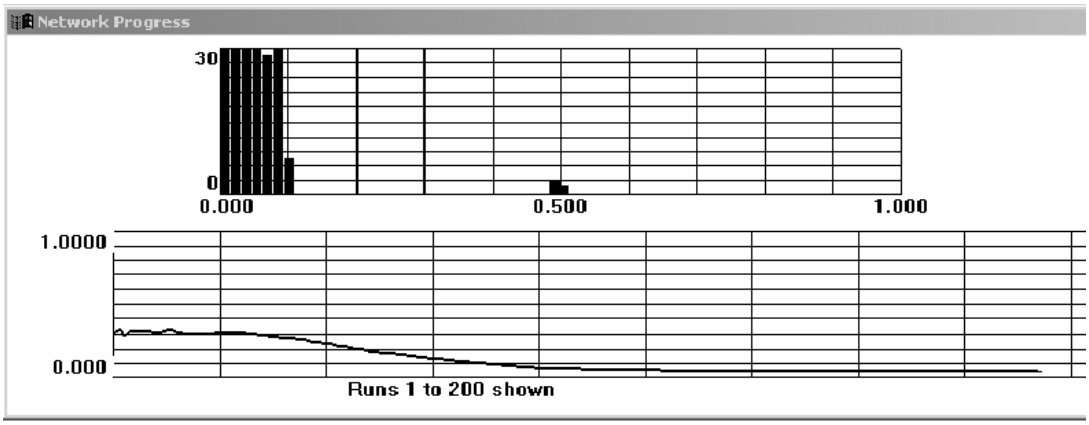


Figure 5.12 Progress of the selected model on the training data set.

The above selected model predicts the type of 22 of the 27 roots correctly and 5 of the 27 roots wrongly. Thus, the model works with a precision of 81.48. The above network has an average error and RMS error of 0.1325 and 0.3314 respectively.

The selected model is tested with 14 new roots in the absence of their type and the following results are obtained.

Actual Root Type	Predicted Root Type				Total
		A	B	C	
	A	4	1	0	5
	B	3	1	0	4
	C	0	0	5	5
Total		7	2	5	14

Table 5.10 Prediction Results of the neural network

As it can be seen from the above confusion matrix, the model wrongly predicted one type A verb form as type B and three type B verb forms as type A The model predicts all type C verb forms correctly.

In the case of perfective verb forms, root types A and B are similar in that they have the same templates. Thus, predicting type A verb forms as type B and vice versa will not practically be taken as a wrong prediction. This problem will be alleviated, hopefully, if all the verbal forms are considered.

5.8 Summary

In this chapter, detailed discussions were made on the algorithms designed. Based on the algorithms, a prototype was developed using Visual Basic with an interface easy enough to be used by individuals. An experiment was also conducted using the prototype on the roots selected. The results achieved and the discussions made out of the experiment are also reported in this chapter. From what has been observed from the discussion in this chapter, the prototype developed using the rule-based approach has high accuracy close to 100%. The neural network is identified to predict with an accuracy of 81.48%. Hopefully, the accuracy will increase if it is tested with a large data. The next chapter will conclude this thesis by diving conclusions and indicating future areas of research.

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusion

The purpose of this study was to design a word synthesizer for Amharic perfective verb forms. The Morphological properties of Amharic in general and that of perfective verb forms in particular that are useful for computer representation are identified. Then the various techniques to morphological synthesis are reviewed. Accordingly, an approach that combines rule-based (CV-based approach and TLM: selected because they take the properties of the language into account) and neural network (used to predict the type of the root not available in the RootTable table) is employed for this study. A database of templates, roots, suffixes and phonological variations (palatalization, consonant change and vowel change), which helps the synthesizer to generate different words from an input, are designed. Training data sets and running facts are also prepared to be used as an input to the BrainMaker. Algorithms are designed from scratch as there are no previously designed algorithms for this purpose based on the morphological properties of the language to generate perfective verb forms from an input root and suffix (subject marker or subject and object markers). Finally, a prototype word synthesizer is developed to evaluate the performance of the designed algorithms.

It has been found throughout the research that Amharic is a morphologically complex language. Most of the words in the language are verbs or derived from verbs. It is indicated that verbs are the most productive classes of words in the language. The root is a basic lexical unit of the language. The root form of a base verb can be bilateral, trilateral, quadrilateral and quinitiliteral.

Verbs are derived from templates by inserting consonant patterns to the template. Therefore, most of the words in the language can be reduced to template and root patterns. In this study, the vowels are inserted into the template due to the reasons given in section 4.3 of chapter three. Roots form their stems based on their type. It is also shown that word formation follows regular patterns: prefixing, suffixing, circumfixing and infixing. Thus it is possible to conclude that, even though Amharic is morphologically complex its regular patterns make it manageable for algorithmic processing.

The records in all the tables except in RootTable and NuralRootTable5 tables are exhaustive. It is practically impossible to list all the roots in the RootTable table. To account for the remaining roots, this study uses neural network as a technique to predict the type of the root not included in the *RootTable* table.

In order to test the accuracy of the algorithms developed the root *sbr* of type A, the root *flg* of type B, the root *mrk* of type C, the root *lkk* of type B and C, the root *lqq* of type A and C, the root *lsn* of type A and B and the root *hlf* of type A were considered. The test results have been presented in the previous chapter. The errors encountered during experimentation have also been corrected. The neural network model predicts the type of 27 (10% of all the roots which constitute the test data set) of the 273 roots available in *NuralRootTable5* table with an accuracy of 81.48%.

Even though the accuracy of the synthesizer developed in this study is somewhat acceptable, it may not have an immediate practical application for the synthesizer was not trained on large quantities of data for various reasons. The reasons for not being able to train the synthesizer on such data are:

1. Lack of large corpora in the language annotated with roots with their type.
2. Manually determining the type of the root 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 a hybrid of rule-based (CV-based and TLM of morphology) and neural network to develop an automatic synthesizer to Amharic.

Apart from such limitations, the research has indicated the possibility to develop an automatic synthesizer for Amharic. That is, it appears to be feasible to develop an efficient word synthesizer for Amharic provided that enough training data is available.

The module *ConvertToAmharic*, that converts phonetic symbols to their Amharic equivalent, could be used whenever needed. This module can also accomplish the reverse process with minor modification.

It is also anticipated that this first work in automatic word synthesis will encourage Ethiopian students and researchers to take part in similar researches 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.

6.3 Recommendations

1. The algorithms are tested on a limited number of roots and the accuracy reported is valid only for this test set. Therefore, other tests using a large number of data must be conducted so as to make it possible to estimate the accuracy of the synthesizer.

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

3. This study considers only perfective verb forms. Enhancing the system to include all the other verb forms is also an interest in future works.

4. Develop synthesizers for Amharic and other local languages, using other approaches (e.g. purely stochastic or purely rule-based), and compare the results obtained with that of the approach followed in this thesis.

5. Launching big project to develop an efficient full-fledged automatic morphological synthesizer for Amharic.

6. Organizing an NLP team that will be responsible to carry out such huge project in particular, and researches in NLP in general may be necessary. As NLP requires expertise from different fields of study, the team for such a project should consist of at least linguists, programmers and information professionals.

7. Word synthesizers are not an end by themselves; they are ways to an end. Therefore, further studies should also focus on implementing application that can use the outputs of word synthesizers such as machine translation, spell-check, speech recognition, dictionary (lexicon) compilation, POS tagging, morphological analysis, automatic sentence construction, etc

8. As derived nouns undergo similar inflections as regular nouns, the algorithms can be extended easily for nouns by designing a lexicon that can handle regular nouns.

9. Amharic has a handful of prepositions and regular adverbs. The synthesizer can be extended easily to include these categories.

10. Test the neural network with large data to see its potentiality for applying it in morphological synthesis.

Bibliography

Abiyot Bayue. 2000. Developing Automatic Word Parser for Amharic Verbs and their Derivation, A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfillment of the Degree of Masters of Science in Information Science.

Alemayehu Haile. 1987. 'An Autosegmental Approach to Amharic Intonation'. PHD. Dissertation at the School of Oriental and African Studies University of London.

Allen, J. 1996. *Natural Language Understanding*. 2nd Ed. California:Redwood, Benjamin/Cummings Publishing Company, Inc.

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

Anderson, D. and McNeil, G. 1992. Artificial Neural Networks Technology. Available at http://www.dacs.dtic.mil/techs/neural/neural.title.html

Anderson, S.R. 1988. 'Morphology as a Parsing Problem'. *Linguistics* (26): 521 – 544.

Anshen, F. and Mark, A. (1988). 'Producing Morphologically Complex words'. *Linguistics* 26(1988), 641-655.

Antworth, E. 1994. *Morphological Parsing with a Unification Based Word Grammar.* North Texas Natural Language Processing Workshop, University of Texas, 1994. Available at http://www.sil.org/pckimmo/ntn1p94.html.

Antworth, E.L. 1991. *Introduction to Two-Level Phonology Linguistics* http://www.sil.org/pckimmo/two_level_phon.html.

Aronof, M. 1976. *Word Formation In Generative Grammar*. MIT:MA.

Artrandi, S. 1976 Machine Indexing: Linguistics and Semiotic Implication. *JASIS*, 27(4) 235-239.

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

Baye Yimam .1998.. Root Reductions and Extensions in Amharic. *Ethiopian Journal of Languages and Literature.* No. 9: 56-88.Baye Yimam.1988.

Baye Yimam. 1999. Towards a definition of the Nominal Specified in Amharic. In *Proceedings of the 8th International Conference of Ethiopian* Studies Vol. 1. Edited by Tadesse Beyene. Addis Ababa, Ethiopia, 599-612.

Bender, L. and Hailu Fullass. 1978. *Amharic Verb Morphology*. Michigan State University.

Bender, M. and Ferguson, C. 1976. *Introduction. Language in Ethiopia***;** edited by Bender, M. and Ferguson, C. [*et al*]; Oxford :London.

Berhanu Aderaw. 1999. Amharic Character Recognition Using Artificial Neural Networks. A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfilment of the Degree of Masters of Science in Electrical Engineering.

Birru Dori.1992. Customizing CDSIS/IS for Amharic Texts. A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfilment of the Degree of Masters of Science in Information Science.

Birungi, P. 1995**.** Improved Strategies for Employment and Human Resources Utilization in the Information and Documentation Sector. In Strategies for Human Resources Development for information Management in Africa, Ed. Francis Inganji, 49-57. Addis Ababa: UNECA, PADIS.

Bybee, J. 1985. Morphology: *a Study of the Relation between meaning and form.* Benyamins: Amsterdam.

Culy, C. 1985. The Complexity of the Vocabulary of Bambara. *In The Formal Complexity of Natural Language* , Edited by Walter J. [et al]; Redial: Boston 349-357.

Dawkins, C. 1969. Fundamentals of Amharic, Addis Ababa Sudan interior mission.

Dereje Teferi .1999. Optical Character Recognition of Type written Amharic Texts. A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfilment of the Degree of Masters of Science in Information Science.

Ermias Abebe. 1998. Application of OCR Techniques for Recognition of Formatted Amharic Texts, A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfillment of the Degree of Masters of Science in Information Science.

Ethiopian Central Statistical Authority (ECSA). 1998. The 1994 Population and Housing Census of Ethiopia: Results at Country Level. Vol. 1 Statistical Report[26]. Addis Ababa.

Gazdar ,G. and G. Pullum.1985.*Computationally Relevant Properties of Natural Language and Their Grammars.* IN the Formal Complexity of Natural Language , edited by Walter J. [et al]; Redial: Boston 387-438.

Gazdar, G. (1996). Paradigm Merger in Natural Language Processing. *Computing Tomorrow: Future research Directions in Computer Science.* Edited by Ian Ward and Robin Milner. Cambridge: NY. PP 88-109.

[26] The latest official statistics available

Getahun Amare.1989. Zämänäwi yäamarðňňa säwäsäw bäqäläl aqäräräb. Commercial Printing Press Addis Ababa.

Girma Halafom. 1994. *The Syntax of Functional Categories: a Study of Amharic*. PHD. Thesis in Linguistic at the Université du Que'bec à Montreal.

Girmay Berhane. 1992. "Word formation in Amharic". Journal of Ethiopian Languages and Literature. No.2, 50-74.

Golding, A. and H. Thompson. 1985. A Morphological Component for Language Programs. *Linguistics,* 23: 263-289.

Gravin, P. 1963. *Natural Language and the Computer*. McGraw-Hill: New York.

Grishman, R. 1984. Natural Language Processing. *JASIS 35*(5):291-296.

Grishman, R. 1986. *The Theory of Linguistic an Introduction.*

Güngördü, Z. and Oflazer, K. 1995. Parsing Turkish Using Lexical Functional Grammar Formalism. In *Machine Translation* 10: 293-319.

Habte Mariam Marcos. 1994. "Towards the Identification of the Morphemic Component of the Conjugation Forms of Amharic". In Proceedings of *The Eleventh International Conference of Ethiopian Studies*. Vol. I pp. 465-486.

Hailu Fullass. 1966. *Derived Nominal Patterns in Amharic* . PHD. Dissertation at the University of California, Los Angeles.

Hirut Woldemariam. 1989. "An Auto-segmental Approach to Geez-Based Amharic Plural Nouns" Ethiopian Journal of Languages and Literature. Addis Ababa. 69-92.

Jirku, P. 1984. *Logical and Linguistic Aspects of Computer-Based Inference Processing.*

Karttunen, L. 1983. Constructing Lexical Transducers. In the Proceedings of 15th International Conference of Computational Linguistics. *COLING*-94 pp. 406 – 411.

Katamba, F. 1993. Morphology. London: Macmillan.

Kazakov, Dimater & Manandhar, Suresh. 2000. *Unsupervised Learning for Word Segmentation Rules with Genetic Algorithms and Inductive Logic Programming.* Available at http://citeseer.nj.nec.com/kuzakov00unsupervised.html.

Koskenniemi, Kimmo. 1983. Two-Level Morphology: A general Computational Model for Word-form recognition and production. *Publication 11,* University of Helsinki, Department of General Linguistics, Helsinki.

Kuznets, S..1966. Modern Economic Growth: Rate, Structure, Spread. New Haven: Yale University Press.

Leslau, W. 1965. *An Amharic Text Book of Everyday Usage*. University of California, Los Angles.

Leslau, W. 1967. *Amharic Text Book*, Otto Harrassowitz, Wiesbaden, Belgium.

Leslau, W. 1973. English-Amharic Context Dictionary. Wiesbaden: Harrasowitz.

Mao, Y. .1997. "Natural Language Processing Module (Part of Speech Tagging and Sentence Parsing) Laboratory Manual" Available at http://www.csic.cornell.edu/201/natural_language/.

McCarthy, J. 1981. A Prosodic Theory of Non Concatenative Morphology. *Linguistic* Inquiry. Vol. 12 (3), 373-417.

Mesifin Getachew. 2001. Automatic Part of Speech Tagging for Amharic Language: An Experiment Using Stochastic Hidden Markov (HMM) Approach, A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfillment of the Degree of Masters of Science in Information Science.

Million Meshesha. 2000. A Generalized Approach to OCR of Amharic Texts, A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfillment of the Degree of Masters of Science in Information Science.

Mullen, D. 1986. *Issues in the Morphology and Phonology of Amharic: The Lexical Generation of Pronominal clitics*. A Dissertation for the Degree of Doctor of Philosophy at the School of Graduate Studies and Research, Department of Linguistic; University of Ottawa.

Nega Alemayehu. 1999. Development of a Stemming Algorithm for Amharic Texts Retrieval. PHD. Dissertation at University of Shifeld.

Pacak, M. 1967. Computational Morphology. *In Papers presented in honour of Levin Dosant***. E**dited by Wiliam M. Austin. The Hague Mouton.

Palmer, Martha and Zhibio, Wu. 1995. Verb Semantics for English-Chinese Translation. In *Machine Translation.* Vol.10: 59-92.

Pinker, S. 1999. Words and rules. *Lingua* , v.106 , 219-242.

Polacek, Z. 1989. Functional Sentence Perspective in Amharic: Preliminary notes. *8th International Ethiopian Studies.* 561-566.

Polak, M. 1967. Computational Morphology. *In Papers presented in honour of Levin Dosant***;** edited by Wiliam M. Austin. The Hague Mouton.

Prasad, A. 1993. *Application of Compute- Based Natural Language Processing Tools and Technique in Developing Subject Indexing Language*. PhD Dissertation at Karanatak University, Dhaward, Department of Library and Information

Science.

Pullman, s.; Russel, G.; Ritchie, G. and Black, A. 1988. *Computational Morphology of English*. *Linguistic* 26, 545 – 560.

Saba Amsalu. 2001. Application of Retrieval Techniques for Amharic Documents on the Web, A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfillment of the Degree of Masters of Science in Information Science.

Salton, G. 1989. *Automatic Text processing: The Transformation, Analysis, and Retrieval of Information by Computer*; Wsely: Massachusetts.

Salton, G.1983. Natural Language Processing. Introduction to Modern Information Retrieva*l*. New York: McGraw-Hill.

Shiber ,S. 1985. Separating Linguistic Analysis from Linguistic Theories. In *Natural Language Parsing and Linguistics Theories*, Edited by U. Reyle and C. Rohrer. Redial:Dordecht V. 35 , 33-68.

Sparc Jones, K. 1974. Progress in Documentation Automatic Indexing. *Journal of Documentation,* 30(4) 393-432.

Taylor, G. 1995. *Neural Networks* .Great Britain, Suffolk: Lavenham.

Thompson, F and Thompson, B. 1975. Practical Natural Language: the REL System as prototype. *Advance in Computers*, 13 110-169.

Trost, H. 2000. *Computational Morphology.* Available at http://www.univie.ac.at/~harald/handbook.html.

Uibo, H. 2001. On Using the Two-Level Model as the Basis of Morphological Analysis and Synthesis of Estonian. Available at http://www.ut.ee.html.

UNESCO. 1976. Moving Towards Change. PADIS, UNESCO.

Warner, A. 1987. Natural Language Processing. *ARIST,* Vol. 22, 1987 79-108.

Wedekind, K. 1996. Which Form Predict all other Best? Variation on the Amharic Verb "Theme". *Journal of Ethiopia Studies* Vol.25: 73-92.

Williams, P. 1981. On the notion of "Lexical Related' and "Head of a word" *Linguistic Inquiry 12*(2): 245 – 274.

Worku Alemu. 1997. The application of OCR Techniques to the Amharic Script; A Thesis Submitted to Graduate Studies of Addis Ababa University in Partial Fulfillment of the Degree of Masters of Science in Information Science.

APPENDIX 1

THE AMHARIC CHARACTER SET (Leslau, 1967)

Order							Labialized				
1st	2nd	3rd	4th	5th	6th	7th					
ሀ ha	ሁ hu	ሂ hi	ሃ ha	ሄ he	ህ h	ሆ ho					
ለ lä	ሉ lu	ሊ li	ላ la	ሌ le	ል l	ሎ lo			ሏ $l^{w}a$		
ሐ ha	ሑ hu	ሒ hi	ሓ ha	ሔ he	ሕ h	ሖ ho					
መ mä	ሙ mu	ሚ mi	ማ ma	ሜ me	ም m	ሞ mo			ሟ $m^{w}a$		
ሠ sä	ሡ su	ሢ si	ሣ sa	ሤ se	ሥ s	ሦ so					
ረ rä	ሩ ru	ሪ ri	ራ ra	ሬ re	ር r	ሮ ro			ሯ $r^{w}a$		
ሰ sä	ሱ su	ሲ si	ሳ sa	ሴ se	ስ s	ሶ so			ሷ $s^{w}a$		
ሸ šä	ሹ šu	ሺ ši	ሻ ša	ሼ še	ሽ š	ሾ šo			ሿ $š^{w}a$		
ቀ qä	ቁ qu	ቂ qi	ቃ qa	ቄ qe	ቅ q	ቆ qo	ቈ $q^{w}ä$	ቍ $q^{w}i$	ቋ $q^{w}a$	ቌ $q^{w}e$	ቍ $q^{w}ə$
በ bä	ቡ bu	ቢ bi	ባ ba	ቤ be	ብ b	ቦ bo			ቧ $b^{w}a$		
ተ tä	ቱ tu	ቲ ti	ታ ta	ቴ te	ት t	ቶ to			ቷ $t^{w}a$		
ቸ čä	ቹču	ቺ či	ቻ ča	ቼče	ች č	ቾčo			ቿ $č^{w}a$		
ኀ hä	ኁ hu	ኂ hi	ኃ ha	ኄ he	ኅ h	ኆ ho	ኈ $h^{w}ä$	ኊ $h^{w}i$	ኋ $h^{w}a$	ኌ $h^{w}e$	ኍ $h^{w}ə$
ነ nä	ኑ nu	ኒ ni	ና na	ኔ ne	ን n	ኖ no			ኗ $n^{w}a$		
ኘ ňä	ኙ ňu	ኚ ňi	ኛ ňa	ኜ ňe	ኝ ň	ኞ ňo			ኟ $ň^{w}a$		
አ a	ኡ u	ኢ i	ኣ a	ኤ e	እ ə	ኦ o					
ወ wä	ዉ wu	ዊ wi	ዋ wa	ዌ we	ው w	ዎ wo					
ዐ a	ዑ u	ዒ i	ዓ a	ዔ e	ዕ ə	ዖ o					
ከ kä	ኩku	ኪ ki	ካ ka	ኬ ke	ክ k	ኮ ko	ኰ $k^{w}ä$	ኲ $k^{w}i$	ኳ $k^{w}a$	ኼ $k^{w}e$	ኵ $k^{w}ə$
ኸ hä	ኹhu	ኺhi	ኻha	ኼhe	ኽ h	ኾ ho					
ዘ zä	ዙ zu	ዚ zi	ዛ za	ዜ ze	ዝ z	ዞ zo			ዟ $z^{w}a$		
ዠžä	ዡžu	ዢži	ዣža	ዤže	ዥ ž	ዦ žo					
የ yä	ዩ yu	ዪyi	ያ ya	ዬ ye	ይ y	ዮ yo					
ገ gä	ጉ gu	ጊ gi	ጋ ga	ጌ ge	ግ g	ጎ go	ጐ $g^{w}ä$	ጒ $g^{w}i$	ጓ $g^{w}a$	ጔ $g^{w}e$	ጕ $g^{w}ə$
ጘ dä	ጙdu	ጚ di	ጓ da	ጜde	ጝ d	ጞ do			ጟ $d^{w}a$		
ጠ ṭä	ጡṭu	ጢṭi	ጣṭa	ጤṭe	ጥ ṭ	ጦ ṭo			ጧ $ṭ^{w}a$		
ጨćä	ጩću	ጪći	ጫća	ጬće	ጭ ć	ጮćo			ጯ $ć^{w}a$		
ጸ şä	ጹ şu	ጺ şi	ጻ şa	ጼ şe	ጽ ş	ጾ şo			ጿ $ş^{w}a$		
ፀ şä	ፁ şu	ፂ şi	ፃ şa	ፄ şe	ፅ ş	ፆ şo					
ፈ pä	ፉ pu	ፊ pi	ፋ pa	ፌ pe	ፍ p	ፎ po					
ፈ fä	ፉ fu	ፊ fi	ፋ fa	ፌ fe	ፍ f	ፎ fo			ፏ $f^{w}a$		
ፐ pä	ፑ pu	ፒ pi	ፓ pa	ፔ pe	ፕ p	ፖ po					
ⷝ vä	ⷞ vu	⷟ vi	ⷠ va	ⷡ ve	ⷢ v	ⷣ vo					

APPENDIX 2

SAMPLE TEST RESULTS

INPUTS

ROOT NAME =sbr
SUFFIX SUBJECT MAKER =3psm
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

1)	alsäbbäräm	**አልሰበረም**
2)	säbbärä	**ሰበረ**
3)	säbabbärä	**ሰባበረ**
4)	assäbbärä	**አሰበረ**
5)	assabärä	**አሳበረ**
6)	täsabbärä	**ተሳበረ**
7)	täsäbabbärä	**ተሰባበረ**
8)	assäbabbärä	**አሰባበረ**
9)	käsäbbärä	**ከሰበረ**
10)	käsäbabbärä	**ከሰባበረ**
11)	släsäbbärä	**ስለሰበረ**
12)	alsäbabbäräm	**አልሰባበረም**
13)	täsäbbärä	**ተሰበረ**
14)	slätäsäbbärä	**ስለተሰበረ**
15)	slätäsabbärä	**ስለተሳበረ**
16)	slassäbbärä	**ስላሰበረ**
17)	slasäbbärä	**ስላሰበረ**
18)	kätäsäbabbärä	**ከተሰባበረ**
19)	kätäsäbbärä	**ከተሰበረ**
20)	kassäbabbärä	**ካሰባበረ**
21)	kassäbbärä	**ካሰበረ**

INPUTS

ROOT NAME =flg
SUFFIX SUBJECT MAKER =3psm
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

1)	alfällägäm	**አልፈለገም**
2)	fällägä	**ፈለገ**
3)	asfällägä	**አስፈለገ**
4)	affaläga	**አፋለገ**
5)	täfallägä	**ተፋለገ**
6)	fälallägä	**ፈላለገ**
7)	täfälallägä	**ተፈላለገ**
8)	affälallägä	**አፈላለገ**
9)	käfällägä	**ከፈለገ**
10)	käfälallägä	**ከፈላለገ**
11)	släfällägä	**ስለፈለገ**
12)	alfälallägäm	**አልፈላለገም**
13)	täfällägä	**ተፈለገ**
14)	slätäfällägä	**ስለተፈለገ**
15)	slätäfallägä	**ስለተፋለገ**
16)	slasfällägä	**ስላስፈለገ**
17)	slafällägä	**ስላፈለገ**
18)	slafallägä	**ስላፋለገ**
19)	kätäfälallägä	**ከተፈላለገ**
20)	kätäfällägä	**ከተፈለገ**
21)	kasfälallägä	**ካስፈላለገ**
22)	kasfällägä	**ካስፈለገ**

INPUTS

ROOT NAME =mrk
SUFFIX SUBJECT MAKER =3psm
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

---

1)	almarräkäm	**አልማረከም**
2)	marräkä	**ማረከ**
3)	tämarräkä	**ተማረከ**
4)	asmarräkä	**አስማረከ**
5)	ammaräkä	**አማረከ**
6)	mararräkä	**ማራረከ**
7)	ammararräkä	**አማራረከ**
8)	kämarräkä	**ከማረከ**
9)	kämararräkä	**ከማራረከ**
10)	slämarräkä	**ስለማረከ**
11)	almararräkäm	**አልማራረከም**
12)	tämararräkä	**ተማራረከ**
13)	slätämarräkä	**ስለተማረከ**
14)	slasmarräkä	**ስላስማረከ**
15)	slamarräkä	**ስላማረከ**
16)	kätämararräkä	**ከተማራረከ**
17)	kätämarräkä	**ከተማረከ**
18)	kasmararräkä	**ካስማራረከ**
19)	kasmarräkä	**ካስማረከ**

INPUTS

ROOT NAME =lkk
SUFFIX SUBJECT MAKER =2psf
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

---

1)	lakkš	**ላክሽ**
2)	lakkäkš	**ላከክሽ**
3)	lällakš	**ለላክሽ**
4)	läkkäkš	**ለከክሽ**

INPUTS

ROOT NAME =lqq
SUFFIX SUBJECT MAKER =2psf
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

---

1)	laqqš	**ላቅሽ**
2)	laqqäqš	**ላቀቅሽ**
3)	läqqäqš	**ለቀቅሽ**
4)	laqqš	**ላቅሽ**
5)	tälaqqäqš	**ተላቀቅሽ**
6)	asläqqäqš	**አስለቀቅሽ**
7)	aläqqäqšm	**አለቀቅሽም**
8)	slätäläqqäqš	**ስለተለቀቅሽ**
9)	slasläqqäqš	**ስላስለቀቅሽ**
10)	slaläqqäqš	**ስላለቀቅሽ**

INPUTS

ROOT NAME =lsn
SUFFIX SUBJECT MAKER =2ppc
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

1)	allässänaccďhum	**አለሰናችሁም**
2)	lässänaccďhu	**ለሰናችሁ**
3)	läsassänaccďhu	**ለሳሰናችሁ**
4)	aslässänaccďhu	**አስለሰናችሁ**
5)	allasänaccďhum	**አላሰናችሁም**
6)	tälassänaccďhu	**ተላሰናችሁ**
7)	täläsassänaccďhu	**ተለሳሰናችሁ**
8)	alläsassänaccďhum	**አለሳሰናችሁም**
9)	kälässänaccďhu	**ከለሰናችሁ**
10)	käläsassänaccďhu	**ከለሳሰናችሁ**
11)	slälässänaccďhu	**ስለለሰናችሁ**
12)	alläsassänaccďhum	**አለሳሰናችሁም**
13)	tälässänaccďhu	**ተለሰናችሁ**
14)	slätälässänaccďhu	**ስለተለሰናችሁ**
15)	slätälassänaccďhu	**ስለተላሰናችሁ**
16)	slaslässänaccďhu	**ስላስለሰናችሁ**
17)	slalässänaccďhu	**ስላለሰናችሁ**
18)	kätäläsassänaccďhu	**ከተለሳሰናችሁ**
19)	kätälässänaccďhu	**ከተለሰናችሁ**
20)	kasläsassänaccďhu	**ካስለሳሰናችሁ**
21)	kaslässänaccďhu	**ካስለሰናችሁ**
22)	allässänaccďhum	**አለሰናችሁም**
23)	lässänaccďhu	**ለሰናችሁ**
24)	aslässänaccďhu	**አስለሰናችሁ**
25)	allasänaccďhum	**አላሰናችሁም**
26)	tälassänaccďhu	**ተላሰናችሁ**
27)	läsassänaccďhu	**ለሳሰናችሁ**
28)	täläsassänaccďhu	**ተለሳሰናችሁ**
29)	alläsassänaccďhum	**አለሳሰናችሁም**
30)	kälässänaccďhu	**ከለሰናችሁ**
31)	käläsassänaccďhu	**ከለሳሰናችሁ**
32)	slälässänaccďhu	**ስለለሰናችሁ**
33)	alläsassänaccďhum	**አለሳሰናችሁም**
34)	tälässänaccďhu	**ተለሰናችሁ**
35)	slätälässänaccďhu	**ስለተለሰናችሁ**
36)	slätälassänaccďhu	**ስለተላሰናችሁ**
37)	slaslässänaccďhu	**ስላስለሰናችሁ**
38)	slalässänaccďhu	**ስላለሰናችሁ**
39)	slalassänaccďhu	**ስላላሰናችሁ**
40)	kätäläsassänaccďhu	**ከተለሳሰናችሁ**
41)	kätälässänaccďhu	**ከተለሰናችሁ**
42)	kasläsassänaccďhu	**ካስለሳሰናችሁ**
43)	kaslässänaccďhu	**ካስለሰናችሁ**

INPUTS

ROOT NAME =hlf
SUFFIX SUBJECT MAKER =2psm
SUFFIX OBJECT MAKER =

GENERATED WORDS/STEMS

---

1)	alalläfhm	**አላለፍህም**
2)	alalläfkm	**አላለፍክም**
3)	alläfh	**አለፍህ**
4)	alläfk	**አለፍክ**
5)	alalläfh	**አላለፍህ**
6)	alalläfk	**አላለፍክ**
7)	asalläfh	**አሳለፍህ**
8)	asalläfk	**አሳለፍክ**
9)	aläfh	**አለፍህ**
10)	aläfk	**አለፍክ**
11)	talläfh	**ታለፍህ**
12)	talläfk	**ታለፍክ**
13)	talalläfh	**ታላለፍህ**
14)	talalläfk	**ታላለፍክ**
15)	alalläfh	**አላለፍህ**
16)	alalläfk	**አላለፍክ**
17)	kalläfh	**ካለፍህ**
18)	kalläfk	**ካለፍክ**
19)	kalalläfh	**ካላለፍህ**
20)	kalalläfk	**ካላለፍክ**
21)	slalläfh	**ስላለፍህ**
22)	slalläfk	**ስላለፍክ**
23)	alalalläfhm	**አላላለፍህም**
24)	alalalläfkm	**አላላለፍክም**
25)	talläfh	**ታለፍህ**
26)	talläfk	**ታለፍክ**
27)	slätalläfh	**ስለታለፍህ**
28)	slätalläfk	**ስለታለፍክ**
29)	slätalläfh	**ስለታለፍህ**
30)	slätalläfk	**ስለታለፍክ**
31)	slasalläfh	**ስላሳለፍህ**
32)	slasalläfk	**ስላሳለፍክ**
33)	slalläfh	**ስላለፍህ**
34)	slalläfk	**ስላለፍክ**
35)	kätalalläfh	**ከታላለፍህ**
36)	kätalalläfk	**ከታላለፍክ**
37)	kätalläfh	**ከታለፍህ**
38)	kätalläfk	**ከታለፍክ**
39)	kasalalläfh	**ካሳላለፍህ**
40)	kasalalläfk	**ካሳላለፍክ**
41)	kasalläfh	**ካሳለፍህ**
42)	kasalläfk	**ካሳለፍክ**

DECLARATION

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

KIBUR LISANU WUDINEH
June 2002

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

Dr. Yonas Admassu

W/o Woinshet Abdella

Ato Solomon Berhanu