ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

SUB-WORD BASED AMHARIC WORD RECOGNITION: AN EXPERIMENT USING HIDDEN MARKOV MODEL (HMM)

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Masters of Science in Information Science

BY

KINFE TADESSE MENGISTU

JUNE 2002

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

SUB-WORD BASED AMHARIC WORD RECOGNITION: AN EXPERIMENT USING HIDDEN MARKOV MODELS (HMMS)

By
KINFE TADESSE

Name and Signature of Members of the Examining Board

Ato Getachew Jemaneh, Chairman, Examining Board _____

Ato Daniel Aberra, Advisor _____

Ato Ermias Abebe, Advisor _____

Ato Solomon Berhanu, Advisor _____

Dr. Fiaz Hussein, External Examiner _____

DEDICATION

To all people who have played a role in shaping me into the person I am today.

ACKNOWLEDGMENT

All of my efforts would have gone for naught if it had not been for the importunate help of the Almighty God. I would like to express my heartfelt gratitude to Him in the first place.

My special thanks are to my advisor Ato Solomon Berhanu who has been my resource in carrying out this research. Without him, things would have been much more difficult. I also owe a debt of gratitude to my advisors Ato Daniel Aberra and Ato Ermias Abebe for critically reviewing this manuscript and enriching the text with their constructive criticism.

I am greatly indebted to Ato Tesfaye Biru without whose suggestions the start of this research would have been impossible.

I sincerely acknowledge all SISA staff for their helpful ideas and cooperation in all possible ways. My earnest gratitude is extended to people who have devoted their precious time to provide me with their voice patterns for the experiment.

My sincere thanks go to MicroLink Information Technology College for sponsoring my study and its stakeholders for being very kind and cooperative during my journey to this end.

Finally, during my stay in the University the support and encouragement of my family, friends and relatives are worth being acknowledged.

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

ABSTRACT

In this study, the potential of Hidden Markov Model (HMM) for the development of Amharic speech recognition system has been investigated and in the course of building the recognizers the popular toolkit Hidden Markov Model Toolkit (HTK) was used.

In the process of building the recognizers, the speech data is recorded at a sampling rate of 16KHz and the recorded speech is then converted into Mel Frequency Cepstral Coefficient (MFCC) vectors for further analysis and processing.

Since large vocabulary systems are envisaged, sub-word modeling is pursued. Sub-word modeling refers to a technique whereby one HMM is constructed for each sub-word unit (phoneme, triphone, syllable, etc.). Phonemes, tied-state triphones and CV-syllables have been considered as the basic sub-word units and have been used to build phoneme-based, tied-state triphone based and CV-syllable based recognizers, respectively.

In this study, an extensible 170-word vocabulary is constructed and both speaker-dependent and speaker-independent models are built using 15 speakers (8 male and 7 female) in the age range of 20 to 30. Five untrained speakers who had no involvement in training the models are also used to test the speaker-independent models.

The results obtained are promising and have shown the potential of tied-state triphones as good sub-word units for Amharic. In fact, phonemes also have produced encouraging recognition performance. Even though CV-syllables appear to be more convenient for Amharic, this research has not proved that and is recommended for further research.

CHAPTER ONE

INTRODUCTION

1.1. BACKGROUND

Speech interface in a user's own language is an ideal means of communication as being the most natural, flexible, efficient and convenient option allowing hands and eyes to be free (Nahm and Slater, 1997). Moreover, speech is human's fastest and highest capacity output channel in that most computer users can speak about 10 times faster than they can type while skilled typists type about 2 times slower than they can speak (Turn, 1974; Lea, 1968).

Speech is the ultimate, universal mode of human communication and it is how man should be able to interact with computers. Armstrong (1994) asserts that now is the time for speech to begin to supplement the keyboard and the mouse.

Historically, man had to tailor his mode of communication to suit the technical and operational requirements of the computer rather than the other way round (Martin, 1976). This need no longer be the case, and given the current state of computer technology, there is no good reason why man should not communicate with machines in the most natural way.

This coupled with man's curiosity to build machines that mimic humans has captured the attention of researchers for decades. The first and probably the most important step toward such a communication mode with computers is building *automatic voice input/output systems* (Juang and Furui, 2000).

With in the area of man – machine communication by voice, there are three main avenues of research; namely, (Philip and Young, 1987):

- *Speech synthesis (Voice output);*
- *Speaker recognition (voice recognition);* and
- *Speech recognition (voice input)*

Speech synthesis is a technology for generating voice output from a computer. *Speaker recognition*, on the other hand, involves identifying a specific speaker out of a known population, or verifying the claimed identity of a user, thus enabling controlled access to locales and services (Zue and Cole, 1995).

Computer *speech recognition* is the ability to take speech as input and produce a transcript of that speech as output (Rodman, 1999: 101). In technical terms, it is the conversion of an acoustic signal to a stream of words.

Speech recognition systems are classified based on a number of parameters. Some of these classifications include:

- *Discrete vs. Continuous;*
- *Speaker-dependent vs. Speaker- independent;*
- *Context-sensitive vs. Context- insensitive; and*
- *Large vocabulary vs. Small vocabulary speech recognition systems.*

1.1.1. Discrete Speech Recognition Systems

Discrete speech recognition systems are systems that require the speaker to pause briefly between words (Zue and Cole, 1995). The pause is required to help the processor identify word boundaries and to prevent crossword coarticulation from distorting the acoustic pattern of the word to be recognized (Markowitz, 1996: 129). Thus, larger vocabularies, and hence richer domains of discourse, are achievable with discrete speech at a cost of lower speech rates and some user inconvenience.

1.1.2. Continuous Speech Recognition Systems

Speech is said to be continuous when it is uttered as a continuous flow of sounds with no inherent separations between them (Markowitz, 1996: 7). It is our recognition of the distinct words in our mind that make us think speech sounds consist of words with clearly marked boundaries.

Continuous speech recognition system is more difficult than discrete recognition system because continuous speech is characterized by massive variability that includes:

- A*coustic variability:* that results from changes in the environment,
- *Intra-speaker variability* that results from changes in the speaker's physical and emotional state; and
- *Inter-speaker variability* that results from differences in socio-linguistic background, dialect, vocal tract size and shape.

Other factors that make continuous speech recognition more difficult include (Lee *et al.*, 1996):

- *Word boundaries* are typically not detectable in continuous speech without knowledge of the language;
- *Coarticulation effects* (inter-unit influences) are much stronger in continuous speech, causing the same sound to appear differently in various contexts; and
- *Content words* (nouns, verbs, adjectives, etc.) are emphasized, while function words (articles, prepositions, pronouns, etc.) are poorly articulated.

These ambiguities can be resolved in human-to-human communication as both parties have the knowledge of the pragmatic and grammatical rules of the language to understand the semantic content of the message.

One approach to build a continuous speech recognition system is to compile and incorporate knowledge from a variety of knowledge sources so that the system mimics a human being. This approach is briefly discussed in section 2.4.4.

Some systems also incorporate *connected word recognition*, wherein the inter-word pauses are not necessary, but the speaker must avoid running adjacent words into one another (Waterworth and Talbot, 1987: 50). Connected means that the words are fully pronounced but that there need not be pauses between them (Rodman, 1999: 111). Like continuous speech, it is difficult to determine where one word ends and another begins (Waibel and Lee, 1990: 9) and unlike continuous speech, connected speech is free of *extraneous sounds*, *slurrings* and massive *inter-word coarticulation effects* (Rodman, 1999: 112).

1.1.3. Speaker-dependent Recognition System

A speaker-dependent system uses speech samples from the target speaker to learn the model parameters of the speaker's voice. To supply a reference copy of ones voice pattern (called templates), one must speak into the system using a microphone, repeating each word several times (Philip and Young, 1987). This process is called *training.*

Speaker-dependent recognition systems may be classified into two; namely,

- *Speaker-dependent /Isolated word system*
- *Speaker-dependent/Continuous speech system*

The simplest and most common type of speaker-dependent recognition system is *Speaker-dependent /Isolated word system.* Basically, this system recognizes the voice patterns of only one person and words must be spoken in isolation with clear pauses between them (Ibid.).

Speaker-dependent/Continuous speech systems are complex systems that are capable of dealing with continuous speech, such as a single sentence from a trained user. These are complex because they must not only recognize the voice patterns produced but also analyze the syntax of the string of words to deduce its meaning. The vocabulary size of speaker-dependent/continuous speech systems is usually small (Ibid.).

1.1.4. Speaker-independent Recognition Systems

Speaker-independent systems are designed to be used by virtually any user who wants to use them with no enrollment. Therefore, the reference models for these applications must recognize large, heterogeneous populations of speakers (Markowitz, 1996: 106). However, given the many sources of inter-speaker variability the design of a speaker-independent recognition system is by far more complex than a speaker-dependent recognition.

Speaker-independent systems, too, can be classified into two; namely,

- *Speaker-independent /Isolated word systems*
- *Speaker-independent /Continuous speech systems*

Speaker-independent /Isolated word systems are intended for virtually anyone without enrollment. The vocabulary is limited, typically only ten digits and fewer words. Systems of these kinds are suitable for a large number of people using a restricted vocabulary (Philip and Young, 1987).

Speaker-independent /Continuous speech systems are the ultimate goal of speech recognition research. However on account of the many difficulties involved in continuous speech recognition, it is generally accepted that the present situation makes only fairly restricted systems feasible (Ibid.)

1.1.5. Context Sensitive vs. Context Insensitive Systems

When speech is produced as a sequence of words, language models or artificial grammars can be used to restrict the permissible combination of words. More general language models approximating natural language are specified in what is known as a *context-sensitive grammar* (Zue and Cole, 1995). Speech recognition systems that increase their accuracy by anticipating or

limiting what can be said at any given time are referred to as *context-sensitive recognition systems* whereas systems that allow users to say anything, anytime without any constraint are called *context insensitive recognition systems*. Context-sensitive recognition systems are more difficult to implement on computers than context-insensitive ones.

1.1.6. Large Vocabulary Systems vs. Small Vocabulary Systems

Despite the fascination with unlimited vocabularies most applications involve much smaller vocabularies (Markowitz, 1996: 75). A telephone dialing system for instance, requires fewer than twenty words and most manufacturing applications use fewer than hundred words. On the other hand, dictation systems require large vocabulary.

Large vocabulary systems are very attractive for applications such as dictation and may be poorly suited to *smaller-vocabulary* command and control applications (Rodman, 1999: 114).

1.1.7. Application Areas of Speech Recognition

There are many reasons why many researchers are being committed to develop systems with speech recognition interfaces. One of the justifications is its *naturalness*. Speech is the most natural and universal method of communication between people. The aim of speech recognition is to extend that communication modality to interaction with computers. Speech recognition is being programmed into computer software to provide a more natural interface for novice computer users, including busy professionals who cannot take time to learn complex keyboard commands. It reduces training time while increasing ease-of-use (Markowitz, 1996: 4).

According to Rodman (1999) five general areas of speech recognition applications have been identified: *assistive technology, command and control, telecommunications, data entry and retrieval,* and *education.*

Speech recognition applications can be used as an *assistive technology* the primary purpose of which is aiding persons with severe disabilities. Speech recognition systems enable handicapped and visually impaired individuals control their environment by providing a voice interface to a computer, telephone, fax machine, etc (Rodman, 1999: 259).

Command and control refers to the control of machines via computers which are themselves instructed through speech. Such applications are useful in industrial, military and healthcare environments (Ibid., 264).

Speech recognition is also used in *Telecommunications*. Since 1995 most major cellular telephone service providers were offering *voice-activated dialing services* and the number of voice-activated systems in operation has been increasing exponentially (Ibid., 262). Moreover, global telecommunication companies and researchers are working very hard on *automated spoken language translation systems*. The service, when realized, enables people speaking different languages to communicate easily and readily without language and cultural barriers (Rodman, 1999: 263).

Speech recognition is also considered vital in *data entry and retrieval* process when "hands-busy, eyes-busy" or "remoteness" conditions prevail (Ibid., 268). Data entry and retrieval refers to storage and retrieval of data by voice.

One of the ultimate goals of speech recognition is to create a human-like *listening typewriter* or *automated secretary* (Markowitz, 1996: 200). Dictation systems are one successful speech recognition services that enable user to dictate directly into the computer.

Speech recognition can also play a role in *Computer Assisted Instruction (CAI)* for disabled persons who are unable to use the keyboard or a pointing device (Ibid., 272). Moreover, properly designed Computer Assisted Instruction (CAI) systems provide an opportunity for anyone to acquire new skills or upgrade old ones in a cost-effective and efficient manner. A computer with speech recognition ability is a perfect *language tutor – patient, positive,* and *cheerful* (Ibid., 272).

These are very motivating application areas in which many subsequent researches could be conducted to adopt these technologies into our country with our own languages. This project is an initial undertaking towards that end.

1.2. STATEMENT OF THE PROBLEM

The ultimate goal of any research in speech recognition is to develop a system that can understand the speech of anyone who needs to use the system. To meet this requirement, the system must be able to handle *inter-speaker* and *intra-speaker variability,* recognize *unrestricted vocabulary* and understand *fluent, conversational speech.* Although research in speech technology is about five decades old, such a system has only been a dream so far. However, intensive researches have been going on toward that end.

The speech recognition systems that have been developed so far are language dependent and are limited to some technologically favored countries of the world. Researches carried out on speech technology for the Amharic language are very few in number. On Amharic speech recognition,

only one research has been done by Solomon Berhanu (2001) who has endeavored to build speaker-dependent and speaker-independent HMM-based isolated Amharic Consonant-Vowel (CV) syllable recognition systems. Based on his findings and the nature of the Amharic language, he proposed that CV-syllables are good candidates to play the role of basic units of recognition.

This paper is an extension of Solomon's work and endeavors to build sub-word based Amharic isolated word recognizers using CV-syllables and other sub-word units as basic units of recognition. This research takes the research in the area one step ahead and marks a milestone for the subsequent researches that may be conducted in the field.

In order to realize the envisaged system, a number of problems need to be addressed. The first problem that must be dealt with is to decide on the basic unit of recognition. The possible units are *words, syllables*[1], *phonemes*[2] and *triphones*[3]. These units should be thoroughly examined based on a number of parameters and this research compares these units in terms of their suitability for building sub-word based recognition systems is concerned.

The next question that should be addressed is "is the speech to be recognized *discrete, connected,* or *continuous?"* As explained in section 1.1.2, continuous speech is considerably more difficult than isolated word recognition. In order to build continuous speech recognition system, some basic linguistic knowledge-base ought to be constructed. The current work does not assume any knowledge-base and the term "*recognition*" is used as distinct from understanding where no linguistic or semantic analysis is involved.

[1] A syllable is a phonological unit of a language with a vowel as its nucleus (Aster, 1987:27).
[2] A phoneme is a sound that distinguishes one word from another (Markowitz, 1996: 58).
[3] A triphone is a phone that takes its left and right phonetic context into consideration (Lee, 1990).

Automatic speech recognition systems that use purely acoustic phonetic analysis perform well only on slowly pronounced isolated words (White, 1990). Even though this is not the way people usually talk to one another, and pronouncing words in isolation is slow and inconvenient compared to continuous speech, it is useful and is the basis of many word recognition systems.

Another dimension of difficulty is the size of the vocabulary. As the current work is based on sub-word units, the envisaged system is a large vocabulary system. In sub-word based speech recognition systems, it is not necessary to train all words in the vocabulary; only sub-word units are trained (Roucos and Dunham, 1987).

The paper also attempts to develop both speaker-dependent and speaker-independent systems using the various sub-word units to be identified.

1.3. JUSTIFICATION OF THE STUDY

Amharic is one of the most widely spoken languages of the nation and the working language of the Federal government. However, little has been done on the language and the opportunities of information technology could not be properly exploited, yet. This research and the preceding work by Solomon (2001) attempt to light a candle in the area of speech technology for Amharic.

Automatic speech recognition system for Amharic is a very rich but almost untouched field which needs comprehensive and subsequent extensive researches for both practical and intellectual reasons. Practically, speech recognition systems solve problems, improve productivity, and change the way we run our lives.

Intellectually, speech recognition holds considerable promise as well as challenges for researchers and students (Waibel and Lee, 1990).

In view of that, this endeavor attempts to investigate the potential of Hidden Markov Model for the development of Amharic speech recognition system and identifies good sub-word units that can be used to build speech recognizers using HMM models.

Furthermore, the output of this investigation may be used by other researchers and students as an input for further study towards the far-reaching goal of developing a full-fledged Amharic speech recognition system.

1.4. OBJECTIVES OF THE STUDY

1.4.1. General Objective

The general objective of this study is to examine Amharic sub-word units and to present a comparative analysis of these sub-word units based on the recognition performance of the recognizers built using these units.

1.4.2. Specific Objectives

To accomplish the above general objective, the following specific objectives have been formulated:

- To identify the distinct sub-word units of the Amharic language in relation to speech recognition.
- To select some of the sub-word units based on their frequency of occurrence in normal Amharic speech and text.
- To choose Amharic words that are composed of these selected sub-word units in such a way that the selected utterances account for the variation in pronunciation of the sub-word units in different contexts.

- To build prototype Amharic word recognizers by employing HTK toolkit using the identified sub-word units.
- To evaluate the performance of the prototype recognizers on Amharic isolated word utterances; and
- To present a comparative analysis of the results obtained using the identified units of recognition.

1.5. METHODS

In order to achieve these objectives, this study has employed the following methods:

1.5.1. Review of Related Literature

It is vital that researchers interested in speech processing have detailed knowledge of the speech sounds of the particular language they are involved in (Rodman, 1999: 6). Therefore, review of literature on Amharic and English is performed to better understand the characteristic features of the languages, their similarity and their differences.

Moreover, to learn what others have done in the area and to better understand the problem a comprehensive investigation of available literature on automatic speech recognition has been made.

1.5.2. Data Preparation Techniques

Data needed for an automatic speech recognition system is obviously speech. The speech data is divided into 2 subsets; namely, *training set* and *test set.* The test set is further divided into two subsets: words selected from the training set and words not included in the training set. These data have been recorded using a tool in HTK called HSLAB and a multimedia PC with a

specification of Intel ® Pentium; ® 4 CPU 1.70 GHz and 261,136 KB RAM with a pair of speakers (Harman/Kardon) and a Labtec® AM-22 microphone.

For each recorded speech, the associated transcription of both training set and testing set are also prepared.

1.5.3. Modeling Techniques

The experiment is performed using the *Hidden Markov Model*. The Hidden Markov Model (HMM) is a composition of two stochastic processes, a hidden Markov chain, which accounts for temporal variability, and an observable process, which accounts for spectral variablility (Mori and Brugnara, 1995). During the past several years it has become the most successful speech modeling technique[4] . The main reason for this success is its wonderful ability to cope with the most important sources of speech ambiguity and its flexibility to allow the realization of recognition systems with dictionaries of tens of thousands of words (Ibid.).

To build and manipulate the HMM, the portable toolkit, the *Hidden Markov Model Toolkit (HTK)* is used in the course of this research. HTK is primarily used for speech recognition research[5].

HTK consists of a set of library modules and tools available in C source form. The tools provide sophisticated facilities for *speech analysis*, *HMM training*, *testing* and *performance evaluation*[6].

[4] http:// jedlikphy.blue.hu/~gerjanos/HMM/node4.html
[5] http://htk.eng.cam.ac.uk
[6] Ibid.

1.5.4. Analysis Techniques

Like other systems, a speech recognition system must pass through a rigorous testing procedure before it is delivered or put to use. The most important testing parameter used in evaluating speech recognition systems is *accuracy of recognition*. Most recognition systems make three common recognition errors (Markowitz, 1996: 49):

- *Deletion:* Dropping of units (words, syllables, or phonemes).
- *Substitution*: Replacing a spoken utterance (word, syllable, or phoneme) with another, usually similar unit. Put another way, substitution errors produce an incorrect translation for a given input. These are the most damaging errors.
- *Insertion*: Adding unwanted unit (a word, a syllable, or a phoneme).

To evaluate the performance of the recognizers, a recognition tool called HVITE and an analysis tool called HRESULTS included in the HTK toolkit are used.

To determine the recognition accuracy, the recognized word string is aligned with the correct word transcription using a string match algorithm. And the matches are counted and are reported as correct recognition and the errors are also reported with their respective types.

1.6. SCOPE AND LIMITATION OF THE STUDY

This study has attempted to carry out three major activities. First of all, it has tried to investigate the basic units of recognition for Amharic and has identified possible contenders. Second, prototype Amharic word recognizers have been built for both speaker-dependent and speaker-independent models using these contending sub-word units. Last but not least, comparative analysis of these contending sub-word units has been made based on the recognition performance

of the respective recognizers. Two of the identified sub-word units proved to be strong contenders and have been subjected to further examination and conclusions have been made based on the results obtained from the examination.

The scope of the study is limited to isolated Amharic word recognition and it is based on pure acoustic phonetic information and the term recognition is used as distinct from understanding where no linguistic and semantic analysis is involved.

The main limitation of this study was the absence of a ready-made Amharic reference database or corpus for speech recognition researches. This paper had to go to the extent of preparing speech database and deal with the rest of the experiment based on this small database. Though every possible effort has been made to impart good coverage and phonetic balance[7] into the '*database*', experimental results have demonstrated that more effort is required to prepare a phonetically balanced database with good coverage.

The other constraint associated with building the database has been to find people who had not been too busy to devote their precious time with the researcher speaking so many words (about 220) possibly more than once. Human and environmental factors had their own impact on the results of the experiments because some of the speakers were stressed and were not consistent in their utterances and the recording laboratory was noisy and was not favorable for the research.

The last and the obvious limitation was time that constrained the researcher to select some (104) CV-syllables out of the 231 or 20 phonemes out of the 39-phoneme set. Had there been more time, the rest of the sub-word units could have been included and any Amharic word spoken in isolation could have been recognized with good accuracy (about 85%).

[7] The sounds used in the system are trained roughly equal number of times.

1.7. ORGANIZATION OF THE THESIS

This paper is organized in five chapters. Chapter 1 introduces the basic concepts of speech recognition including definition of terms and application areas of automatic speech recognition. Furthermore, statements of the problem, justification of the study, objectives of the research, and methods employed for the successful completion of the study are briefly presented.

Chapter 2 presents the basic principles of articulation, representation and speech recognition approaches. Amharic phonetics and their representation as well as the Amharic writing system have been introduced very briefly. Signal analysis and representation of speech are briefly introduced. Furthermore, the different automatic speech recognition approaches are briefly discussed in this chapter.

Chapter 3 deals with the basic principles of Hidden Markov Models and introduces the HTK toolkit briefly.

Chapter 4 discusses the design and implementation details of the Amharic word recognizers using different sub-word units as the basic units of recognition and the results obtained are discussed.

Chapter 5 presents the conclusions drawn out of the experiments and forwards recommendations for future work.

CHAPTER TWO

SPEECH: ARTICULATION, REPRESENTATION, AND RECOGNITION APPROACHES

2.1. INTRODUCTION

In order to build a speech recognition system, a comprehensive analysis of speech as a stream of acoustic signals (*acoustics*) is essential. Moreover, understanding of how speech sounds are produced by the human vocal apparatus (*articulation*) and how speech is represented inside a computer (*digitization*) are vital.

Section 2.2. presents a review of Amharic *phonetics* and discusses the Amharic consonants and vowels as classified by *point* and *manner of articulation* very briefly.

Since the acoustic waveform need to be converted somehow to meet the operational requirements of the computer, *signal analysis* and *representation* of speech in computer are briefly discussed in section 2.3.

Once an utterance is recognized, it has to be transcribed to an appropriate symbol sequence of the target language. To this end, basic units of recognition and the Amharic writing system are briefly discussed in section 2.4 and 2.5, respectively.

Finally, the various approaches to speech recognition are briefly introduced in section 2.6.

2.2. BASICS OF AMHARIC PHONETICS

Linguists decompose a spoken language into elements of linguistically distinct sounds called *phonemes*. These phonemes are determined and classified according to their corresponding articulatory configurations (Juang and Furui, 2000). *Articulatory phonetics* deals with how the human vocal apparatus is manipulated to produce sounds (Clark *et al.*, 1985:204). The basic assumption of articulatory phonetics is that sounds are best described in terms of the configurations of the vocal tract necessary to utter the sounds (Ibid.)

Sounds can also be classified as vowels and consonants. Vowel and consonant sounds are produced in fundamentally different ways. While consonants are articulated with a substantial degree of obstruction in the oral cavity, vowels are produced with a relatively free airflow (Clark *et al.*, 1985: 216).

2.2.1. Amharic Consonant Articulations

For the purpose of speech recognition, the following summary suffices to describe any Amharic consonant by referring to its *point of articulation*, its *manner of articulation*, and the presence or absence of *voice* (Ibid., 205).

Table 2.1. Amharic Consonants

Manner of articulation	Point of Articulation	Bilabial	Labio-dental	Alveolar	Palatal	Velar	Labio-Velar	Glottal
Stops	Voiced	ለ /b/		ደ /d/		ገ/g/	ጓ/g^w/	
	Voiceless	ፐ /p/		ተ /t/		ከ/k/	ኳ/k^w/	ዕ (?)
	Ejective	ጰ /p'/		ጠ /t'/		ቀ/k'/	ቋ/k'^w/	
Fricatives	Voiced		ቨ /v/	ዝ /z/	ዠ /ž/			
	Voiceless		ፍ /f/	ስ /s/	ሸ /š/			ሀ/h/ ኍ /h^w/
	Ejective			ጸ /s'/				
Affricates	Voiced				ጀ /j/			
	Voiceless				ቸ /č/			
	Ejective				ጭ/c'/			
Nasals		ም /m/		ን /n/	ኘ /ñ/			
Liquids				ል /l/	ር /r/			
Glides		ው /w/			ይ /y/			

2.2.2. Vowel Articulations

Vowels are open sounds, made largely by shaping the vocal tract rather than by interfering with the flow of air stream (Clark *et al.*, 1985:216).

Vowels are most usefully described in terms of the *position of the tongue* as they are articulated. A vowel articulated with the body of the tongue relatively forward is classified as a *front vowel*; one made with the body of the tongue relatively high is a *high vowel*. Vowels produced with the body of the tongue neither high nor low are called *mid vowels*. Vowels produced with the tongue body front are called *front vowels* while those made with the tongue body back are called *back vowels*. Those vowels made with the tongue body neither front not back are called *central vowels* (Ibid.).

Vowels accompanied by lip rounding as in (u and o) are called *rounded vowels* while the other vowels are called *unrounded vowels.*

Table 2.2. Vowel Articulations

	Front/Unrounded	**Central/Unrounded**	**Back/Rounded**
High	ኢ /i/	እ /I/	ኡ /u/
Mid	ኤ /e/	ኧ /E/	ኦ /o/
Low		አ /a/	

The Amharic vowels, the transcription used in this paper, their symbols in the Amharic orthography and some example words illustrating their sound values are given below as Dawkins (1969) presented it:

- ኧ(E), as in "her"
- ኡ (u), as in "mood"
- ኢ (i), as in "seat"
- አ (a), as in "bath" or "but"
- ኤ (e), as in "late"
- እ(I), as in " speaker"
- ኦ(o), as in "coat"

2.3. SIGNAL ANALYSIS AND REPRESENTATION

Signal analysis is the first step in every automatic speech recognition system. The aim of signal analysis is to obtain the most important features in the speech waveform that are critical to the recognition or identification of the unknown utterance.

Like all sounds, speech is an analog waveform: *a continuous flow of sound waves and silences.* In order for a recognition system to utilize speech data, all formants, noise patterns, silences, and coarticulation effects must be captured and converted to a *digital* format (Markowitz, 1996:33).

The computer's "ear" consists of a *microphone* and an *analog-to-digital* (A-to-D) converter. The microphone converts the vibrations of sound to an analog electrical signal. The analog to digital converter reduces the continuously varying voltages of the electrical signal to a sequence of digits (Rodman, 1999:51).

Since the amount of data contained in speech signals is tremendously large, the amount of data in the signal must be dramatically reduced. In fact, some of the data in the speech wave are irrelevant to the recognition process while other pieces of data are redundant (Markowitz, 1996:6). The intuitive solution to reduce the quantity of data is to take representative samples from the signal.

The representative samples extracted from the analog signal must then be converted into a digital form. This process of converting the analog waveform representation into a digital code is called *analog-to-digital conversion* or *encoding* (Ibid.: 34).

2.3.1. Sampling

The most commonly used sampling technique is based on the *Nyquist sampling theory (*Rodman, 1990:60*)*. The Nyquist theorem states that if an analog signal is sampled at regular intervals at a rate at least twice the highest frequency in the channel, the samples will contain sufficient information about the signal to allow its reconstruction. The sampling rate is the number of points observed per second.

Most speech recognition preprocessors take 8,000 to 10,000 samples per second (Markowitz, 1996:34). In statistical automatic speech recognition, the speech waveform is sampled at a rate between 6.6 KHz and 20KHz (Hunt, 1995).

No matter what the sampling rate is, the speech signal must be suitably *low-pass filtered* prior to the sampling process to eliminate undesired high frequencies of the speech and high frequency noise (Shafer and Rabiner, 1975).

A *filter* takes sound of any frequencies as input and outputs sound only in certain frequency ranges. A *low-pass filter* with a cutoff of N Hz outputs only frequencies of N Hz or less. A *high-pass filter* with the same cutoff outputs only frequencies of N Hz or greater. There are also *band-pass filters* that output frequencies within a certain predefined range, or ranges (Rodman, 1999:72).

2.3.2. Speech Encoding

Speech encoding is a process by which voice signals are digitally represented in the computer for further processing. Broadly speaking, there are two methods for encoding speech (Philip and Young, 1987):

- *Waveform Coding*, and
- *Vocoding (Analysis/Synthesis method)*

Wave coding is the process of representing speech by sampling and quantizing the analog signal after filtering. Waveform coders employ waveform reconstruction strategy – they are capable of reproducing an output speech which is almost an exact replica of the original input speech. Thus, they are able to preserve speaker-dependent characteristics of speech such as emotion,

articulation, stress, and speaker identity. This is achieved by the use of high voice digitization rate (VDR) (Ibid).

The most widely used and the simplest of the waveform coding techniques is the Pulse Code Modulation (PCM). It can be roughly viewed as a three-step process consisting of *Sampling, Quantizing*, and *Encoding*

PCM is based on the Nyquist sampling theory and each sample is called Pulse Amplitude Modulation (PAM) signal.

Quantization is the process of assigning a value to each PAM signal. Once the PAM signals have been assigned a value by the quantizing process, the next step encodes the samples into a binary bit string which is the digital representation of the input data.

The analog-to-digital conversion can be schematically represented:

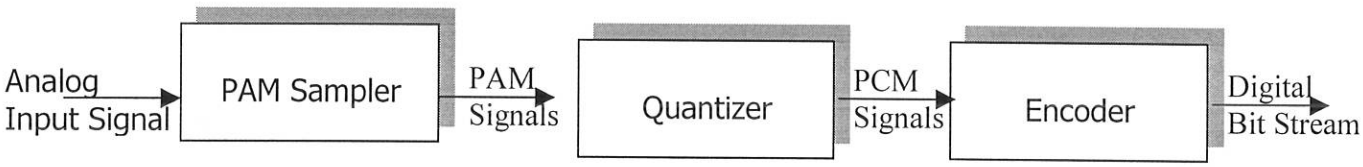

Figure 2.1. PCM analog to digital Conversion

A *vocoder* (also called *voice coder*) is an electronic model of the vocal tract capable of producing speech when provided with the proper input (Rodman, 1999:83). As pointed out earlier in this section, speech consists of a great deal of inherent redundancy. Vocoders take advantage of this fact and extract those parameters of voice which are absolutely essential for reproducing it without any perceptible loss of intelligibility (Philip and Young, 1987). Unlike waveform coders,

however, they are unable to retain the speaker-dependent attributes of speech. Vocoders operate at a relatively low voice digitization rate.

The most widely used vocoder is implemented by *Linear Predictive Coding* (LPC) technique (Rodman, 1999:84). *Linear Predictive Coding (LPC)* has become the dominant coding methodology for speech recognition. The basic idea behind linear predictive coding (LPC) is that a sample of speech can be approximated as a linear combination of the past '*P*' speech samples (Markowitz, 1996:34).

2.4. BASIC UNITS OF SPEECH

Speech recognition systems generally assume that a speech signal is a realization of some message encoded as a sequence of one or more symbols (Young *et al.*, 2000: 3). The role of a recognizer in a recognition system is, therefore, to identify the underlying symbol sequence for a given utterance.

In order to design a speech recognizer for any language, one must first establish some way of symbolizing the spoken utterances by some group of symbols that distinctly represent the sounds produced.

Words are the most natural units of speech because they are exactly what we want to recognize. Moreover, word models are able to capture within-word contextual effects (Lee, 1990). Therefore, for small-vocabulary systems word models will usually yield the best performance. However, using word models in large-vocabulary recognition introduces several problems. One of these problems is that since training data cannot be shared between words, each word has to be trained individually and each word has to be uttered *several times consistently*. Yet, this is

impractical since there are too many words to train adequately. In addition to this, the same word spoken with different pitches (high or low) may be recognized as different words and may degrade performance.

These problems can be alleviated, at least in theory, if smaller units, called sub-word units, are used to form the basis of the recognition process. Taking this fact into consideration, and since the intention of the project is to build a large vocabulary system, the first task undertaken is to identify good sub-word units that can be used to build the recognizer.

Good sub-word units should be *consistent* and *trainable* (Lee, 1990). Consistency means different instances of the same sub-word model have similar characteristics and trainability means that each speech unit can be trained on sufficiently many examples. Both consistency and trainability are important parameters because speech models require extensive amount of training data spoken several times by several speakers.

Based on these parameters, the main contenders considered to play the role of good sub-word units are the *phoneme,* the *triphone* and the *syllable.*

Phonemes are sounds that distinguish one word from another in a language and they economically represent speech because most languages have only fifty or fewer phonemes (Markowitz, 1996, 58). Amharic, for instance, has only 39 phonemes including (ሸ) (Baye, 1997). The list of Amharic phonemes is attached (Appendix B).

This implies the fact that vocabulary growth will not necessarily produce the corresponding linear increase in storage and computational demands associated with other methodologies (Markowitz, 1996:59).

However, this approach has several limitations one of which is it provides little assistance to handle *coarticulation effects*. By coarticulation effect, we mean, the effect of the preceding and following sounds in a given utterance. In order to handle this problem, research on statistical modeling of word segments proposed the use of *triphones* or *phonemes-in-context* (Markowitz, 1996:16). A triphone is a phone that takes its left and right phonetic contexts into consideration (Lee, 1990).

A syllable, on the other hand, may be defined as a phonological unit of a language with a vowel as its nucleus (Aster, 1981: 27). The words of all languages are made up of syllables (Rodman, 1999:33). However, languages differ in what comprises a syllable. Amharic, for instance, possesses about 14 syllable types as classified by Aster (1981), one of which is a CV type (a consonant followed by a vowel).

The use of a syllable as the unit for recognition has two advantages (CSTR, 1999):

1. Syllables exhibit far less context dependencies than phones; and
2. Groups of features (such as voicing, frication, etc.) are not rigidly aligned at phone boundaries, and can spread into neighboring segments. Therefore, it is believed that a syllable model can represent this feature overlap in a more natural way than phone models.

One basic limitation of syllables as basic units of recognition is that there are much more syllables than there are phoneme in a language. This imposes trainability problems.

2.5. THE AMHARIC WRITING SYSTEM

While the sound and the spelling systems of many languages do not correspond exactly (Clark *et al.*, 1985:205), the degree of misfit between sound and spelling in Amharic is not significant. Provided that one knows the alphabets of the Amharic language, the chances of pronouncing a word upon seeing it written for the first time are pretty good; furthermore, upon hearing a word in Amharic, one has a good chance of spelling it correctly.

There is disagreement among scholars whether the Amharic writing system is *syllabic* or *alphabetic*. Writing systems in which one symbol represents one syllable are called syllabic while writing systems in which one symbol represents one sound segment are called alphabetic (Clark *et al.*, 1985:705). Some writers (Bender, 1976; Cowley, 1976; Mullen, 1986) argue that the Amharic writing system is syllabic while others (Tadesse, 1994; Baye, 1997) say that it is not syllabic.

Amharic adopts the principle of writing one CV-syllable with one character. This reduces the number of signs to manageable proportions. For instance, the sequence 'laba' will always be expressed by signs representing la(ላ) ba (ባ) never by signs for syllables as lab ha, or la aba,....

In Semitic languages, generally, the consonants are all important. They are permanent framework of a word, while the vowels filling the intervening spaces are subject to variation (Dawkins, 1969:5). In Amharic writing system, each consonant of the language is paired with a vowel. According to this pairing every Amharic consonant has seven forms called *orders*. The entire table where these consonants and their orders are listed is known as *fidEl*. Two of the Amharic consonants with their seven forms and orthographic symbols are given below:

Orders:		1st	2nd	3rd	4th	5th	6th	7th
Transcription:		bE	bu	bi	ba	be	bI	bo
Orthographic symbol	:	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
CV representation	:	ብኧ	ብኡ	ብኢ	ብኣ	ብኤ	ብእ	ብኦ

Transcription:		dE	du	di	da	de	dI	do
Orthographic symbol	:	ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
CV representation	:	ድኧ	ድኡ	ድኢ	ድኣ	ድኤ	ድእ	ድኦ

A list of such symbols is called a *syllabary (*Clark *et al.*, 1985:705). As can be seen, the other forms are formed by taking the first form and then simply using diacritic marks.

The Amharic writing system has the following limitations, however (Bender, *et al.* 1976).

1. There are several duplicate characters that are normally used interchangeably.
2. It lacks a mechanism to mark gemination of consonants
3. There exists ambiguity between sixth-order symbols with and without vowel in different contexts.

These problems need to be properly addressed to realize a speech recognition system that accurately recognizes and transcribes Amharic Speech. The intuitive solution for the first problem is to use one form of the duplicate characters. For instance (ሰ,ሠ), (ፀ,ጸ), (ሃ, ሐ, ኀ, ኸ ,ኃ, ኘ), (ዐ, ዓ,አ, ኣ) are duplicates. The underlined ones are selected for the current work.

The symbols used for writing are not adequate to denote the frequent gemination encountered in Amharic speech. For instance, the Amharic word "ገና" gives completely different meanings based on whether the sound "ን" is geminated or not.

ገና gEna (meaning Yet, Still),

ገና gEn-na (meaning Christmas).

This implies that multiple forms of the same sound may require different reference models in the database depending on whether the sound is geminated or not.

The Amharic orthography as it is represented in the Amharic character set "FidEl" consists of 268 characters excluding the numerals and the punctuation marks.

If (ቭ) which is used only in loan words and its six forms are considered, the total character set will be 275. These list of characters are attached as Appendix A. After excluding *duplicates, numerals* and *punctuation marks,* the total number of distinct CV syllables and hence orthographic symbols in Amharic becomes 231.

2.6. SPEECH RECOGNITION APPROACHES

Four basic approaches have been followed and tested over the years[8] for developing automatic speech recognition system:

2.6.1. Template Matching

The most widespread, and commercially available approach to automatic speech recognition is *whole-word matching* or *template matching* (Waterworth and Talbot, 1987:36). The method involves obtaining one or more utterances of every word that is to be recognized from one or

[8] http://www.hltcentral.org/page-827.0.shtml

more speakers and storing the spectrographic representations of these utterances via a "*vocoding*" (filtering and digitizing) process. This is known as *training* the recognizer (Ibid.).

Spoken input by end users is organized into templates prior to performing the recognition process. During recognition the incoming utterance is compared against some or all of the stored templates. An utterance will be recognized as being a particular word in the event that the template for that word provides both the closest match to the utterance, and provides a sufficiently close match. The requisite closeness of the match, the *'reject threshold criterion'*, can be preset by the user to suit the characteristics of the task (Ibid.:50). A "*word*" here can be a single word or a short phrase.

A whole-word template matching recognizer can only be successful on those words with which it has been trained. Nor can it succeed in the recognition of continuous or connected speech. With a reasonably large vocabulary, it is obviously inefficient.

2.6.2. Stochastic Approaches

The term stochastic refers to the process of making a sequence of non- deterministic selections from among sets of alternatives. They are non-deterministic because the selections are governed by the characteristics of the input and not specified in advance (Markowitz, 1996:39).

Speech recognition can be viewed as a situation where decisions or selections have to be made based on *incomplete* or *uncertain* information and stochastic modeling is a flexible and general method for handling situations where uncertainty prevails. Stochastic approaches use *probability* as a measure of uncertainty, where a joint probability distribution of a set of variables is used to

describe the relationships among the variables, and conclusions are drawn using certain well-known formulas (Castillo *et al.*, 1997:9).

Like template matching, stochastic processing requires the creation and storage of models of each of the items that will be recognized. However, unlike template matching stochastic processing involves no direct matching between stored models and input. Instead, it is based upon complex statistical and probabilistic analyses that are best understood by examining the network like structure in which those statistics are stored (Markowitz, 1996:39). The hidden Markov model (HMM) is one popular stochastic approach that can be used to deal with speech recognition problems (For details of HMM, see Chapter 3).

2.6.3. Neural Networks

There are many different ways in which people refer to the same type of neural networks technology[9]. Neural networks are described as *connectionist systems* because of the connections between the individual processing nodes. They are also known as *adaptive systems*, because the values of these connections can change so that the neural network performs more effectively. People also refer to them as *parallel distributed processing systems* after the way in which the many nodes or *neurons* in a neural network operate.

Neural Networks are excellent classification systems. They specialize in classifying noisy, patterned and variable data streams containing multiple, overlapping, interacting, and incomplete signals (Markowitz, 1986:41). Speech recognition is a classification task that has all of these characteristics, making neural networks a possible alternative to speech recognition.

[9] http://www.accurate.automation.com/products/nnets.htm

Like stochastic and template matching techniques, neural networks require training. However, neural networks do not require that a complete specification of a problem be created prior to developing a network-based solution. Instead, networks *learn patterns* solely through exposure to large numbers of examples, making it possible to construct neural networks for auditory models and other poorly understood areas (Ibid.).

Neural networks use manually-segmented or labeled speech elements and often store speech items as whole units rather than as time sequences (Ibid.:44).

2.6.4. Knowledge-Based Approaches

The basic idea of the knowledge-based approach to speech recognition is to compile and incorporate knowledge from a variety of knowledge sources so that the system mimics a human being.

The knowledge sources that are considered useful, at least potentially, include (Rabiner and Juang, 1993:52-53):

1. *Acoustic knowledge* – evidence of which sounds are spoken on the basis of spectral measurements and presence or absence of features
2. *Lexical knowledge* – the combination of acoustic evidence so as to postulate words as specified by a lexicon that maps sounds into words (or equivalently decomposes words into sounds)
3. *Syntactic knowledge* – the combination of words to form grammatically correct strings (according to a language model) such as sentences or phrases.

4. *Semantic knowledge* – understanding of the task domain so as to be able to validate sentences (or phrases) that are consistent with the task being performed, or which are consistent with previously decoded sentences.
5. *Pragmatic knowledge* – inference ability necessary in resolving ambiguity of meaning based on ways in which words are generally used.

All these knowledge sources interact in order to arrive at a decision about a speech sample. The crucial question, with such a complex scheme is "how do these levels interact?"

One of the most popular knowledge based speech understanding systems developed so far is *HEARSAY II* (Ibid:41). HEARSAY II attempted to provide what is known as the *'blackboard"* model to allow the interaction of the various knowledge sources. Each *knowledge source* (KS) is viewed as an information gathering process that 'writes' all its hypotheses on a globally accessible *'blackboard'* (Ibid.).

CHAPTER THREE

HMM AND HTK TOOLKIT

3.1 INTRODUCTION

HTK is a toolkit for building hidden Markov models (HMMs) where hidden Markov models are one type of stochastic signal processing models. The toolkit is is primarily designed for building HMM- based speech recognizers (Young *et al.*, 2000: 2).

In building HMM-based speech recognizers using HTK, there are three main stages[10]. The first stage called *data interface stage* provides the interface for data input and output. The second stage is the *training stage* that contains the corresponding training algorithms required by HMM. These training algorithms are used to estimate the parameters of a set of HMMs using training data and their associated transcription. The third stage is the *recognition stage* that contains the main recognition processing algorithms by which the recognizers attempt to recognize and transcribe the unknown input speech. An additional step is performance evaluation of the recognizer. HTK provides a number of tools for each of these processes.

In this chapter, the basic ideas of HMMs and their use in speech recognition are briefly introduced (Section 3.2). The sections that follow provide a brief overview of the HTK toolkit (Section 3.3). Emphasis is made on the tools that have been used in this work.

The major source for the latter sections in this chapter is mainly Young *et al.* (2000) and mention is made when the source differs from this.

---

[10] http://www.comp.nus.edu.sg/ ~luakt/htk/htk design. html

3.2. THE HIDDEN MARKOV MODEL (HMM)

Speech recognizers are meant to recognize and transcribe a spoken utterance. To achieve this, the continuous speech waveform should first be converted to a sequence of equally spaced *discrete parameter vectors*. This sequence of parameters is assumed to form an exact representation of the input speech. This, in fact, follows from the underlying assumption that for the duration covered by a single vector (typically 10ms or so), the speech waveform can be regarded as being stationary (Young *et al.*, 2000: 3). The recognition task attempts to map these sequences of speech vectors and the underlying symbol sequences.

HMM's approach to speech recognition can be, very briefly, outlined as follows (Jelinek, 1985):

Let $W=w_1, w_2, w_3,\ldots,w_n$ denote a string of n words, and let O denote the acoustic evidence on the basis of which the recognizer will make its decision to recognize the spoken words.

If P (W/O) denotes the probability that the words W were spoken given that the evidence O was observed, the recognizer is designed to decide in favor of a word string w_i satisfying

$$P(w_i/O)=\max P(w_i/O) \qquad (3.1)$$

3.2. THE HIDDEN MARKOV MODEL (HMM)

Speech recognizers are meant to recognize and transcribe a spoken utterance. To achieve this, the continuous speech waveform should first be converted to a sequence of equally spaced *discrete parameter vectors*. This sequence of parameters is assumed to form an exact representation of the input speech. This, in fact, follows from the underlying assumption that for the duration covered by a single vector (typically 10ms or so), the speech waveform can be regarded as being stationary (Young *et al.*, 2000: 3). The recognition task attempts to map these sequences of speech vectors and the underlying symbol sequences.

HMM's approach to speech recognition can be, very briefly, outlined as follows (Jelinek, 1985):

Let $W=w_1, w_2, w_3,\ldots,w_n$ denote a string of n words, and let O denote the acoustic evidence on the basis of which the recognizer will make its decision to recognize the spoken words.

If P (W/O) denotes the probability that the words W were spoken given that the evidence O was observed, the recognizer is designed to decide in favor of a word string w_i satisfying

$$P(w_i/O)=\max P(w_i/O) \qquad (3.1)$$

This probability is not directly computable, but using *Baye's* formula:

$$P(w_i/O) = \frac{P(w_i).\ P(O/w_i)}{P(O)} \qquad (3.2)$$

From the above formula it can be inferred that for a given set of prior probabilities $P(w_i)$, the most probable spoken word depends only on the likelihood of $P(O/w_i)$. Therefore, to help the recognizer's decision making, it is necessary to determine the nature of the acoustic evidence O. The transformation of the speech signal into the evidence O is called acoustic processing (Jelinek, 1985).

Let the acoustic evidence O be represented by speech vectors, defined as

$$O = o_1, o_2, o_3, \ldots o_t$$

where o_t is the speech vector (evidence) observed at time t. Given the dimensionality of the sequence O, the direct estimation of the joint conditional probability $P(o_1, o_2, \ldots / w_i)$ from examples of spoken words is not practicable. However, if *Markov model* is assumed, then estimation from data is possible since the problem of estimating $P(O/w_i)$ is replaced by the much simpler problem of estimating the Markov model parameters.

In Markov models, each state corresponds to an observable (physical) event. It is generated by a doubly embedded probabilistic process with an underlying stochastic course of action that is not observable (hidden), but can only be observed through another set of stochastic processes that produce the sequence of observations (Rabiner, 1990).

In HMM based speech recognition, it is assumed that the sequence of observed speech vectors corresponding to each unit is generated by a Markov model as shown in Fig 3.1. (Ibid., 4).

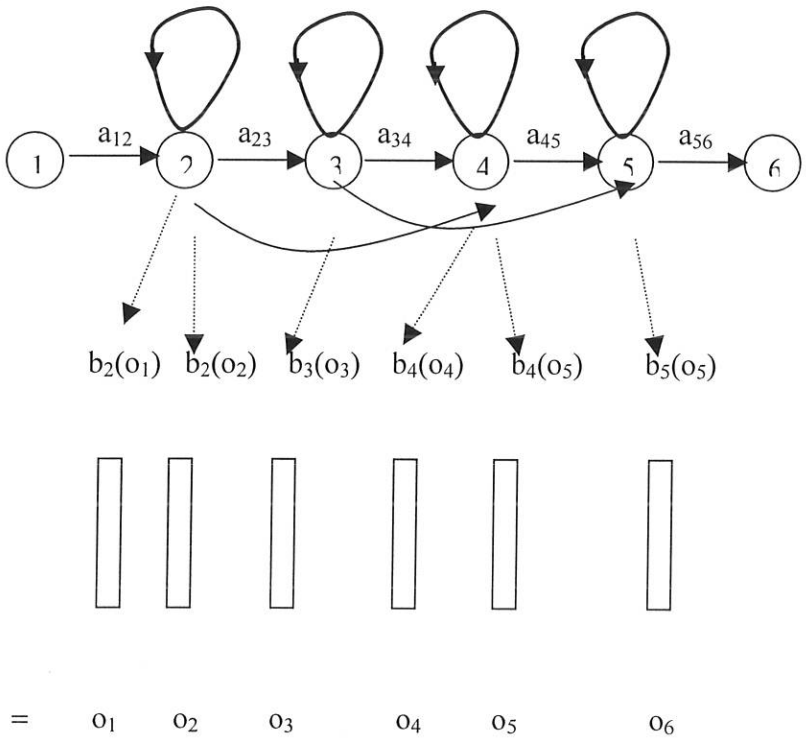

Figure 3.1. The Markov Model

The above model consists of 6 states. Each state j has an associated probability distribution $b_j(o_t)$ which determines the probability of generating observation o_t at time t and each pair of states i and j has an associated transition probability a_{ij}. The model entry state and the model exit state are non-emitting.

The joint probability that O is generated by the model M can be calculated from:

$$P(O/M) = \sum_{x} a_{x(0)x(1)} \prod_{1}^{T} b_{x(t)}(o_t)\, a_{x(t)x(t+1)} \qquad (3.3)$$

where x (1), x (2), x (3), . . ., x (T) are all possible state sequences. Since the underlying sequence is not known x(0) is constrained to be the model entry state and x(T+1) is constrained to be the model exit state.

The idea is to solve equation (3.1) because if it is computable, then the recognition problem is solved. Given a set of models M_i corresponding to recognition units w_i and assuming that

$$P(O/w_i) = P(O/M_i)$$

Equation (3.1) is solved by equation (3.2). In fact this assumes that the parameters $\{a_{ij}\}$ and $\{b_j(o_t)\}$ are known for each model M_i.

Given a set of training examples corresponding to a particular model, the parameters of that model can be determined automatically by a re-estimation procedure. Thus, provided that a sufficient number of representative examples of each unit can be collected then an HMM can be constructed which implicitly models all of the many sources of variability inherent in real speech (Young *et al.*, 2000:6).

For an HMM model to be useful in building speech recognizers, three fundamental problems must be solved. These problems are (Rabiner and Juang, 1993: 333):

Problem 1: Given a model and a sequence of observations, how do we compute the probability that the model produced the observed sequence?

Problem 2: Given the observation sequence $O = o_1, o_2, o_3, \ldots o_T$ and the model M, how do we choose a corresponding state sequence $Q = q_1, q_2, \ldots q_T$ which is optimal in some meaningful sense?

Problem 3: How do we adjust the model parameters so as to account for the observed signal?

In order to deal with these problems HMM possesses elegant and efficient algorithms; namely,

1. The *forward- backward* algorithm
2. The *Viterbi* algorithm
3. The *Baum-Welch* algorithm

Problem 1 is a problem of evaluating how well a given model matches a given observation sequence (Ibid.). To solve this problem the forward-backward algorithm is used (For details of the forward-backward algorithm refer Rabiner &Juang (1993); Young *et al.* (2000)).

To solve problem 2; that is, to find the single best state sequence,

$Q = \{q_1 q_2 \ldots q_T\}$ for the given observation sequence $O=\{o_1 o_2 \ldots o_T\}$ the *Viterbi* algorithm is used. The Viterbi algorithm is based on dynamic programming and it looks through a network of nodes for a sequence of HMM states that correspond most closely to the input. This is called the best path (Markowitz, 1996: 40). (For details of the Viterbi algorithm refer Rabiner and Juang (1993); Young *et al.* (2000)).

Problem 3 may be solved by using the *Baum-Welch re-estimation* algorithm. Given any finite observation sequence as training data, the model parameters can be estimated. In order to determine these parameters of a given HMM, it is first necessary to make a rough guess at what they might be. Once this is done, more accurate (in the maximum likelihood sense) parameters can be found by applying the so-called Baum-Welch re-estimation formula (Young *et al.*. 2000). (For details of the Baum-Welch algorithm refer Rabiner and Juang (1993); Young *et al.* (2000)).

The observation sequence used to adjust the model parameters is known as a training sequence since it is used to train the HMM. The training problem is the crucial one for most applications of HMMs, since it allows us to optimally adapt model parameters to observed training data – that is, to create best models for real phenomena (Rabiner, 1989).

3.2.1. Types of HMMs

HMMs can be classified in a variety of ways:

1. The *left-to-right*, directional HMM vs. *Ergodic* Models
2. *Continuous Density* vs. *Discrete Density* HMMs

3.2.1.1. Left-to-right vs. Ergodic Models

The most common HMM architecture found in speech recognition systems is the left-to-right, directional HMM (Markowitz, 1996: 57). The fundamental property of all left-right HMMs is that the state-transition coefficients have the property:

$$a_{ij} = 0, \text{ where } j<i$$

that is, no transitions are allowed to states whose indices are lower than that of the current state (Rabiner and Juang, 1993:348).

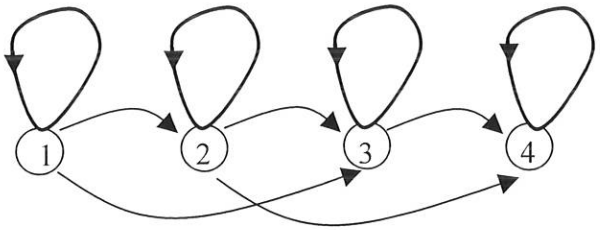

Figure 3.2. Left-to-right HMM

The *ergodic* or fully connected HMM is another common HMM architecture in which every state of the model could be reached from every other state of the model (Ibid.). Ergodic models are non-directional and link every state to every other state (Markowitz, 1996:57).

3.2.1.2. Discrete HMMs vs. Continuous HMMs

HMMs may be based on either:

1. Discrete probability distributions (*Discrete HMMs*)
2. Continuous output probability density functions (*Continuous HMMs*)

An HMM consists of states. Each state j has an associated observation probability distribution $b_j(O_t)$ which determines the probability of generating observation o_t at time t and each pair of states i and j has an associated transition probability (Young *et al.*, 2000:91). The *transition parameters* in HMM model temporal variabilities, while the *output distributions* model spectral variabilities.

The observation probability distribution may be a *discrete output probability distribution*, $b_j(O_k)$ for discrete HMMs or *continuous output probability density function* $b_j(X)$ for continuous HMMs (Huang and Jack, 1989).

HTK is designed primarily for modeling continuous parameters using continuous density multivariate output distributions (Young *et al.*, 2000:6). It can also handle observation sequences consisting of discrete symbols in which case, the output distributions are discrete probabilities. In continuous density HMMs each observation probability distribution is represented by a mixture Gaussian density.

3.3. THE TOOLKIT

HTK is a toolkit for use in automatic speech recognition research and has been developed by the Speech Vision Robotics Group at the Cambridge University Engineering Department[11] and Entropic Ltd[12]. It is written in C-programming language and works on Unix and Windows platforms.

HTK consists of a set of standard library modules and command line tools. Each tool in HTK uses these standard library modules to communicate with the outside world. The library modules in HTK are HSHELL, HMEM, HMATH, HSIGP, HLABEL, HLM, HNET, HDICT, HVQ, HMODEL, HAUDIO, HGRAF, HUTIL, HTRAIN, HFB and HREC while the command line tools include HBUILD, HCOMPV, HCOPY, HDMAN, HEADAPT, HEREST, HHED, HINIT, HLED, HLIST, HLSTATS, HPARSE, HQUANT, HREST, HRESULTS, HSGEN, HSLAB, HSMOOTH and HVITE.

[11] http://svr-www.eng.cam.ac.uk

[12] http://www.entropic.com

Much of the functionality of HTK is built into the library modules. For instance, user input/output and interaction with the operating system is controlled by the library module HSHELL and all memory management is controlled by HMEM. HMATH provides math support and HSIGP provides signal-processing operations needed for speech analysis.

HTK requires different file types and it has different dedicated interface modules for each file type. HLABEL provides the interface for label files, HNET for networks and lattices, HDICT for dictionaries and HMODEL for HMM definitions.

All speech input and output at the waveform level is via HWAVE and at the parameterized level is via HPARM. Direct audio input is supported by HAUDIO and simple interactive graphics is provided by HGRAF. HUTIL provides a number of utility routines for manipulating HMMs while HTRAIN and HFB contain support for the various HTK training tools. HADAPT provides support for the various HTK adaptation tools. Finally, HREC contains the main recognition processing functions.

HTK tools are designed to run with a traditional command-line style interface. Each tool has a number of required and optional arguments. In addition to command line arguments, the operation of a tool can be controlled by parameters stored in *configuration files*. The main use of configuration files is to control the detailed behavior of the library modules on which all HTK tools depend.

In building HMM-based speech recognizers using HTK, the tools that may be used in the process can be categorized into 4 main classes; namely, *data preparation tools, training tools, testing tools* and *analysis tools.*

3.3.1. Data Preparation Tools

In the process of building HMM-based speech recognizers, a set of speech data files and their associated transcriptions are required. The tool HSLAB is used to record the speech data and to manually annotate it with any required transcriptions. HSLAB is the only tool in the HTK Package, which makes use of the graphics library HGRAF.

HSLAB is invoked by typing the command line:

HSLAB [options] dataFile

where *dataFile* is the name of the speech file. If the given speech file does not exist, then HSLAB will assume that a new file is to be recorded with this name. For instance, the command

HSLAB trial

will cause the window in the figure below to appear with a waveform display area in the upper half and a row of buttons, including a record button in the lower half.

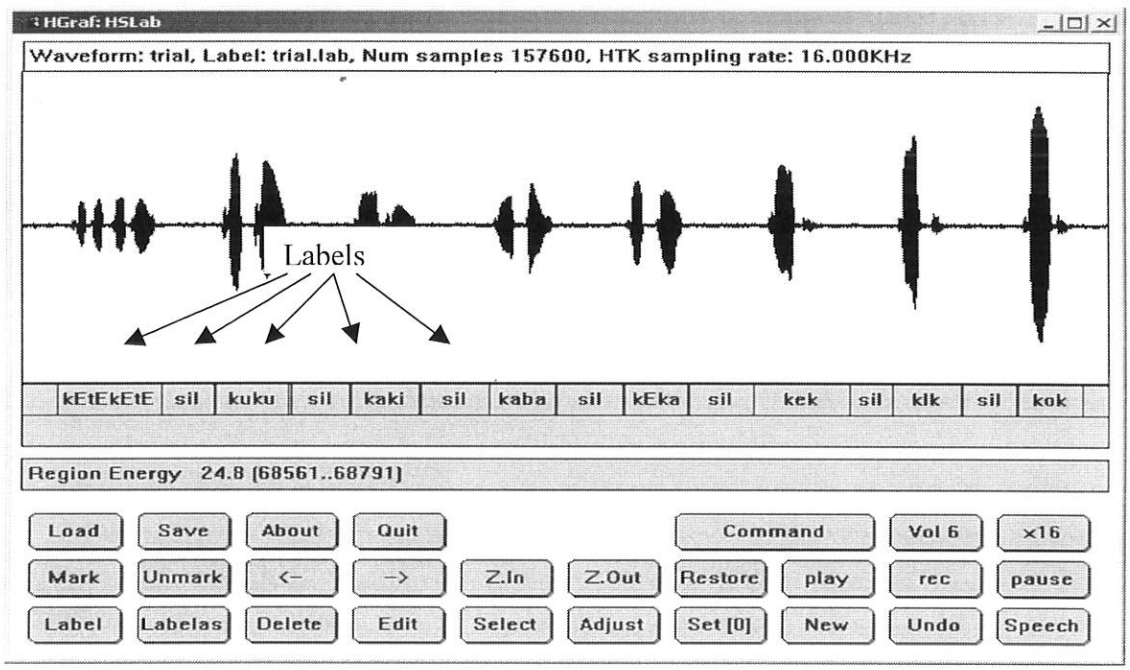

Figure 3.3. The Recording and Labeling Environment

Once the speech data is recorded, it must be converted into the appropriate *parametric* form. The tool HCOPY is used for this purpose. This program will read one or more data files to a designated output file converting the data into a parameterized form. By setting the appropriate configuration variables, all input files can be converted to parametric form as they are read in. Thus, copying each file in this manner performs the required encoding.

The general form of invocation for HCOPY is:

HCOPY src tgt

which will make a new copy called *tgt* of the file *src*. HCOPY can also concatenate several sources together as in

HCOPY src1 + src2 + src3 tgt

When coding a large database, the separate invocation of HCOPY for each file would incur a very large overhead[13]. Hence, it is better to create a file containing a list of all source files and target files.

Most of the operations performed by HTK assume that the speech data is divided into segments and each segment has a name or a *label*. The set of labels associated with a speech file constitute a *transcription* and each transcription is stored in a separate file. These transcription files should be prepared. The tool HLED is an editor for manipulating label files. It works by reading in a list of editing commands from an edit script file and then makes an edited copy of one or more label files.

[13] In HTK, tools which require potentially very long list of files always allow the files to be specified in a script file via the –S option instead of via the command line. This is particularly useful when running under the OS with limited file name expansion capability.

HLED is invoked by typing the command line

HLED [options] cmdFile labFiles. . .

This causes HLED to be applied to each *labFile* in turn using the edit commands listed in *cmdFile*. The *labFiles* may be *master label files*[14].

3.3.2. Training Tools

The next step of system building using HTK is to define the *topology* or *layout* of the Hidden Markov Models by writing a *prototype definition*. The purpose of the prototype definitions is to specify the overall characteristics and blueprint of the HMM and the actual parameters will be computed later by the training tools. Though sensible values for the transition probabilities must be given, the training process is very insensitive to these. An acceptable simple strategy for specifying sensible values for these probabilities is to make all of the transitions out of any state equally likely.

Once the structure and overall form of a set of HMMs is defined, the next step is to estimate the parameters of the HMMs from examples of the data sequences. This process of parameter estimation is usually called *training*.

The basic operation of the HTK training tools involves reading in a set of one or more HMM definitions, and estimate the parameters of these definitions using the speech data. The speech data are normally stored in parameterized form such as LPC or MFCC parameters.

[14] A Master Label File (MLF) is a single file containing a complete set of transcriptions.

In order to perform the required parameter estimation, HTK supplies four basic tools; namely, HCOMPV, HINIT, HREST, and HEREST. Which tool to use depends on whether the HMM models to be used are *whole-word based* or *sub-word based* .

To build whole-word based HMMs, the most common approach is to calculate initial parameters for the model using HINIT and then use HREST to refine the parameters using *Baum-Welch re-estimation* algorithm.

For sub-word based models, a different approach called sub-word modeling is pursued. Sub-word modeling refers to a technique whereby one HMM is constructed for each sub-word unit (Rabiner and Juang, 1993: 451).

The core process in sub-word modeling involves the embedded training[15] tool HEREST. HEREST uses continuously spoken utterances as its source of training data and simultaneously re-estimates the complete set of sub-word HMMs. For each input utterance, HEREST needs a transcription i.e. a list of the sub-word units in that utterance. HEREST then joins together all of the sub-word HMMs corresponding to this transcription to make a single composite HMM for each utterance.

HEREST is invoked via the command line:

HEREST [Options] hmmList trainFile ...

This causes the set of HMMs given in *hmmList* to be loaded. The given list of training files is then used to perform one re-estimation cycle. As always, the list of training files can be stored in

[15] *Embedded training uses Baum-Welch re-estimation algorithm and all models are trained in parallel*

a *script file* if required. On completion, HEREST outputs new updated versions of each HMM definition.

In order to use HEREST, it is first necessary to construct a file containing a list of all HMMs in the model set with each model name being written on a separate line. The names of the models in this list must correspond to the labels used in the transcriptions and there must be a corresponding model for every distinct transcription label.

However, before training the recognizer using HEREST, some pre-processing is required. The sub-word models must be initialized and one initialization strategy is to make all models equal initially and move to *embedded training*. This is called *flat-start training*. The idea behind flat-start training is to calculate the *global data mean and covariance* and assign these to all component *means* and *covariances* as start up values. This is done by the tool HCOMPV.

HCOMPV is invoked via the command line:

HCOMPV [options] hmm trainFiles

where *hmm* is the name of the physical HMM whose parameters are to be initialized. The effect of this command is to compute the covariance of the speech training data and then copy it into every *Gaussian component* of the given HMM definition. As always, list of training files can be stored in a script file, if required.

The sub-word based training process can be depicted schematically as follows:

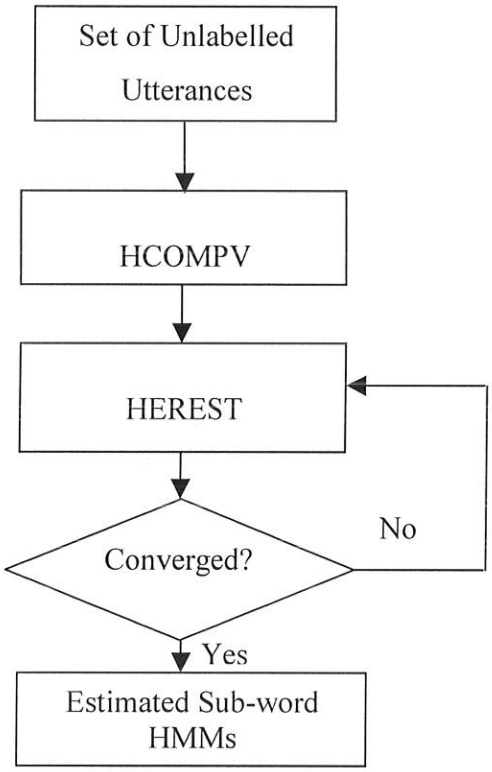

Figure 3.4. Sub-word based Training Strategy

3.3.3. Recognition Tools

HTK provides a single recognition tool called HVITE. HVITE is a general purpose Viterbi word recognizer. It will match a speech file against a network of HMMs and outputs a transcription for each of the recognized utterances.

HVITE is invoked via the command line.

HVITE [options] dictFile hmmList testFiles....

where *hmmList* should contain a list of the models required to construct the network from the word level representation. If you wish the recognition to be performed from direct audio *testFiles*

need not be specified and the configuration file (*config*) must be modified to include configuration variables that specify how the waveform is to be captured and how the captured waveform is to be converted to parameterized form.

HVITE takes as input a *network* describing the allowable word sequences, a *dictionary* defining how each word is pronounced and a set of *HMMs*. It operates by converting the word network to a sub-word network and then attaching the appropriate HMM to the sub-word units. Recognition can then be performed on either a list of stored speech files or on direct audio input.

The word networks needed to drive HVITE are usually either single word loops in which any word can follow any other word or they are directed graphs representing a finite-state task grammar.

3.3.4. Analysis Tool

Once the HMM-based recognizer has been built, it is necessary to evaluate its performance. This is usually done by using it to transcribe some pre-recorded test sentences and match the recognizer output with the correct reference transcriptions. This comparison is performed by a tool called HRESULTs which uses dynamic programming to align the two transcriptions and then count *substitution*, *deletion* and *insertion* errors. HRESULTS can also provide speaker-by-speaker breakdowns for speaker-independent models.

HRESULTS is invoked by typing the command line:

HRESULTS [options] hmmList recFiles....

This causes HRESULTs to be applied to each *recFile* in turn. The *hmmList* should contain a list of all model names for which result information is required. For each *recFile*, a transcription file with the same name but the extension *.lab* is read in and matched with it. The *recFiles* may be Master Label Files (MLFs).

When HRESULTS is invoked an output of the following form will appear:

```
----------------------------------Overall Results-----------------------------------------
SENT:        %Correct =94.00   [H=47, S=6, N=50]
WORD:        %Correct= 94.00, Acc=  94.00 [H=44,D=0, S=6,I=0, N=50]
=============================================================================
```

The first line gives the sentence-level accuracy based on the total number of label files which are identical to the transcription files. The second line is the word level accuracy based on the matches between the label files and the transcriptions. In the second line, H is the number of correct labels (Hits), D is the number of deletions, S is the number of substitutions, I is the number of insertions and N is the total number of utterances. The percentage number of labels correctly recognized is given by

$$\%\text{Correct}= \frac{H}{N} x100\%$$

and the accuracy is computed by:

$$\%\text{Accuracy} = \frac{H-I}{N} x100\%$$

It is often useful to visually inspect the recognition errors. Setting the –t option causes aligned test and reference transcriptions to be output for all labels containing errors. A typical output might be:

```
Aligned transcription: tsI0104_1.lab vs tsI0104_1.rec
```

```
 LAB: bIrr
 REC: bIl
Aligned transcription: tsI0107_0.lab vs. tsI0107_0.rec
 LAB: tErEsu
 REC: tErEtu
Aligned transcription: tsI0115_1.lab vs. tsI0115_1.rec
 LAB: sEw
 REC: sEwu
Aligned transcription: tsI0122_1.lab vs. tsI0122_1.rec
 LAB: awo
 REC: alu
====================== HTK Results Analysis===============
  Date: Wed Jun 12 03:35:37 2002
  Ref : testrefI.mlf
  Rec : recout1sp1.mlf
------------------------ Overall Results -----------------
SENT: %Correct=92.00 [H=46, S=4, N=50]
WORD: %Corr=92.00, Acc=92.00 [H=46, D=0, S=4, I=0, N=50]
============================================================
```

CHAPTER FOUR

EXPERIMENTATION

4.1. INTRODUCTION

In this experiment, *context-independent* phones, *context-dependent* triphones and Amharic CV-syllable[16] have been used as basic units of recognition to build phoneme-based, triphone-based and CV-syllable based recognizers, respectively. The suitability of these sub-word units as basic units of recognition has been measured based on the recognition performance of the respective recognizers.

The recognizers developed in this paper are intended to recognize isolated Amharic words. Because the envisaged systems are large-vocabulary recognition systems, sub-word based approach has been adopted, where each HMM represents a sub-word unit. They are sub-word based so that adding new words to the vocabulary involves only modifying the *pronunciation dictionaries* and the *task grammar.*

Since there does not exist a reference database or corpus already developed for Amharic, the experiment had to go to the extent of preparing and recording training as well as testing data using the HTK tool HSLAB. The first part of this experiment deals with *speaker-dependent* models and the second part deals with *speaker-independent* models. The design of the experiment is the same for both systems but the number of speakers required for the latter system is relatively large.

---

[16] Amharic CV-syllables are preferred to the other syllable types of the language for reasons discussed in section 2.3.

The design for both systems can be divided into two major stages. Firstly, an HMM model is trained for each sub-word unit using a number of examples words containing the sub-word units. Secondly, to recognize some unknown word, the likelihood of a sequence of models generating that word is calculated and the most likely sequence of models identifies the word.

In the presentation here, the tasks undertaken, the tools used, and the objectives of the individual steps are briefly described. As has been noted in the previous chapter, the process of developing a recognizer can be seen as a four-phase project where each phase consists of one of more sub-tasks. The presentation here is chronological (Young *et al.*, 2000:21) and each phase is described in enough detail.

The major source for this chapter is Young *et al.* (2000) and mention is made when the sources differ from this.

4.2. THE EXPERIMENT

4.2.1. Data Preparation

The first stage of any recognizer development project is *data preparation*. In the experiment here, both the training data set and the test data set have been prepared. Since the application requires that arbitrary words composed of the trained sub-word units can be added to the recognizer, effort has been made to prepare training data with "fairly good" phonetic balance and coverage.

In the experiment 84 Amharic consonants, the seven vowels of the language and thirteen 6^{th} order vowel-less consonants (a total of 104 units) have been considered for the CV-syllable based approach. These CV-syllables are formed by some combination of 20 phonemes and the seven vowels. These 20 phonemes are considered (out of the 39) for the phoneme-based approach. The

main criterion used for the selection of these units is their frequency of occurrence in normal Amharic speech and text (Baye, 1997). Amharic example words composed of these units have been selected in such a way that the contextual variations of pronunciations are accounted for. These words and the 84 context-independent CV-syllables were used to train the models. The inventory of these Amharic sub-word units and the corresponding example words are attached as Appendix C.

Two sets of test data have been prepared to test the performance of the recognizer. In order to evaluate the performance of a recognizer, it is important to see how well the system works with the training set. Hence, testsetI is prepared to include words randomly picked from the training set. In fact, in sub-word based speech recognition systems, it is also important to see how well the system works with data that it has not been trained with. Thus, testsetII consists of words that are not included in the training set. These test data are attached as Appendix D.

Before the required training and testing speech data can be recorded, it is required to construct a *pronunciation dictionary* and define a *task grammar.*

4.2.1.2. The Pronunciation Dictionary

A pronunciation dictionary is a file that contains all words that may be recognized and the way they are pronounced in a sorted order. It describes the sequence of HMMs that constitute each word.

Pronunciation dictionary is a valuable resource in the process of building a sub-word based recognizer. However, if it is produced manually, it can require considerable investment. For English and other technologically favored languages, commercial and public domain dictionaries

are available. This experiment is at a disadvantage in this respect and the pronunciation dictionary is constructed manually and letter-by-letter spelling of each word is used as its pronunciation. Since two types of sub-word units are needed to build the models, two dictionaries one based on phonemes and the other based on CV-syllables are required. Some portions of these pronunciation dictionaries are attached as Appendix E.

The dictionary for use in HTK has a very simply format. Each line consists of a single word and the corresponding pronunciation. The general format of each dictionary entry is:

WORD [output] *p1, p2, p3 . . .*

which means that the word *WORD* is pronounced as the sequence of sub-word units p1,p2,p3,... and each sub-word unit is represented by an HMM. These sequence of models are used in recognizing the words. The string in square brackets specifies the string to output when that word is recognized. If it is omitted then the word itself is output. If it is included but empty, then nothing is output as in the case of SILENCE where the entry is:

SILENCE [] sil

to mean the entry SILENCE has *sil* as its pronunciation and null output symbol.

4.2.1.3. The Task Grammar or a Word Network

A *word network* describes the sequence of words that can be recognized by the recognizer. A word level network will typically represent either a *Task Grammar* which defines all of the legal word sequences explicitly or a *word loop* which simply puts all words of the vocabulary in a loop and therefore allows any word to follow any other word. In this experiment, it is assumed that

words are independent and equally probable; there is no grammar, whatsoever, that decides word sequences.

A word network is defined using HTK Standard Lattice Format (SLF). An SLF word network is just a text file and it can be written directly with a text editor or a tool can be used to build it.

In order to build the word network, the task grammar should be written first using a higher-level grammar notation. This notation is based on the *Extended Backus Naur Form (EBNF)* that is used in compiler specification. The tool HPARSE is supplied to convert this notation into the equivalent word network.

In this undertaking, since simple isolated word recognizer is required, the allowable sequence of models now consist of an optional silence followed by a word that is followed by another optional silence. The task grammar definition used in this experiment is attached as Appendix F.

Once the task grammar is defined, a network is generated by the command:

HPARSE grammar network

where *grammar* is the name of the file containing the above grammar definition and *network* is the file name for the SLF format network generated. In this file each word instance and each word-to-word transition is listed explicitly and it can be represented pictorially as in Figure 4.1. A portion of the SLF file generated in this experiment is attached as Appendix G.

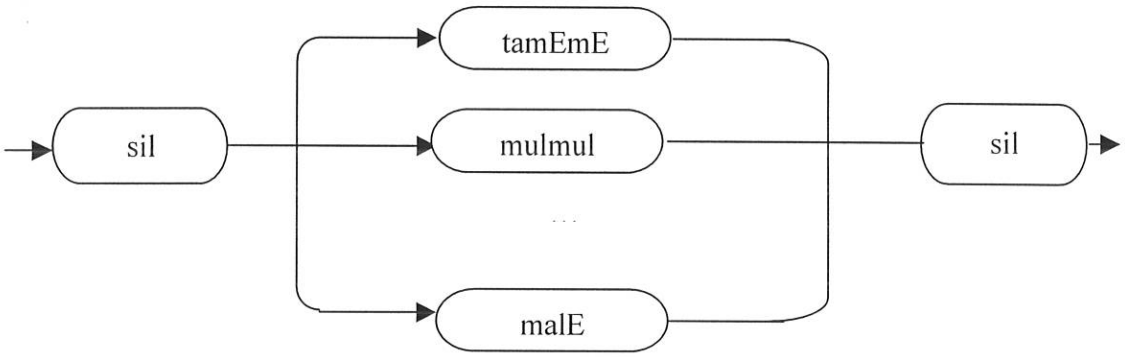

Figure 4.1. Word network for Isolated word recognition system

4.2.1.4. Recording the Data

Once the data are prepared and the pronunciation dictionary as well as the task grammar are built, the next step is to record (and label) the data using the HTK tool HSLAB. *Labeling* is the process by which each segment of an utterance is assigned a label or a name (Reddy,1976) and doing so by hand for the recorded speech data is, obviously, not practical. Therefore, this experiment has started with unlabelled set of utterances adopting what is known as the *flat-start* scheme.

Speech samples of 15 speakers (8 male and 7 female) using the training set and the test sets have been recorded. Besides, five more speakers (3 female and 2 male) who have not taken part in training the models have also been recorded using the test sets.

4.2.1.5 Creating the Transcription Files

With or without labeled data, HTK requires the transcription of each speech data be prepared and stored in a separate file.

In training an HMM system it is useful to have both the word level transcriptions and the sub-word unit level transcriptions side-by-side. HTK training tools typically expect the labels used in

transcription files to correspond directly to the names of the HMMs chosen to build the application.

The starting point for both sets of sub-word units transcription is an orthographic transcription in HTK label format. The complete set of the orthographic transcription is stored in a single file called *Master Label File (MLF). Master Label Files (MLFs)* may be regarded as index files holding pointers to the actual label files which can either be embedded in the same index file or stored elsewhere in the file system. The Master Label file built in this experiment has the following format.

```
#!MLF!#
"tr0101_0.lab"
tamEmE
sil
mulmul
sil
mimi
sil
Imama
sil
Itete
sil
Imete
sil
tarIme
sil
etc.
.

"tr0102_0.lab"
tEnEtEnE
sil
tErEtu
sil
etc.
```

Once the word level MLF has been created, the phone level MLF and the CV-syllable level MLF can be generated using the label editor HLED. Assuming that the above word level transcription file is named *master*, the command

HLED -d dict1 -i phones.mlf mkphones.led master.mlf

generates a phone level transcription of the following form and stores the output in the file called *phones.mlf.*

```
#!MLF!#
"tr0101_0.lab"
sil
t
a
m
E
m
E
sil
m
u
l
m
u
l
sil
m
i
m
i
etc.
```

and the command :

HLED -d dict2 -i syllables.mlf mkphones0.led master.mlf

generates a syllable level transcription of the following form and stores it in a file called syllables.mlf.

```
#!MLF!#
"tr0101_0.lab"
sil
ta
mE
mE
sil
mu
l
mu
l
sil
etc.
```

The HLED edit script *mkphones0.led* contains the following commands

```
EX
IS sil sil
DE sp
```

The EX command replaces each word in *master.mlf* by the corresponding pronunciation in the dictionary file *dict1* or *dict2*. The IS command inserts a silence model *sil* at the start and end of every utterance. Finally, the delete DE command deletes all short pause *sp* labels, which are not wanted in the transcription labels at this point.

4.2.1.6. Coding the Data

Most digital representation methods are based on the assumption that parameters of speech remain unchanged over a short-time period (Reddy, 1976). In other words, the fundamental assumption underlying any short-time analysis method is that over a sufficiently short time interval speech can be considered stationary (Davis and Mermeistein, 1980).

Parametric representations may be divided into two groups:

- Those based on the Fast Fourier Transform spectrum (FFT), and
- Those based on Linear Prediction Spectrum (LPC)

The commonly used short time spectrum can be obtained using the Fast Fourier Transform method. Though HTK supports both FFT-based and LPC-based analysis, Mel Frequency Cepstral Coefficients (MFCCs) which are derived from FFT-based log spectra, consisting of the length of the parameterized static vector (+13), the delta coefficients (+13) and the acceleration coefficients (+13) which add up to 39 are used in this experiment.

In order to code the data, a configuration file (*config*) is created which specifies all of the conversion parameters. The following is the setting used in this experiment.

```
# Coding parameters
TARGETKIND=MFCC-0-D-A
TARGETRATE=100000.0
SAVECOMPRESSED=T
SAVEWITHCRC=T
WINDOWSIZE =250000.0
USEHAMMING = T
PREEMCOEF =0.97
NUMCHANS=26
CEPLIFTER =22
NUMCEPS =12
ENORMALISE =F
```

The settings in the configuration file specify that the target parameters are to be MFCC, the frame period is 10msec (HTK uses units of 100ns), the output should be saved in compressed format, a CRC checksum should be added, and so on.

Coding can be performed using the tool HCOPY. A script file containing the list of each source file and its corresponding output file is supplied to the HCOPY command as an argument. The first few lines of this script file look like:

```
tr0101_0        tr0101_0.mfc
tr0101_1        tr0101_1.mfc
tr0102_0        tr0102_0.mfc
tr0102_1        tr0102_1.mfc
etc.
```

Given that the above script is stored in the file *codetr.scp*, the training data would be coded by executing:

HCOPY -T 1 -C config -S codetr.scp

A similar procedure has been used to code the test data after which all of the pieces have been put in place to start training the HMMs.

4.3. TRAINING OF SUB-WORD UNITS

If the speech data is labeled in such a way that the location of the sub-word unit (i.e. phone or CV-syllable) boundaries have been marked, then this can be used as *bootstrap* data. When no bootstrap data is available, as in the current experiment, one good strategy is to make all models equal initially and move to embedded training using HEREST.

4.3.1. Creating Flat Start Sub-word Units

As stated in the previous chapter, the first step in HMM training is to define a *prototype* model. The parameters of this model are not really important since the purpose of the prototype definition is only to define the blue print or the layout of the *model* and the actual parameters will be re-estimated later.

As stated by Young *et al.* (2000), a good topology for phone-based system is 3-state left-right topology. The same 3-state left-right model was tested for the CV-syllable based recognizer and resulted in a severely degraded performance. Consequently, an 8-state left-right topology has been used for each CV-syllable with two non-emitting states. In fact, other number of states were also tested for CV-syllable based recognizer and better recognition results are obtained at n=10, where n is the number of states.

In an HTK-based HMMs, since the entry and exit states are non-emitting they have no output probability distributions associated with them. Due to this fact, the model used in the phone-based recognizer is considered as a 3-state model (while the actual number of states used is 5) and that used in the CV syllable-based recognizer is considered as an 8-state model (while the actual number of states used is 10).

The topologies defined for the phone-based system (protoI) and the CV-syllable based system (protoII) are attached as Appendix H.

During the first cycle of embedded re-estimation, each training utterance will be uniformly segmented with the hope that enough of the sub-word models align with actual realizations of the units so that on the second and subsequent iterations, the models align as intended.

The starting point for embedded re-estimation is a set of identical *monophones* or *monosyllables* in which every mean and variance is identical. The HTK tool HCOMPV is used to compute the global mean and variance of the training set. Given that a list of all the training files for the first speaker, for instance, is stored in the script file *train1.scp* (Appendix I), the command

HCOMPV -C config -f 0.01 -m -S train1.scp -M hmm0 protoI

creates a new version of *protoI* in the directory *hmm0* for the phone models in which the zero means and unit variances have been replaced by the global speech means and variances. The –f option causes a variance floor macro (called vFloors) to be generated which is equal to 0.01 times the global variance. This is a vector of values which will be used to set a floor on the variances estimated in the subsequent steps. The –m option asks for means to be computed as well as variances.

Given this new prototype model stored in the directory *hmm0*, a Master Macro File[17] (MMF) called *hmmdefs* containing a copy for each of the required sub-word unit HMMs was constructed by manually copying the newly generated prototype and re-labeling it for each required monophone or monosyllable. The file containing the definitions for the monophones is stored as

[17] Master Macro File (MMF) is similar to that of a Master Label File and it serves a similar purpose in that it avoids having a large number of individual HMM definition files.

hmmdefssp1 and the file containing the definitions for the monosyllables is stored as *hmmdefssylX* in directory hmm0 for the first speaker.

The flat-start sub-word units for the first speaker that are stored in the directory hmm0 are next re-estimated using the embedded re-estimation tool HEREST invoked as:

HEREST -C config -I phones0.mlf -S train1.scp -H hmm0/macrossp1 -H hmm0/hmmdefssp1 -M hmm1 monophones0

for the phone models and

HEREST -C config -I syllables0.mlf -S train1.scp -H hmm0/macrossylX -H hmm0/hmmdefssylX -M hmm1 monosyllables0

for the CV-syllable models where *monosyllables0* and *monophones0* are files consisting of phones and CV-syllables for each of which an HMM is built.

Embedded training using HEREST simultaneously updates all of the HMMs in a system using all of the training data. Each time HEREST is run, it performs a single *Baum-Welch* re-estimation of the whole set of HMM phone models or CV-syllable models simultaneously. For each training utterance, the corresponding sub-word units are concatenated and then the forward-backward algorithm is used to accumulate the statistics of state occupation, means, variance, etc., for each HMM in the sequence.

HEREST was run twice more. After each run of HEREST, each new HMM set is stored in a new directory, changing the name of the input and output directories (set with the options -H and -M) each time, until the directory *hmm3* contains the final set of initialized sub-word unit HMMs.

4.3.2. The Silence Models

Speech recognition systems will often distinct models for silence and short pauses. The silence model *sil* may have the normal 3-state topology whereas a short pause model *sp* may have just a single state since, as Rabiner (1993:441) pointed out, sp is generally stationary and has no temporal structure to exploit.

To make the model more robust by allowing individual states to absorb the various impulsive noises in the training data, extra transitions from states 2 to 4 and from states 4 to 2 are added in the silence model.

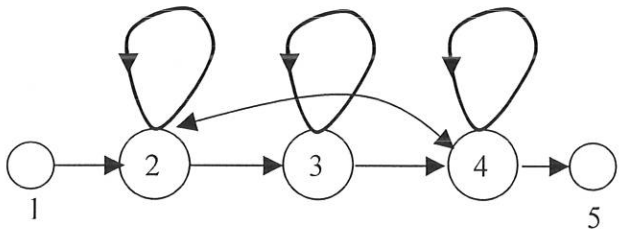

Figure 4.2. The Silence Model

A single state short pause sp model is created and to avoid the *sil* and *sp* models *competing* with each other, the sp model state can be tied to the center state of the sil model. The required topology for the sp-model is shown in the figure below.

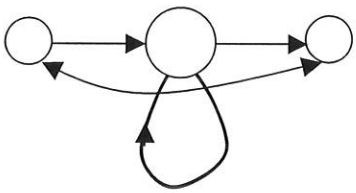

Figure 4.3. The sp Model

These silence models were created in two stages:

1. Open the file hmm3/hmmdefssp1 for the first speaker and copy the center state of the sil model to make a new sp model and store the resulting MMF hmmdefs, which includes the new sp model, in the new directory
2. Run the HMM editor HHED to add the extra transitions and tie the sp state to the center sil state as follows:

   HHED -H hmm4/macrosI -H hmm4/hmmdefsI -M hmm5 sil.hed monophones1

where sil.hed contains the following commands

```
AT    2    4    0.2    {sil.transP}
AT    4    2    0.2    {sil.transP}
AT    1    3    0.3    {sp.transP}
TI    silst    {sil.state[3], sp.state[2]}
```

The AT commands add transitions to the given transition matrices and the final TI command creates a tied-state called *silst*. The parameters of this tied state are stored in the hmmdefs file and within each silence model, the original state parameters are replaced.

The monophones and monosyllables list now include the sp model and have been saved as *monophones1* and *monosyllables1*.

The context-independent phone models could be directly used in recognition, or they could be used to initialize the training of the context-dependent phone models (Lee, 1990). In this experiment, both the context independent phone models and tied-state triphone models have been used for recognition.

To use the context-independent phone models for recognition, another 13 sequence of HEREST iterations were applied using the phone transcriptions with silence model after each word. The same is done for CV-syllable models. This leaves the set of monophone HMMs and the set of monosyllable HMMs created in the directory hmm15.

When performing embedded training, it is good to monitor the performance of the models and stop training when no further improvement is obtained. Repeated re-estimation may take an impossibly long time. Worse still, it can lead to *over training* since the models can become too closely matched to the training data and fail to generalize well on unseen test data.

For the phone models, it was not necessary to go for another iteration using HEREST because, the models nearly converged after the 15^{th} iteration and the HMM definitions in directory 15 have been used for recognition. For the CV-syllable based approach, five more iterations made the models converge and the HMM definitions (hmmdefs) in directory 20 have been used for recognition.

To create tied-state triphone models, all that has been done for the phone models were done up to the 10^{th} iteration except that a different name is used for the HMM definitions (for the first speaker for instance, hmmdefspdtiedsp1). The models in hmm10 are used to initialize the training of the context dependent tied-state triphone models.

4.3.3. Creating Tied-State Triphones

One of the major problems to be faced in building any HMM based system is that of data insufficiency. To overcome this problem, HTK allows a variety of sharing mechanisms to be

implemented whereby HMM parameters are tied together so that the training data is shared and more robust estimates result.

This involves converting the set of initialized and trained context independent monophone HMMs to a set of context dependent models. In this experiment, this has been done in two steps. Firstly, the monophone transcriptions have been converted to triphone transcriptions and a set of triphone models have been created by copying the monophones and re-estimating.

4.3.3.1. Triphone Construction

The triphone transcription has been created automatically using HLED. As a result of the following invocation, the monophone transcriptions have been converted to an equivalent set of triphone transcriptions and are stored in a file named *wintri.mlf*. At the same time, a list of triphones is written to the file named *triphones*

HLED -n triphones -i wintri.mlf mktri.led phonesII.mlf

The edit script *mktri.led* contains the commands:

```
WB sp
WB sil
TC
```

The two WB commands define sp and sil as word boundary symbols.

The clone command CL in the script file below takes as its argument the name of the file containing the list of triphones (and biphones, for that matter) and for each model of the form a-b+c in this list, it looks for the monophone b and makes a copy of it. The TI command is used to explicitly tie all members of a set of transition matrices together. The list of items within brackets are patterns designed to match the set of triphones, right biphones and left biphones for each phone.

The following invocation performs the cloning of the models efficiently:

HHEd -H hmm9/macrosIII -H hmm9/hmmdefsIII -M hmm10 mktri.hed monophones

where mktri.hed is the following script file:

```
CL triphones
TI T_a {(*-a+*, a+*, *-a).transP}
TI T_b {(*-b+*, b+*, *-b).transP}
TI T_d {(*-d+*, d+*, *-d).transP}
TI T_e {(*-e+*, e+*, *-e).transP}
TI T_E {(*-E+*, E+*, *-E).transP}
TI T_g {(*-g+*, g+*, *-g).transP}
TI T_i {(*-i+*, i+*, *-i).transP}
TI T_I {(*-I+*, I+*, *-I).transP}
TI T_k {(*-k+*, k+*, *-k).transP}
TI T_l {(*-l+*, l+*, *-l).transP}
TI T_m {(*-m+*, m+*, *-m).transP}
TI T_n {(*-n+*, n+*, *-n).transP}
TI T_o {(*-o+*, o+*, *-o).transP}
TI T_r {(*-r+*, r+*, *-r).transP}
TI T_rr {(*-rr+*, rr+*, *-rr).transP}
TI T_s {(*-s+*, s+*, *-s).transP}
TI T_t {(*-t+*, t+*, *-t).transP}
TI T_u {(*-u+*, u+*, *-u).transP}
TI T_w {(*-w+*, w+*, *-w).transP}
TI T_y {(*-y+*,y+*,*-y)transP}
```

The following is an example from the file *wintri.mlf* which consists of a set of triphone transcriptions:

```
#!MLF!#
"tr0101_0.lab"
sil
t+a
t-a+m
a-m+E
m-E+m
E-m+E
m-E
sil
m+u
m-u+l
u-l+m
l-m+u
m-u+l
etc.
```

Once the context-dependent models have been cloned, the new triphone set can be re-estimated using HEREST. This is done as previously except that the monophone model list is replaced by a triphone list and the triphone transcriptions are used in place of the monophone transcriptions as in the following:

HEREST –C config –I wintri.mlf –S train.scp –H hmm10/macrosIII –H hmm10/hmmdefsIII –M hmm11 triphones

One more re-estimation was performed so that the resultant model sets were ultimately saved in hmm12.

4.3.3.2. Making Tied-State Triphones

As discussed in the previous section, the first stage of model refinement is usually to convert a set of initialized and trained context independent monophone HMMs to a set of triphone models. The next step in the model building process is to tie states within triphone sets in order to share data and thus to be able to make robust parameter estimates.

The idea of *parameter tieing* is appropriate in the case where there is insufficient training data to estimate a large number of model parameters reliably (Rabiner, 1989).

For the purpose of parameter tieing, HHED provides a mechanism that uses decision trees and is based on asking questions about the left and right contexts of each triphone. The decision tree attempts to find those contexts which make the largest difference to the acoustics and which should therefore distinguish clusters.

Decision tree state tieing is performed by running HHED in the normal way, i.e.

HHED -H hmm12/macrosspdtiedsp1 -H hmm12/hmmdefsspdtiedsp1 -M hmm13 tree.hed triphones>log

The edit script *tree.hed*, which contains the instructions regarding which contexts to examine for possible clustering can be rather long and complex. However, it has been tried to generate this script file manually to test for the effects of tieing. The *tree.hed* file used in this experiment is attached as Appendix J.

Each QS command loads a single question and each question is defined by a set of contexts. For example, the first QS command defines a question called L-Class-Stop which is true if the left context is either of the stops ብ /b/, ት /t/, ድ /d/, ክ /k/, or ግ /g/.

For a triphone system, it is necessary to include questions referring to both the right and left contexts of a phone, the questions should progress from wide, general classifications (such as consonant, vowel, nasal, diphthong, etc.) to specific instances of each phone.

The second TR command enables intermediate level progress reporting so that each of the following TB commands can be monitored. Each of these TB commands cluster one specific set of states. For example, the first TB command applies to the first emitting state of all context-dependent models for the phone *a*.

The values given in the RO (100, in this experiment) which is used to set the outlier threshold and TB (375 in this case) commands affect the degree of tieing and therefore the number of states

produced in the clustered system. The values should be varied according to the amount of training data available.

An important advantage of tree-based clustering is that it allows triphone models which have no training data to be synthesized. This is done HHED using the AU command which has the form

AU *hmmlist*

Its effect is to take as its argument a new list of triphones (*hmmlist*) expanded to include all those needed for recognition and any physical models listed which are not in the currently loaded set are synthesized.

Once all state-tieing has been completed and new models synthesized, some models may share exactly the same 3-states and transition matrices and are thus identical. The CO command is used to compact the model set by finding all identical models and tieing them together, producing a new list of models called *tiedlist*.

Finally and for the last time, the models have been re-estimated twice using HEREST changing the name of the input and output directories (set with the options -H and -M) each time HEREST is run. The first of the two invocations is as follows:

HEREST –C config –i wintri.mlf –S train.scp –H hmm13/macrosspdtiedsp1 –H hmm13/hmmdefsspdtiedsp1 –M hmm14 tiedlist

Now that the models have been sufficiently trained, the next step is to evaluate the performance of the recognizer.

4.4. SPEAKER-INDEPENDENT MODELS

The same design principles have been followed in order to build the speaker-independent models. The only and the major difference is it requires relatively large number of speakers. In order to build this model, 20 speakers (10 female speakers and 10 male speakers) in the same age group (20-30) have been used. Moreover, 15 of them have been involved in the training process and 5 of them (3 female and 2 male) have not been involved in the training process and used to test the speaker-independent models for previously unknown users.

4.5. RECOGNIZER EVALUATION

Both test sets; namely, testsetI (also called training data) and testsetII (also called testing data) consist of 50 discrete words each and are used to test the performance of the models. Since the recognition network and the dictionary have already been constructed and the test data have already been recorded., the tool HVITE has directly been used to perform the recognition task.

4.5.1. Recognition

To run the recognizer for the speaker-dependent models the tool HVITE is used. The invocations for the first speaker, for instance, are given below:

For the phone models, on the training data:

HVITE -H hmm15/macrosspl -H hmm15/hmmdefsspl -S testsetI.scp -i recoutlspl.mlf -w network dictl monophonesl

For the phone models, on the testing data:

HVITE -H hmm15/macrosspl -H hmm15/hmmdefsspl -S testsetII.scp -i recout2spl.mlf -w network dictl monophonesl

For the CV syllable based models, on the training data:

HVITE -H hmm20/macrossylX -H hmm20/hmmdefssylX -S testsetI.scp -i recout1syl.mlf -w network dict2 monosyllables1

For the CV syllable based models, on the testing data:

HVITE -H hmm20/macrossylX -H hmm20/hmmdefssylX -S testsetII.scp -i recout2syl.mlf -w network dict2 monosyllables1

For the tied-state triphone models, on the training data:

HVITE -H hmm15/macrospdtiedsp1 -H hmm15/hmmdefspdtiedsp1 -S testsetI.scp -i recoutspd1sp1.mlf -w network dict1 tiedlist

For the tied-state triphone models, on the testing data:

HVITE -H hmm15/macrosspdtiedsp1 -H hmm15/hmmdefsspdtied -S testsetII.scp -i recoutspd2sp1.mlf -w network dict1 tiedlist

where *testsetI.scp* and *testsetII.scp* are the parameterized test files, *dict1* and *dict2* are the dictionaries for the phone models and the syllable models respectively, and network is the network for the recognition. *monophones1*, *monosyllables1* and *tiedlist* are the various sub-word unit lists consisting of phones, syllables and tied state triphones respectively.

For the speaker-independent model using monophones as basic units, the recognition command has been invoked as follows:

For phone models on the training data:

HVITE -H hmm15/macrosind -H hmm15/hmmdefsind -S codetsI.scp -i recoutindI.mlf -w network dict1 monophones1

For phone models on the testing data:

HVITE -H hmm15/macrosind -H hmm15/hmmdefsind -S codetsII.scp -i recoutindII.mlf -w network dict1 monophones1

For tied-state triphones on the training data:

HVITE –H hmm15/macrosindtied –H hmm15/hmmdefsindtied –S codetsI.scp –i recoutindItied.mlf –w network dict1 tiedlist

For tied-state triphones on the testing data:

HVITE –H hmm15/macrosindtied –H hmm15/hmmdefsindtied –S codetsII.scp –i recoutindIItied.mlf –w network dict1 tiedlist

where codetsI.scp and codetsII.scp are the two coded testing files that consist of the parameterized form of the utterances of all of the 20 speakers.

4.5.2. Analysis

The performance of the recognizer is then determined by the tool called HRESULTS. HRESULTS reads in a set of label files that are output from the recognition tool HVITE and compares them with the corresponding reference transcription files. The comparison is based on a Dynamic Programming based string alignment procedure.

The following command lines have been invoked after executing their respective recognition command:

For the phone-based recognizer:

HRESULTS –I testrefI.mlf monophones1 recout1sp1.mlf

HRESULTS –I testrefII.mlf monophones1 recout2sp1.mlf

For the CV syllable based recognizer:

HRESULTS –I testrefI.mlf monosyllables1 recout1syl.mlf

HRESULTS –I testrefII.mlf monosyllables1 recout2syl.mlf

For the tied-state triphone based recognizer:

HRESULTS –I testrefI.mlf tiedlist recoutspd1sp1.mlf

HRESULTS –I testrefII.mlf tiedlist recoutspd2sp1.mlf

Sample recognition output is attached as Appendix K.

For the speaker-independent models HRESULT is invoked as follows:

For the phone-based recognizer:

HRESULTS –k tsI%% -I testrefIind.mlf monophones1 recoutindI.mlf*

HRESULTS –k tsII%% -I testrefIIind.mlf monophones1 recoutindII.mlf*

For the tied-state triphone based recognizer:

HRESULTS –k tsI%% -I testrefIind.mlf tiedlist recout1tied.mlf*

HRESULTS –k tsII%% -I testrefII.mlf tiedlist recout2tied.mlf*

where *testrefIind.mlf* and *testrefIIind.mlf* are actual transcription files of the test sets against which the transcription files generated by the recognizer; namely, recout1tied.mlf and recout2tied.mlf are compared. The *–k s* option is used so that recognition results are displayed on a speaker by speaker basis as well as globally where *s* defines a pattern which is used to extract the speaker identifier from the test label file name. The character % matches any character whilst including it as part of the speaker identifier. The wildcard * is used to match zero or more characters.

4.5.3. Comparison of Units of Recognition

Using the researcher's own voice and two more speakers, experiments have been carried out to compare the performance of:

1. *Phoneme-based recognition;*
2. *CV-syllable-based recognition* and
3. *Tied-state triphone based recognition.*

The results obtained are presented in the table below.

Table 4.1. Comparison of Units of Recognition

Speaker Code	TestsetI (Accuracy %)			TestsetII (Accuracy %)		
	Phonemes	CV-Syllable	Tied-state Triphones	Phonemes	CV-Syllable	Tied-state Triphones
01	92	84	94	92	70	90
02	94	76	90	82	58	82
06	98	74	92	84	60	86

4.5.3.1. Speaker-dependent Models

As observed from the results of the experiment, the CV-syllable based approach has not been considered in the subsequent experiments due to its relatively poor performance. However, since phonemes and tied-state triphones produce competing results they have been considered for further investigation. For the purpose of evaluating these units in terms of recognition performance, speaker-dependent models have been developed for 15 speakers (8 male speakers and 7 female speakers) in the same age group (20-30) using phonemes and tied-state triphones as the basic units of recognition. The results obtained are presented in Table 4.2.

Table 4.2. Speaker-dependent Models

Speaker Code	TestsetI		TestsetII	
	Phoneme-based HMMs(%)	Tied-State Triphones (%)	Phoneme-based HMMs (%)	Tied-State Triphones (%)
01	92	94	92	90
02	94	90	82	82
03	80	90	86	72
04	86	92	78	82
05	86	96	78	70
06	98	92	84	86
07	88	84	84	74
08	90	86	86	78
09	78	58	80	66
10	82	84	90	92
11	88	84	94	74
12	64	72	68	66
13	82	84	82	86
14	76	76	78	70
15	88	78	84	82
Avg.	**84.80**	**84.00**	**83.07**	**78.00**

As can be seen from the table, the speaker-dependent recognition system led to a recognition accuracy from 64% to 98% with an average accuracy of 84.80% for the phone based models and from 58% to 96% with an average recognition of 84% for the tied state models on the training data. On the testing data, an average recognition accuracy of 83.07% and 78% has been obtained for the phone based models and the tied-state models respectively.

4.5.3.2. Speaker-independent Model

The following table summarizes the recognition accuracy results for the speaker-independent model using both sets of test data.

Table 4.3. Speaker-independent Model Output

Speaker Code	Sex	TestsetI		TestsetII	
		Phoneme-based HMMs (%)	Tied-State Triphones (%)	Phoneme-based HMMs (%)	Tied-State Triphones (%)
01	M	90	96	82	86
02	M	88	94	80	80
03	M	72	92	78	82
04	F	74	94	72	78
05	F	86	96	70	86
06	F	84	94	72	80
07	F	78	92	82	72
08	M	80	86	82	78
09	M	70	96	76	74
10	M	76	92	72	84
11	M	84	90	88	80
12	F	62	82	54	60
13	M	80	96	70	76
14	F	64	82	72	66
15	F	78	90	80	86
Avg.		**77.73**	**91.46**	**75.33**	**77.87**

As can be observed from Table 4.5, an average recognition accuracy of 77.73% on the training set and 75.33% on the test set have been obtained for the phone models. For tied-state triphone models an average recognition accuracy of 91.46% and 77.87% have been achieved on the training data and testing data respectively

4.5.3.3. Effect of Sex on Recognition Accuracy

The following table summarizes the distribution of recognition by sex for the speaker-dependent model.

Table 4.4. Male Speakers

Speaker Code	TestsetI		TestsetII	
	Phoneme-based HMMs (%)	Tied-State Triphones (%)	Phoneme-based HMMs (%)	Tied-State Triphones (%)
01	92	94	92	90
02	94	90	82	82
03	80	90	86	72
08	90	86	86	78
09	78	58	80	66
10	82	84	90	92
11	88	84	94	74
13	82	84	82	86
Avg.	**85.75**	**83.75**	**86.5**	**80.00**

Table 4.5. Female Speakers

Speaker Code	TestsetI		TestsetII	
	Phoneme-based HMMs (%)	Tied-State Triphones (%)	Phoneme-based HMMs (%)	Tied-State Triphones (%)
04	86	92	78	82
05	86	96	78	70
06	98	92	84	86
07	88	84	84	74
12	64	72	68	66
14	76	76	78	70
15	88	78	84	82
Avg.	**83.71**	**84.28**	**79.14**	**75.71**

As can be observed, male speakers have an average recognition accuracy of 85.75% on the training data and 86.5% on the testing data for the phone-based models. For the tied-state triphone models an average recognition accuracy of 83.75% on the training data and 80% on the testing data have been obtained for male speakers.

Female speakers have an average accuracy of 83.71% on the training set and 79.14%on the test set for the phone-based models and an average recognition accuracy of 84.28% on the training data and 75.71% on the test set for the tied-state triphone models.

For the speaker-independent model, the following table summarizes the distribution of recognition by sex.

Table 4.6. Male Speakers (Speaker-independent)

	TestsetI		TestsetII	
Speaker Code	**Phoneme-based HMMs (%)**	**Tied-Sate Triphones HMMs (%)**	**Phoneme-based HMMs (%)**	**Tied-Sate Triphones HMMs (%)**
01	82	96	90	86
02	80	94	88	80
03	78	92	72	82
08	82	86	80	78
09	76	96	70	74
10	72	92	76	84
11	88	90	84	80
13	70	96	80	76
Avg.	**78.50**	**92.75**	**80.0**	**80.0**

Table 4.7. Female Speakers (Speaker-independent)

Speaker Code	TestsetI		TestsetII	
	Phoneme-based HMMs	**Tied-Sate Triphones HMMs**	**Phoneme-based HMMs**	**Tied-Sate Triphones HMMs**
04	72	94	74	78
05	70	96	86	86
06	72	94	84	80
07	82	92	78	72
12	54	82	62	60
14	72	82	64	76
15	80	90	78	86
Avg.	**71.71**	**90.0**	**75.14**	**76.85**

As can be seen from the tables above, male speakers are better recognized than female speakers in both models. An average recognition accuracy of 78.50% vs. 71.71% on the training data and 80% vs. 75.14 on the testing data are observed for phone based models while an average accuracy of 92.75% vs. 90% on the training data and 80% vs. 76.85% on the testing data are obtained for the tied-state triphone models.

4.5.3.4. Untrained Speakers

The following table summarizes the recognition accuracy of speakers who did not involve in training the model:

Table 4.8. Untrained Speakers

Speaker Code	Testset I		Testset II	
	Phoneme-based HMMs	**Tied-Sate Triphones HMMs**	**Phoneme-based HMMs**	**Tied-Sate Triphones HMMs**
19	76%	96%	78%	76%
16	60%	76%	74%	76%
17	70%	88%	56%	54%
20	78%	94%	78%	84%
22	60%	76%	74%	52%
Avg.	**68.80%**	**86.00%**	**72.00%**	**68.40%**

As can be observed from the results above, untrained speakers have an average recognition accuracy of 68.8% for phoneme-based models on the training data and 72% on the testing data and a recognition accuracy of 86% for the tied-state triphone based models on the training set and 68.40% on testing set.

4.6. ANALYSIS OF RESULTS

As can be observed from the results of the experiments phonemes and tied-state triphones have come up with better performance. In fact, it is natural to expect tied-state triphones to produce better recognition results as it captures most of the important contextual effects. And considering the nature of Amharic language, CV-syllables could also be anticipated to produce good results. However, the CV-syllable based recognizer has led to a relatively poor performance when compared with the other units of recognition.

The main reason for this relatively degraded performance may be attributed to the toolkit used in the experiment. Since CV-syllable as a basic unit of recognition is not common for many languages, the designers of the toolkit might have not considered CV-syllables. In fact, the research had to hit upon the necessary parameters of the model by trial and error, since no optimum values are proven.

The results have shown that there is no significant difference in the performance of the phoneme-based model and tied-state triphone-based model on the training data for speaker-dependent models. On the testing data, however, about 5% performance degradation has been observed in the tied-state triphone models. The speaker-independent models, on the other hand, have resulted in a very good performance for the training data using tied-state triphone models (91.46% vs. 77.33%) but practically speaking, insignificant improvement has been obtained for the testing data (77.87% vs. 75.33%). The reasons why the results for the testing data is not as good as that of the training data is believed to be inadequate test set triphone coverage in the training set. This can be handled by training the triphones (context-dependent models) from a large database or corpus.

Another possible reason could be lack of complete and exhaustive information about the rules of Amharic. All sequences of sounds are not permissible in a language (Clark *et al.*, 1986:221) and it would be interesting to determine which clusters are possible in Amharic and which are prohibited by the rules of the language. To put it another way, it would have been useful to determine the rules of Amharic that decide which consonants may cluster together in which order, and which are prohibited. This is a task beyond the scope of this project, but the availability of these information would have helped this project during constructing the script file *tree.hed* which contains the instructions regarding the contexts to examine for possible clustering

The study admits that the rules specified in the script file *tree.hed* are by no means exhaustive and are used simply to test the effects of tieing.

It is hoped that the availability of a large database with good phonetic balance and coverage will make both models; namely, the speaker-dependent and the speaker-independent models, more robust with the tied-state triphone approach.

Analysis of recognition accuracy on a demographic variable; viz., sex showed that male speakers have a slightly better recognition accuracy (85.75% vs. 83.71%) on the training data and (86.5% vs. 79.14%) on the testing set for the phoneme-based models and (83.75% vs. 84.28 %) on the training data and (80% vs. 75.71%) on the testing set for the tied-state triphone models.

For untrained speakers tied-state triphones resulted in a very good recognition accuracy (86%) on the training data. Better results could still be anticipated for the testing set with tied-state models if the limitations discussed above are fully addressed.

The process of building the recognizers has been carried out in a laboratory where other student researchers were working, hence, interpersonal discussions and frequent door slams plus background noise were unavoidable. The background noise can affect a recognition system in two ways. In the first place, it possibly could have introduced unintended extraneous acoustic elements into the channel via the microphone. Second, speakers are not indifferent to stress and background noise (Markowitz, 1996:156). Stress produces emotional and physical responses that affect the way a person speaks. Since many of the speakers were strangers to the student researcher in the room, they were not at ease and the result of some of the speakers was very low because of the observable stress they were in during recording. Moreover, some of the speakers could not speak consistently and considerable degraded performance has been observed for these

speakers on the training data. Besides, since the system used flat-start training scheme, the automatic segmentation and labeling procedures normally make errors in addition to the fundamental ambiguity of the signals.

Comparison has also been made based on the number of models used in building the different recognizers. The phoneme-based recognizer was made up of 22 HMM models one for each phoneme including the silence and the sp model. On the other hand, the tied-state triphone-based recognizer consisted of 553-models including the silence and sp models. These context dependent triphones were generated automatically using the HTK tool HLEd. However, some extra work had to be done in order to tie states together for the purpose of sharing data using phonetic decision trees.

The CV-syllable based recognizer, on the other hand, was made up of 106 models. Once the global-mean and variance has been calculated by the HCOMPV tool, the master macro file *hmmdefssylX* has been created by manually coping the newly generated prototype and re-labeling it for each of the 106 monosyllables in the directory hmm0. This adds to the complexity of the preprocessing compared to the 22-phoneme models.

Furthermore, an attempt was made to mark the effect of gemination using double consonants as in the following example:

alE **አለ** (he said) – not geminated

allE **አለ** (he is present) – geminated

However, this degraded the performance of CV-syllable based recognizer significantly. Nonetheless, as can be seen above in Amharic both words are written with the same combination

of characters and the difference in meaning is normally derived from context. Therefore, marking the gemination is not worth the sacrifice because in the ultimate representation both words are represented with same symbols. In fact, in training the models, both sounds (geminated and not geminated) are included.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1. INTRODUCTION

The purpose of this study was twofold: the first purpose was to build sub-word based speaker-dependent and speaker-independent Amharic word recognizers using the contending sub-word units – the phoneme, the CV-syllable and tied-state triphones. The second purpose was to compare the various sub-word units of the language based on the recognition performance of the recognizers that are built using these units.

This chapter presents a summary of the conclusions learnt from the experiments undertaken, and future research areas pertaining to speech recognition are recommended.

5.2. CONCLUSIONS

In this study the effect of using small context-independent (phonemes) and context dependent sub-word units (triphones) for large vocabulary recognition has been explored. Since these units can be trained from speech samples with no human intervention to segment and label individual sections of speech, context independent sub-word units are easy to train. Moreover, they can be extended to context-dependent triphone models automatically with almost no extra effort.

Even though the CV-syllable was an attractive sub-word unit for Amharic given the nature of language, it has resulted in a relatively poor performance, i.e., 84% on the training data and 70% on the testing data using the researcher's own voice. The same voice has resulted in 92% percent recognition accuracy for the phoneme-based recognizer on both training and testing data while 94% and 90% recognition accuracies have been obtained for tied-state triphones on the training

data and testing data respectively. To confirm the fact, two more speakers (speaker 2 and speaker 6) have also been tested and the recognition results obtained are relatively poor as indicated in Table 4.1.

The HMM parameters used also affect the performance of the recognizers. In this experience, the CV-syllable based recognizer has resulted in a severely degraded performance with single-Gaussian, 3-state left-right HMM model and improved results have been obtained with multiple (5 mixtures) Gaussian, 8-state left-right HMM model.

The other factor used for comparison is the amount of preprocessing required to build the recognizers. Considering the complexity of model building, phoneme-based recognizers stand first followed by tied-state triphones. CV-syllables have been discovered to be out of competition in this respect, too.

Another point worth mentioning is that despite the insufficiency of training data and small number of speakers, the results obtained for the speaker-independent system is enticing – 91.46% for tied-state triphone models. Moreover, an average recognition accuracy of 86% has been obtained when the system is tested with people who had no involvement in training the models. These results show the potential of tied-state triphones for further improvements in both speaker-dependent and speaker-independent models.

In conclusion, in the absence of a sufficiently large and a balanced reference database, the results obtained are encouraging. When the various recognizers so built are compared in terms of performance and ease of modeling, phoneme-based recognizers stand first. With sufficiently large training data with good phonetic coverage, tied-state triphone models proved themselves promising.

Based on the experience of this experiment, the following recommendations are drawn to point out areas for further research.

5.3. RECOMMENDATIONS

The following recommendations are made based on the findings and limitations of this research. The Amharic word recognizer built in this endeavor is partial in the sense that it does not consider the entire sound system of the Amharic language, which are "nearly" in one-to-one correspondence with the Amharic CV-syllables. Therefore, one natural extension of this work is to include all the Amharic sounds with corresponding representative example words and extending the training set. It is also worthwhile to increase the number of speakers for the speaker-independent models.

One of the problems of speech recognition is that of striking a balance between the conflicting desires of high accuracy and insufficient data. Training data is often insufficient if the developers themselves have to develop a corpus. This problem of data sufficiency can be solved if a ready-made corpus or reference database is available for researchers. A good corpus contains speech samples from significantly large number of speakers from different demographic and socio-linguistic groups. Thus, it is strongly recommended that the development of a reliable, sufficiently large corpus is an area worth consideration and investment.

It has been said (Markowitz, 1996:252) that a long-standing goal of the speech recognition industry has been to create a listening typewriter. The first step towards the realization of such a system has been taken with this project – discrete-word recognition system. It is natural for the next phase to deal with continuous speech recognition. As a first step towards a continuous

speech recognition system, *application-oriented* continuous or connected speech recognition is recommended.

As discussed in section 4.7, the construction of a resource on the rules of Amharic that decides which consonants may cluster together in which order, and which are prohibited is necessary for the subsequent researches. This, in fact, requires the involvement of scholars with strong linguistics background.

The current system uses statistical language models to help reduce the search space and resolve ambiguity. As vocabulary size grows, it is important to get additional constraints that cannot be captured by purely statistical models. This involves incorporating syntactic and semantic constraints of the language. Building such a language model is very important in order to build continuous speech recognition systems.

The construction of public domain pronunciation dictionaries which are valuable in sub-word based speech recognition is also an area worth deliberation.

It has been noted that the laboratory where this research was conducted was noisy and was not in any way favorable for the research. Therefore, a laboratory where the environmental factors are optimum should be set up to facilitate research in the area.

Even though the current work does not prove the fact, for Amharic CV-syllables appear to be an opportunity. It is, therefore, recommended that other researches be done to exploit the prospects of CV-syllables using another toolkit or a different approach.

Finally, as an alternative to the method applied in this research, phoneme, CV-syllable or word n-grams may be tested and the results may be compared with that of the current research output.

REFERENCES

Armstrong, B. "Speech Recognition Application Program Interface Committee." AVIOS '94 Proceedings, 1994 .

Aster Tadesse. *The syllable Structure of Amharic and the Syllabification of medial consonant clusters and geminates*. Addis Ababa: Addis Ababa University, 1981 .

Barr, A and Feigenbaum, Edward A. T*he Handbook of Artificial Intelligence.* California: William Kaufmann, Inc., 1981 .

Bender, L. M. *et al.*. "The Ethiopian Writing System." *The Languages of Ethiopia*. Ed. Bender, M. *et al.* , London: Oxford University Press, 1976 . 120-130.

Cambridge University Engineering Department HTK Home Page. http://htk.eng.cam.ac.uk . 2000

Castillo, E., Gutierrez, J.S. and Hadi, A.S. *Expert Systems and Probabilistic Network Model Network.* Springer, 1997 .

Center for Speech Technology Research (CSTR). Espresso I: Phonetically featured Syllables for Speech Recognition. Available at *http://www.cstr.ed.ac.uk/projects/espresso/espresso_l.html*. *1999.*

Clark, V.P, A. Eschholz, and A.F. Rosa, eds. *Language: Introductory Readings.* 4th ed. New York: St. Martin's Press, 1985 .

Cowley, R, *et al.*. "The Amharic Language." *Languages in Ethiopia.* Ed. Bender, M. *et al.*, London: Oxford University Press, 1976 . 77-90.

Davis, S.B. and P. Mermelstein. "Comparison of Parametric Representations for Monosyllabic word recognition in Continuously spoken sentences." *Readings in Speech Recognition.* Eds. Waibel, Alex and Kai-Fu Lee, California: Morgan, 1980 .

Dawkins, C. *Fundamentals of Amharic.* Addis Ababa Sudan Interior Mission. 1969 .

De Mori, R and Brugnara,F. "HMM Methods in Speech Recognition." *Survey of the State of the Art in Human Language Technology.* Eds. Cole, R.A *et al.*, Oregon Graduate Institute, 1995 .

Godin, R.J. "Your Word in Your computer's command." *Electronics.* Vol. 56. 1983 . 131

Hon, H-W and K-F, Lee. "On Vocabulary-Independent Speech Modeling." *Proceedings of the International Conference On Acoustics, speech and signal processing* 2, 1990 . 725-728 .

Huang, X.D. and Jack, M. A. "Semi-Continuous Hidden Markov Models for Speech Signals." *Readings in Speech Recognition*. Eds. Waibel, Alex and Kai-Fu Lee, California: Morgan, 1989 .

Hunt, M.J. "Signal Representation" *Survey of State of the Art in Human Language Technology*. Eds. Cole, R. A. *et al.*, Oregon Graduate Institute, 1995 . 11-16 .

Jelinek, F. "Self-Organized Language Modeling for Speech Recognition." *Readings in Speech Recognition*. Eds. Waibel, Alex and Kai-Fu Lee, California: Morgan, 1985 . 450-507

Juang, B-H and S. Furui. "Automatic Recognition and Understanding of spoken language – A First Step Toward Natural Human-Machine Communication." *Proceedings of the IEEE*. Vol. 88. No. 8. 2000 . 1142-1163 .

Lea, W.A. "Establishing the Value of voice communication with computers", IEEE Trans. Audio and Electroacoustics. Vol. AU-16. 1968. 184-197 .

Lee, K-F. "Context-Dependent Phonetic Hidden Markov Models for Speaker-Independent Continuous Speech Recognition." *Readings in Speech Recognition*. Eds. Waibel and K-F, Lee. 1990 . 347-362 .

Lee, K-F, H-E, Hon, and R. Reddy. "An overview of SPHINX Speech Recognition System." *Readings in Speech Recognition*. Eds. Waibel and K-F, Lee. 1990 . 600-610 .

Markowitz, J. A. *Using Speech Recognition*. Upper Saddle River, New Jersey: Prentice Hall, Inc., 1996 .

Martin, T.B. "Practical Applications of Voice Input to machines." *IEEE proc*. Vol. 64. 1976 . 487-500.

Mullen, D.S., "Issues in the Morphology and Phonology of Amharic: the Lexical generation of Pronominal Clitcs." *Ph.D. Thesis*. University of Ottawa, 1986 .

Nahm, E. and D. Slater. "Speech Recognition." *The Ultimate Multimedia Handbook.* Ed. Keyes, Jessica , New York: McGraw-Hill, 1997 .

Philip, G and E. S. Young. "Man-machine Interaction by voice: Developments in Speech Technology" *Journal of Information Science* vol. 13. 1987. 3-14.

Rabiner, L.R. "A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition." *Proceedings of the IEEE.* Vol. 77. 1989 . 257-286.

Rabiner, L.R., and B-H, Juang. *Fundamentals of Speech Recognition.* Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1993 .

Reddy, D.R. "Speech Recognition by Machine: a Review." *Readings in Speech Recognition.* Eds. Waibel, Alex and Kai-Fu Lee, California: Morgan, 1976 .

Rodman, R. D. *Computer Speech Technology.* Norwood: Artech House, Inc., 1999 .

Roucos, S. and M.O. Dunham. "A stochastic Segment Model for phoneme-Based Continuous Speech Recognition." *Readings in Speech Recognition.* Eds. Waibel, A. and K-F, Lee, California: Morgan, 1987. 367-370.

Schafer, R.W. and L.R. Rabiner. "Digital Representation." *Readings in Speech Recognition.* Eds. Waibel, A. and K-F, Lee, California: Morgan, 1975. 49-63.

Solomon Berhanu. *Isolated Amharic Consonant-Vowel (CV) Syllable Recognition: An experiment Using the Hidden Markov Model.* Master's Thesis, Addis Ababa University, Addis Ababa, 2001 .

Tadesse Beyene. "The Ethiopian Writing System." Paper presented at *the 12th International Conference of Ethiopian Studies*, Michigan State University, 1994 .

Turn, R., " The use of speech for man machine computer interaction", RAND Report-1386-ARPA, RAND Corp., Santa Monica, CA. 1974 .

Waibel, A. and K-F., Lee. "Stochastic Approaches: Introduction." *Readings in Speech Recognition.* Eds. Waibel, A. and K-F., Lee. California: Morgan Kaufmann Publishers, Inc., 1990 . 263-265 .

----., "Why Study Speech Recognition: Introduction." *Readings in Speech Recognition.* Eds. Waibel, A. and K-F, Lee . California: Morgan Kaufmann Publishers, Inc., 1990

Waterworth, J. A. and M. Talbot. *Speech and Language-based Interaction with machines: Towards the Conversational Computer.* Chichester: Ellis Horwood Limited, John Wiley and Sons, 1987 .

White, G.M. , "Natural Language Understanding" *Communication of the ACM.* vol. 33. 1990 . 74-82 .

Young, Steve *et al.*. The HTK Book. Microsoft Corporation. 2000 .

Zue, V and R. Cole. "Spoken Language Input: Overview." *Survey of the State of the Art in Human Language Technology*. Eds. Cole, R.A *et al.*, Oregon Graduate Institute, 1995 .

ባዬ ይማም "ፊደል እንደገና" ፣ *የኢትዮጵያ ቋንቋዎችና የሥነ ጽሑፍ መጽሔት፣ ቁጥር* 7፣ (1 – 32) ። 1997

APPENDICES

Appendix A. The Amharic Character Set

Adapted from: Bender et al. (1976).

Order							Labialised				
1st	2nd	3rd	4th	5th	6th	7th					
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ					
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ			ሏ		
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ					
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ			ሟ		
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ					
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ			ሯ		
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ			ሷ		
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ			ሿ		
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ	ቈ	ቊ	ቋ	ቌ	ቍ
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ			ቧ		
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ			ቷ		
ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ			ቿ		
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ	ኈ	ኊ	ኋ	ኌ	ኍ
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ			ኗ		
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ			ኟ		
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ					
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ					
ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ					
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ	ኰ	ኲ	ኳ	ኴ	ኵ
ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ					
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ			ዟ		
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ			ዧ		
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ					
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ	ጐ	ጒ	ጓ	ጔ	ጕ
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ			ዷ		
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ					
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ			ጧ		
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ			ጯ		
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ			ጿ		
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ					
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ			ፏ		
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ					
ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ					

Appendix B. The Amharic Phonemes

Adapted from: Solomon (2002)

Amharic Symbol	In this paper	Amharic Word Example	
ም	m	/mar/	'honey'
ት	t	/tEmari/	'student'
ን	n	/nIgus/	'king'
ል	l	/lIb/	'heart'
ብ	b	/bet/	'house'
ር	r	/kIrIkIr/	'debate'
ይ	y	/ayn/	'eye'
ው	w	/wIyIyIt/	'discussion'
ስ	s	/sEw/	'man'
ግ	g	/gEmEd/	'rope'
ድ	d	/dabbo/	'bread'
ክ	k	/kErEmela/	'candy'
ች	č	/ ĉIgIr/	'problem'
ጥ	t'	/t'Ewat/	'morning'
ቅ	k'	/k'ElEm/	'paint'
ሀ	h	/hasab/	'idea'
ፍ	f	/fIlagot/	'interest'
ጅ	ǰ	/ǰEmErE/	'he started'
ኝ	ň	/tE ň a/	'he slept'
ሽ	š	/ šErErit/	'spider'
ዕ	s'	/s'Ehai/	'sun'
ጭ	č'	/č'ErEk'a/	'moon'
ዕ	ʔ	/ʔntE/	'you'
ፕ	p	/polis/	'police'
ጓ	g^w	/g^wdEňa/	'friend'
ቋ	k'^w	/k'wEt'ErE/	'he tied'
ኋ	h^w	/bEh^wala/	'after'
ኳ	k^w	/ank^{w}k^w/	'he knocked'
ዥ	ž	/žIwažIwe/	'swing'
ዝ	z	/zEmEd/	'relative'

Amharic Symbol	In this paper	Amharic Word Example	
ጵ	p’	/p’ap’p’as/	‘bishop’
ኧ	E	/ErE/	‘exclamation’
ኡ	u	/udEt/	‘circulation’
ኢ	i	/itIyop’iya /	‘Ethiopia’
አ	a	/abat/	‘father’
ኤ	e	/eli /	‘tortoise’
እ	I	/Inat/	‘mother’
ኦ	o	/oromo/	‘oromo’

Appendix C. The Training Set Utterances

CV-syllable	Phonemes		Example	CV-syllable	Phonemes		Example
mE	m	E	ታመመ	lE	l	E	ኣለ ኣለ (Geminated)
mu	m	u	መ·ልመ·ል	lu	l	u	ኣሉ· ኣሉ· (Geminated)
mi	m	i	ሚሚ	li	l	i	ገሊላ
ma	m	a	እማማ	la	l	a	ላባ ሞላ
me	m	e	እሜቴ ታርሜ	le	l	e	ሌባ
mI	m	I	ምርምር ሲታረም	lI	l	I	ልብ ብል
mo	m	o	ሞለሞለ	lo	l	o	ኣሎሎ
tE	t	E	ተነተነ	bE	b	E	ኣበበ
tu	t	u	ተረቱ ተረቱ (Geminated)	bu	b	u	በ·ና
ti	t	i	ቲማቲም	bi	b	i	ቢራቢሮ ጋላቢ
ta	t	a	ታታሪ መታ·	ba	b	a	ኣባባ ኣበባ
te	t	e	እቴቴ	be	b	e	ቤት
tI	t	I	ምርት ትርታ	bI	b	I	ብር ልብስ
to	t	o	ቶሎቶሎ	bo	b	o	ቦካ ኣልቦ
nE	n	E	ገነነ	rE	r	E	ተረተረ
nu	n	u	ኑግ	ru	r	u	ሩር
ni	n	i	ኒያላ	ri	r	i	ተማሪ
na	n	a	ገና ገና (Geminated)	ra	r	a	በራ· በራ· (Geminated)
ne	n	e	ወኔ	re	r	e	በሬ በሬ (Geminated)
nI	n	I	ንጉስ መለመን	rI	r	I	ክርክር ክርር
no	n	o	ኖረ መኖ	ro	r	o	በረሮ

CV-syllable	Constituent Phonemes		Example	CV-syllable	Constituent Phonemes		Example
yE	y	E	ተለያየ	gE	g	E	ገመገመ
yu	y	u	ተለያዩ	gu	g	u	ጉንዳን
yi	y	i	እዪ	gi	g	i	ጊደር
ya	y	a	ሙያ	ga	g	a	ተንጋጋ
ye	y	e	እማዬ	ge	g	e	ጉራጌ
yI	y	I	ኣይን ይመስገን	gI	g	I	ግግር
yo	y	o	ዮናስ መለዮ	go	g	o	ጎረጎረ
wE	w	E	ወረወረ	dE	d	E	ደረደረ
wu	w	u	ሰዉ	du	d	u	ዱላ
wi	w	i	ልባዊ	di	d	i	ዲዳ
wa	w	a	ዋና ዋና (Geminated)	da	d	a	ዳዳ
we	w	e	ደዌ	de	d	e	ዳዴ
wI	w	I	ውድ ሰው	dI	d	I	ድርድር ድርድር (Geminated)
wo	w	o	ኣዎ እርስዎ	do	d	o	እንዶድ
sE	s	E	ሰረሰረ	kE	k	E	ከተከተ
su	s	u	ሱሪ ተረሱ	ku	k	u	ከኩ
si	s	i	ሲጋራ	ki	k	i	ካኪ
sa	s	a	ሳሳ	ka	k	a	ካባ ከካ
se	s	e	ሴት መነኩሴ	ke	k	e	ኬክ
sI	s	I	ስብስብ ስብስብ(Geminated)	kI	k	I	ክክ
so	s	o	ሶስት	ko	k	o	ኮክ

Vowel	Example
E	ኧረ
u	ኡደት
I	ኢላማ
a	ኣላማ
e	ኤሊ
I	እመቤት
o	ኦሮሞ

Appendix D. The Testing Set

No.	Testset I			No.	Testsetll		
1	ሞላ	2	ደዌ	1	በሶ	2	በላይ
3	ኣለ(Geminated)	4	በረሮ	3	በላ	4	መንገድ
5	በራ	6	ሲጋራ	5	ስኒ	6	መዳይ
7	ካኪ	8	ብር	7	ከበደ	8	ታየ
9	ኤሊ	10	ይመስገን	9	ሰለሞን	10	ሰማይ
11	ወኔ	12	መረዋ	11	ለበሰ	12	መሬት
13	ተረሱ	14	ምርት	13	ነው	14	ምሬት
15	ዱላ	16	ጉንዳን	15	ናት	16	በኣል
17	ኣይን	18	ኣበበ	17	ደረሰ	18	ሳር
19	ቤት	20	ዋና	19	ሞተ	20	ንጉሴ
21	ሱሪ	22	ዋና (Geminated)	21	መላኩ	22	ኣስተማሪ
23	ተረቱ(Geminated)	24	ዳዲ	23	ምሳ	24	ደላላ
25	ኬክ	26	ኣሎሎ	25	ቦርሳ	26	ገበያ
27	ኢላማ	28	ቶሎቶሎ	27	ደብተር	28	ወንድም
29	ስብስብ(Geminated)	30	ሰው	29	ታሪክ	30	ምግብ
31	ጉጉት	32	ጊደር	31	ተማሪ	32	ቢላዋ
33	ደረደረ	34	ሴት	33	ዋጋ	34	ካሳ
35	ኑግ	36	ጋላቢ	35	ሜዳ	36	ከሳ
37	ገና	38	ከተከተ	37	ኣንድ	38	ድርሰት
39	ቢራቢሮ	40	ሚሚ	39	ኣባት	40	ገደለ
41	በራ	42	ኣለቦ	41	ካባ	42	ረታ
43	ኣለ	44	ኣዎ	43	ኣረሰ	44	ጎዳና
45	ስብስብ	46	ልባዊ	45	በሳ	46	ከበሮ
47	ገና	48	ቲማቲም	47	መና	48	ቤተሰብ
49	ተረቱ	50	ክክ	49	መና(Geminated)	50	ማለ

Appendix E. The Pronunciation Dictionaries

E.I. Phoneme-based Dictionary

ababa	a b a b a sp
abat	a b a t sp
abEba	a b E b a sp
abEbE	a b E b E sp
alama	a l a m a sp
albo	a l b o sp
alE	a l E sp
alolo	a l o l o sp
alu	a l u sp
and	a n d sp
arEsE	a r E s E sp
astEmari	a s t E m a r i sp
awo	a w o sp
ayn	a y n sp
bEal	b E a l sp
bEla	b E l a sp
bElay	b E l a y sp
bEra	b E r a sp
bEre	b E r e sp
bErEro	b E r E r o sp
bEsa	b E s a sp
bEso	b E s o sp
bet	b e t sp
betEsEb	b e t E s E b sp
.	
.	
.	
sil []	sil
sIni	s I n i sp
sitarEm	s i t a r E m sp
sost	s o s t sp
.	
.	
.	
waga	w a g a sp
wana	w a n a sp
wEndIm	w E n d I m sp
wEne	w E n e sp
wErEwErE	w E r E w E r E sp
wId	w I d sp
yImEsgEn	y I m E s g E n sp
yonas	y o n a s sp

E.II. CV-syllable based Dictionary

ababa	a ba ba sp
abat	a ba t sp
abEba	a bE ba sp
abEbE	a bE bE sp
alama	a la ma sp
albo	a l bo sp
alE	a lE sp
alolo	a lo lo sp
alu	a lu sp
and	a n d sp
arEsE	a rE sE sp
astEmari	a s tE ma ri sp
awo	a wo sp
ayn	a y n sp
bEal	bE a l sp
bElay	bE la y sp
bEla	bE la sp
bEra	bE ra sp
bEre	bE re sp
bErEro	bE rE ro sp
bEsa	bE sa sp
bEso	bE so sp
bet	be t sp
betEsEb	be tE sE b sp
bilawa	bi la wa sp
birabiro	bi ra bi ro sp
.	
.	
.	
sil []	sil
sIni	sI ni sp
sitarEm	si ta rE m sp
sost	s o s t sp
.	
.	
.	
wEndIm	wE n dI m sp
wEne	wE ne sp
wErEwErE	wE rE wE rE sp
yImEsgEn	yI mE s gE n sp
yonas	yo na s sp

Appendix F. The Task Grammar

```
(
 sil( tamEmE|mulmul|mimi|Imama|tarIme|mIrImIr|sitarEm|molEmolE|tEnEtEnE|
      tErEtu|tErEtu|timatim|tatari|mEta|Itete|mIrt|tIrIta|tolotolo|gEnEnE|nug|niyala|
      gEna|gEna|wEne|nIgus|mElEmEn|norE|mEno|alE|alE|alu|alu|gElila|laba|
      mola|leba|lIb|bIl|alolo|abEbE|buna|birabiro|galabi|ababa|abEba|bet|bIrr|lIbs|
      boka|albo|tErEtErE|rur|tEmari|bEra|bEra|bEre|bEre|kIrIkIr|kIrIr|bErEro|
      tElEyayE|tElEyayu|Iyi|muya|Imaye|ayn|yImEsgEn|yonas|mElEyo|wErEwErE|
      sEwu|lIbawi|wana|wana|mErEwa|dEwe|wId|sEw|awo|IrsIwo|sErEsErE|suri|tErEsu|
      sigara|sasa|set|mEnEkuse|sIbsIb|sIbIsIb|sost|gEmEgEmE|gundan|gidEr|tEngaga|
      gugut|gIgIr|gorEgorE|dErEdErE|dula|dida|dada|dade|dIrdIr|dIrIdIr|dId|Indod|
      kEtEkEtE|kuku|kaki|kaba|kEka|kek|kIk|kok|ErE|udEt|ilama|alama|eli|ImEbet|
      oromo|bEso|bEla|sIni|kEbEdE|sElEmon|lEbEsE|nEw|nat|dErEsE|motE|mElaku|
      mIsa|borsa|dEbtEr|tarik|tEmari|waga|meda|and|abat|kaba|arEsE|bEsa|mEna|
      bElay|mEngEd|muday|tayE|sEmay|Eret|mIret|bEal|sar|nIguse|astEmari|dElala|
      gEbEya|wEndIm|mIgIb|bilawa|kasa|kEsa|dIrsEt|gEdElE|rEta|godana|kEbEro|
      betEsEb|malE)sil

)
```

Appendix G. The SLF of the Network

VERSION=1.0

Define size of network: N=number of nodes and L=number of arcs

N=176 L=345

#List of nodes: I = node-number, W=word

I=0 W=!NULL I=1 W=!NULL I=2 W=sil I=3 W=tamEmE

I=4 W=!NULL I=5 W=mulmul I=6 W=mimi I=7 W=lmama

I=8 W=tarlme I=9 W=mlrlmlr I=10 W=sitarEm I=11 W=molEmolE

I=12 W=tEnEtEnE I=13 W=tErEtu I=14 W=tErEtu I=15 W=timatim

I=16 W=tatari I=17 W=mEta I=18 W=Itete I=19 W=mlrt

I=20 W=tlrlta I=21 W=tolotolo I=22 W=gEnEnE I=23 W=nug

I=24 W=niyala I=25 W=gEna I=26 W=gEna I=27 W=wEne

I=28 W=nlgus I=29 W=mElEmEn I=30 W=norE I=31 W=mEno

I=32 W=alE I=33 W=alE I=34 W=alu I=35 W=alu

.

.

.

List arcs: J=arc-number, S=Start-node, E=end-node

J=0 S=174 E=1 J=1 S=0 E=2 J=2 S=2 E=3 J=3 S=3 E=4

J=4 S=5 E=4 J=5 S=6 E=4 J=6 S=7 E=4 J=7 S=8 E=4

J=8 S=9 E=4 J=9 S=10 E=4 J=10 S=11 E=4 J=11 S=12 E=4

J=12 S=13 E=4 J=13 S=14 E=4 J=14 S=15 E=4 J=15 S=16 E=4

J=16 S=17 E=4 J=17 S=18 E=4 J=18 S=19 E=4 J=19 S=20 E=4

J=20 S=21 E=4 J=21 S=22 E=4 J=22 S=23 E=4 J=23 S=24 E=4

J=24 S=25 E=4 J=25 S=26 E=4 J=26 S=27 E=4 J=27 S=28 E=4

J=28 S=29 E=4 J=29 S=30 E=4 J=30 S=31 E=4 J=31 S=32 E=4

J=32 S=33 E=4 J=33 S=34 E=4 J=34 S=35 E=4 J=35 S=36 E=4

.

.

.

J=328 S=2 E=159 J=329 S=2 E=160 J=330 S=2 E=161 J=331 S=2 E=162

J=332 S=2 E=163 J=333 S=2 E=164 J=334 S=2 E=165 J=335 S=2 E=166

J=336 S=2 E=167 J=337 S=2 E=168 J=338 S=2 E=169 J=339 S=2 E=170

J=340 S=2 E=171 J=341 S=2 E=172 J=342 S=2 E=173 J=343 S=4 E=174 J=344 S=2 E=175

Appendix H. The Prototype Definitions

H.I. For the Phoneme-based Approach

```
~o <VecSize> 39 <MFCC_0_D_A>
~h "protoI"
<BeginHMM>
  <NumStates> 5
  <State> 2
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <State> 3
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <State> 3
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <State> 4
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

  <TransP> 5
   0.0 1.0 0.0 0.0 0.0
   0.0 0.6 0.4 0.0 0.0
   0.0 0.0 0.6 0.4 0.0
   0.0 0.0 0.0 0.7 0.3
   0.0 0.0 0.0 0.0 0.0
<EndHMM>
```

H.II. For the CV syllable-based Approach

~o <VecSize> 39 <MFCC_0_D_A>
~h "protoII"
<BeginHMM>
<NumStates> 10
<State> 2
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 3
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 4
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 5
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 6
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 7
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39

1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

<State> 8

<Mean> 39

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

<Variance> 39

1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

<State> 9

<Mean> 39

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

<Variance> 39

1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

<TransP> 10

0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

0.0 0.2 0.2 0.2 0.2 0.1 0.1 0.0 0.0 0.0

0.0 0.0 0.2 0.2 0.2 0.2 0.1 0.1 0.0 0.0

0.0 0.0 0.0 0.2 0.2 0.2 0.2 0.1 0.1 0.0

0.0 0.0 0.0 0.0 0.2 0.2 0.2 0.2 0.1 0.1

0.0 0.1 0.1 0.2 0.2 0.2 0.2 0.0 0.0 0.0

0.0 0.0 0.1 0.1 0.2 0.2 0.2 0.2 0.0 0.0

0.0 0.0 0.0 0.1 0.1 0.2 0.2 0.2 0.2 0.0

0.0 0.0 0.0 0.0 0.1 0.1 0.2 0.2 0.2 0.2

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

<EndHMM>

Appendix I. Sample Script File

tr0101_0.mfc
tr0101_1.mfc
tr0102_0.mfc
tr0102_1.mfc
tr0103_0.mfc
tr0103_1.mfc
tr0104_0.mfc
tr0104_1.mfc
tr0105_0.mfc
tr0105_1.mfc
tr0106_0.mfc
tr0106_1.mfc
tr0107_0.mfc
tr0107_1.mfc
.
.
.
tr0118_0.mfc
tr0118_1.mfc
tr0119_0.mfc
tr0119_1.mfc
tr0120_0.mfc
tr0120_1.mfc
tr0121_0.mfc
tr0121_1.mfc
tr0122_0.mfc
tr0122_1.mfc
tr0123_0.mfc
tr0123_1.mfc
tr0124_0.mfc
tr0124_1.mfc
tr0125_0.mfc
tr0125_1.mfc

Appendix J. The File tree.hed

```
RO 100.0 stats
TR 0
QS  "R_NonBoundary"      { *+* }
QS  "R_Stop"             { *+d,*+b,*+t,*+k,*+g }
QS  "R_Nasal"            { *+m,*+n}
QS  "R_Fricative"        { *+s}
QS  "R_Liquid"           { *+l,*+r,*+w,*+y}
QS  "R_Vowel"            { *+e,*+i,*+o,*+u,*+a,*+E,*+I}
QS  "R_C-Front"          { *+b,*+m,*+w }
QS  "R_C-Central"        { *+t,*+d,*+n,*+s,*+l,*+r }
QS  "R_C-Back"           { *+y,*+k,*+g}
QS  "R_V-Front"          { *+i,*+e }
QS  "R_V-Central"        { *+E,*+a,*+I}
QS  "R_V-Back"           { *+u,*+o }
QS  "R_Front"            { *+b,*+m,*+w,*+i,*+e}
QS  "R_Central"          { *+t,*+d,*+n,*+s,*+l,*+r,*+a,*+E,*+I}
QS  "R_Back"             { *+y,*+k,*+g,*+o,*+u}
QS  "R_Fortis"           { *+t,*+k,*+s}
QS  "R_Lenis"            { *+b,*+d,*+g}
QS  "R_UnFortLenis"      { *+m,*+n,*+l,*+r,*+y,*+w }
QS  "R_Coronal"          { *+t,*+d,*+n,*+s,*+l,*+r }
QS  "R_NonCoronal"       { *+b,*+m,*+k,*+g,*+y,*+w }
QS  "R_Anterior"         { *+b,*+m,*+t,*+d,*+n,*+s,*+l,*+w }
QS  "R_NonAnterior"      { *+k,*+g,*+r,*+y }
QS  "R_Continuent"       { *+m,*+n,*+s,*+l,*+r,*+y,*+w}
QS  "R_NonContinuent"    { *+b,*+t,*+d,*+k,*+g}
QS  "R_Strident"         { *+s }
QS  "R_UnStrident"       { *+b,*+m,*+t,*+d,*+n,*+k,*+g,*+l,*+r,*+y,*+w }
QS  "R_Glide"            { *+l,*+r,*+y,*+w }
QS  "R_Syllabic"         { *+m,*+l }
QS  "R_Unvoiced-Cons"    { *+t,*+k,*+s}
QS  "R_Voiced-Cons"      { *+b,*+d,*+g,*+y,*+l,*+m,*+n,*+r,*+w }
QS  "R_Unvoiced-All"     { *+t,*+k,*+s,*+sil }
QS  "R_Long"             { *+a,*+l }
QS  "R_Fronting"         { *+e,*+i }
QS  "R_High"             { *+i,*+I,*+u}
QS  "R_Medium"           { *+e,*+E, *+o,*+m,*+l }
QS  "R_Rounded"          { *+u,*+o,*+w }
QS  "R_Unrounded"        { *+a,*+e,*+i,*+I,*+E,*+l,*+r,*+y }
QS  "R_NonAffricate"     { *+s}
QS  "R_IVowel"           { *+i}
QS  "R_EVowel"           { *+E,*+e }
QS  "R_AVowel"           { *+a,*+o }
QS  "R_UVowel"           { *+a,*+m,*+l }
QS  "R_Voiced-Stop"      { *+b,*+d,*+g }
QS  "R_Unvoiced-Stop"    { *+t,*+k }
QS  "R_Front-Stop"       { *+b }
QS  "R_Central-Stop"     { *+t,*+d }
QS  "R_Back-Stop"        { *+k,*+g }
QS  "R_Unvoiced-Fric"    { *+s }
QS  "R_Central-Fric"     { *+s }
QS  "R_a"                { *+a }
QS  "R_b"                { *+b }
```

```
QS  "R_d"               { *+d }
QS  "R_e"               { *+e }
QS  "R_g"               { *+g }
QS  "R_i"               { *+i }
QS  "R_k"               { *+k }
QS  "R_l"               { *+l }
QS  "R_m"               { *+m }
QS  "R_n"               { *+n }
QS  "R_o"               { *+o }
QS  "R_r"               { *+r }
QS  "R_s"               { *+s }
QS  "R_t"               { *+t }
QS  "R_u"               { *+u }
QS  "R_w"               { *+w }
QS  "R_y"               { *+y }
QS  "L_NonBoundary"     { *-* }
QS  "L_Stop"            { b-*,t-*,d-*,k-*,g-* }
QS  "L_Nasal"           { m-*,n-* }
QS  "L_Fricative"       { s-*}
QS  "L_Liquid"          { l-*,r-*,w-*,y-* }
QS  "L_Vowel"           { o-*,a-*,u-*,E-*,I-*,*-e }
QS  "L_C-Front"         { b-*,m-*,w-* }
QS  "L_C-Central"       { t-*,d-*,n-*,s-*,l-*,r-* }
QS  "L_C-Back"          { y-*,k-*,g-* }
QS  "L_V-Front"         { i-*,e-* }
QS  "L_V-Central"       { I-*,E-*,a-* }
QS  "L_V-Back"          { u-*,o-* }
QS  "L_Front"           { b-*,m-*,w-*}
QS  "L_Central"         { t-*,d-*,n-*,s-*,l-*,r-*}
QS  "L_Back"            { y-*,k-*,g-*,o-*,u-* }
QS  "L_Fortis"          { t-*,k-*,s-*}
QS  "L_Lenis"           { b-*,d-*,g-* }
QS  "L_UnFortLenis"     { m-*,n-*,l-*,r-*,y-*,w-* }
QS  "L_Coronal"         { t-*,d-*,n-*,s-*,z-*,l-*,r-* }
QS  "L_NonCoronal"      { b-*,m-*,k-*,g-*,y-*,w-* }
QS  "L_Anterior"        { b-*,m-*,t-*,d-*,n-*,s-*,l-*,w-* }
QS  "L_NonAnterior"     { k-*,g-*,r-*,y-* }
QS  "L_Continuent"      { m-*,n-*,s-*,l-*,r-*,y-*,w-* }
QS  "L_NonContinuent"   { b-*,t-*,d-*,k-*,g-*}
QS  "L_Strident"        { s-* }
QS  "L_UnStrident"      { b-*,m-*,t-*,d-*,n-*,k-*,g-*,l-*,r-*,y-*,w-* }
QS  "L_Glide"           { l-*,r-*,y-*,w-* }
QS  "L_Syllabic"        { m-*,l-* }
QS  "L_Unvoiced-Cons"   { t-*,k-*,s-* }
QS  "L_Voiced-Cons"     { b-*,d-*,g-*,y-*,l-*,m-*,n-*,r-*,w-* }
QS  "L_Unvoiced-All"    { t-*,k-*,s-*,sil-* }
QS  "L_Long"            { o-*,u,m-*,l-*}
QS  "L_Short"           { e-*,a-*,i-*,I-*,E-*}
QS  "L_Front-Start"     { i-*,e-* }
QS  "L_Fronting"        { i-*,e-* }
QS  "L_High"            { i-*,u-*,I-* }
QS  "L_Medium"          { e-*,E-*,o-*,m-*,l-* }
QS  "L_Low"             { a-* }
QS  "L_Rounded"         { o-*,u-*,w-* }
QS  "L_Unrounded"       { e-*,i-*,a-*,E-*,I-*,m-*,l-*,r-*,y-* }
QS  "L_NonAffricate"    { s-*}
QS  "L_IVowel"          { i-* }
```

```
QS  "L_EVowel"          { E-*,e-* }
QS  "L_AVowel"          { a-*,I-* }
QS  "L_OVowel"          { o-* }
QS  "L_UVowel"          { a-*,m-*,l-* }
QS  "L_Voiced-Stop"     { b-*,d-*,g-* }
QS  "L_Unvoiced-Stop"   { t-*,k-* }
QS  "L_Front-Stop"      { b-* }
QS  "L_Central-Stop"    { t-*,d-* }
QS  "L_Back-Stop"       { k-*,g-* }
QS  "L_Unvoiced-Fric"   { s-* }
QS  "L_Central-Fric"    { s-* }
QS  "L_a"               { a-* }
QS  "L_b"               { b-* }
QS  "L_d"               { d-* }
QS  "L_e"               { e-* }
QS  "L_g"               { g-* }
QS  "L_i"               { i-* }
QS  "L_k"               { k-* }
QS  "L_l"               { l-* }
QS  "L_m"               { m-* }
QS  "L_n"               { n-* }
QS  "L_o"               { o-* }
QS  "L_r"               { r-* }
QS  "L_s"               { s-* }
QS  "L_u"               { u-* }
QS  "L_w"               { w-* }
QS  "L_y"               { y-* }

TR 2

TB 375.0 "a_s2"{(a,*-a,*-a+*,a+*).state[2]}
TB 375.0 "b_s2"{(b,*-b,*-b+*,b+*).state[2]}
TB 375.0 "d_s2"{(d,*-d,*-d+*,d+*).state[2]}
TB 375.0 "e_s2"{(e,*-e,*-e+*,e+*).state[2]}
TB 375.0 "E_s2"{(E,*-E,*-E+*,E+*).state[2]}
TB 375.0 "g_s2"{(g,*-g,*-g+*,g+*).state[2]}
TB 375.0 "i_s2"{(i,*-i,*-i+*,i+*).state[2]}
TB 375.0 "k_s2"{(k,*-k,*-k+*,k+*).state[2]}
TB 375.0 "l_s2"{(l,*-l,*-l+*,l+*).state[2]}
TB 375.0 "m_s2"{(m,*-m,*-m+*,m+*).state[2]}
TB 375.0 "n_s2"{(n,*-n,*-n+*,n+*).state[2]}
TB 375.0 "o_s2"{(o,*-o,*-o+*,o+*).state[2]}
TB 375.0 "r_s2"{(r,*-r,*-r+*,r+*).state[2]}
TB 375.0 "s_s2"{(s,*-s,*-s+*,s+*).state[2]}
TB 375.0 "t_s2"{(t,*-t,*-t+*,t+*).state[2]}
TB 375.0 "u_s2"{(u,*-u,*-u+*,u+*).state[2]}
TB 375.0 "w_s2"{(w,*-w,*-w+*,w+*).state[2]}
TB 375.0 "y_s2"{(y,*-y,*-y+*,y+*).state[2]}

TB 375.0 "a_s3"{(a,*-a,*-a+*,a+*).state[3]}
TB 375.0 "b_s3"{(b,*-b,*-b+*,b+*).state[3]}
TB 375.0 "d_s3"{(d,*-d,*-d+*,d+*).state[3]}
TB 375.0 "e_s3"{(e,*-e,*-e+*,e+*).state[3]}
TB 375.0 "E_s3"{(E,*-E,*-E+*,E+*).state[3]}
TB 375.0 "g_s3"{(g,*-g,*-g+*,g+*).state[3]}
TB 375.0 "i_s3"{(i,*-i,*-i+*,i+*).state[3]}
TB 375.0 "k_s3"{(k,*-k,*-k+*,k+*).state[3]}
```

```
TB 375.0 "l_s3"{(l,*-l,*-l+*,l+*).state[3]}
TB 375.0 "m_s3"{(m,*-m,*-m+*,m+*).state[3]}
TB 375.0 "n_s3"{(n,*-n,*-n+*,n+*).state[3]}
TB 375.0 "o_s3"{(o,*-o,*-o+*,o+*).state[3]}
TB 375.0 "r_s3"{(r,*-r,*-r+*,r+*).state[3]}
TB 375.0 "s_s3"{(s,*-s,*-s+*,s+*).state[3]}
TB 375.0 "t_s3"{(t,*-t,*-t+*,t+*).state[3]}
TB 375.0 "u_s3"{(u,*-u,*-u+*,u+*).state[3]}
TB 375.0 "w_s3"{(w,*-w,*-w+*,w+*).state[3]}
TB 375.0 "y_s3"{(y,*-y,*-y+*,y+*).state[3]}

TB 375.0 "a_s4"{(a,*-a,*-a+*,a+*).state[4]}
TB 375.0 "b_s4"{(b,*-b,*-b+*,b+*).state[4]}
TB 375.0 "d_s4"{(d,*-d,*-d+*,d+*).state[4]}
TB 375.0 "e_s4"{(e,*-e,*-e+*,e+*).state[4]}
TB 375.0 "E_s4"{(E,*-E,*-E+*,E+*).state[4]}
TB 375.0 "g_s4"{(g,*-g,*-g+*,g+*).state[4]}
TB 375.0 "i_s4"{(i,*-i,*-i+*,i+*).state[4]}
TB 375.0 "k_s4"{(k,*-k,*-k+*,k+*).state[4]}
TB 375.0 "l_s4"{(l,*-l,*-l+*,l+*).state[4]}
TB 375.0 "m_s4"{(m,*-m,*-m+*,m+*).state[4]}
TB 375.0 "n_s4"{(n,*-n,*-n+*,n+*).state[4]}
TB 375.0 "o_s4"{(o,*-o,*-o+*,o+*).state[4]}
TB 375.0 "r_s4"{(r,*-r,*-r+*,r+*).state[4]}
TB 375.0 "s_s4"{(s,*-s,*-s+*,s+*).state[4]}
TB 375.0 "t_s4"{(t,*-t,*-t+*,t+*).state[4]}
TB 375.0 "u_s4"{(u,*-u,*-u+*,u+*).state[4]}
TB 375.0 "w_s4"{(w,*-w,*-w+*,w+*).state[4]}
TB 375.0 "y_s4"{(y,*-y,*-y+*,y+*).state[4]}

TR 1

AU "fulllist"
CO "tiedlist"

ST "trees"
```

Appendix K. Sample Recognition Output

```
#!MLF!#
"tsII0101_0.rec"
2800000 7500000 bEso -4389.463867
.
"tsII0101_1.rec"
6700000 10600000 bEre -3906.544434
.
"tsII0102_0.rec"
1200000 6700000 bEla -5079.095215
.
"tsII0102_1.rec"
4300000 10700000 mEngEd -6147.696289
.
"tsII0103_0.rec"
1300000 5400000 sIni -3540.826172
.
"tsII0103_1.rec"
3700000 8300000 muday -4658.436523
.
.
.
.
"tsII0124_0.rec"
4600000 9200000 mEna -4119.130859
.
"tsII0124_1.rec"
3600000 9900000 gIgIr -6329.874512
.
"tsII0125_0.rec"
700000 6100000 mEna -4627.381348
.
"tsII0125_1.rec"
2700000 7200000 malE -4209.477539
.
```

DECLARATION

This thesis is my original work and has not been presented for a degree in any other university and that all sources of material used for the thesis have been duly acknowledged.

Kinfe Tadesse

June 2002

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

Ato Daniel Aberra June 2002	Ato Solomon Berhanu June 2002	Ato Ermias Abebe June 2002