B.Sc. in Computer Science and Engineering Thesis

Automatic Phoneme Recognition for Bangla Spoken Language

Submitted by

Israt Jahan Mouri
201014040

Tahsin Sultana
201014023

Md. Wahiduzzaman Khan
201014063

Supervised by

Dr. Mohammad Nurul Huda
Professor & MSCSE Coordinator
Department of Computer Science and Engineering
United International University (UIU), Dhaka, Bangladesh.

Department of Computer Science and Engineering
Military Institute of Science and Technology

CERTIFICATION

This thesis paper titled **"Automatic Phoneme Recognition for Bangla Spoken Language "**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering on December 2013.

Group Members:

Israt Jahan Mouri
Tahsin Sultana
Md. Wahiduzzaman Khan

Supervisor:

_____

Dr. Mohammad Nurul Huda
Professor & MSCSE Coordinator
United International University (UIU), Dhaka, Bangladesh.

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis paper is the outcome of the investigation and research carried out by the following students under the supervision of Dr. Mohammad Nurul Huda, Professor & MSCSE Coordinator ,United International University (UIU), Dhaka, Bangladesh.

It is also declared that neither this thesis paper nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

_____

Israt Jahan Mouri
201014040

_____

Tahsin Sultana
201014023

_____

Md. Wahiduzzaman Khan
201014063

ACKNOWLEDGEMENT

We are thankful to Almighty Allah for his blessings for the successful completion of our thesis. Our heartiest gratitude, profound indebtedness and deep respect go to our supervisor Dr. Mohammad Nurul Huda,Professor & MSCSE Coordinator,United International University (UIU), Dhaka, Bangladesh, for his constant supervision, affectionate guidance and great encouragement and motivation. His keen interest on the topic and valuable advices throughout the study was of great help in completing thesis.

We are especially grateful to the Department of Computer Science and Engineering (CSE) of Military Institute of Science and Technology (MIST) for providing their all out support during the thesis work.

Finally, we would like to thank our families and our course mates for their appreciable assistance, patience and suggestions during the course of our thesis.

Dhaka
December 2013

Israt Jahan Mouri
Tahsin Sultana
Md. Wahiduzzaman Khan

ABSTRACT

Speech recognition (also known as automatic speech recognition or computer speech recognition) converts spoken words to text. The term "voice recognition" is sometimes used to refer to recognition systems that must be trained to a particular speaker as is the case for most desktop recognition software. Recognizing the speaker can simplify the task of translating speech.

For the past two decades, research in speech recognition has been intensively carried out worldwide, spurred on by advances in signal processing, algorithms, architectures, and hardware. Speech recognition systems have been developed for a wide variety of applications, ranging from small vocabulary keyword recognition over dial-up telephone lines, to medium size vocabulary voice interactive command and control systems on personal computers, to large vocabulary speech dictation, spontaneous speech understanding, and limited-domain speech translation.

In this paper, we prepare a Bangla Phoneme recognition system of Bangla Automatic Speech Recognition (ASR). Most of the Bangla ASR system uses a small number of speakers, but 30 speakers selected from a wide area of Bangladesh, where Bangla is used as a native language, are involved here. In the experiments, mel-frequency cepstral coefficients (MFCCs) are inputted to the hidden Markov model (HMM) based classifiers for obtaining phoneme recognition performance. It is shown from the experimental results that MFCC-based method of 39 dimensions provide higher phoneme correct rate and accuracy. Moreover, it requires fewer mixture components in the HMMs .

Moreover, this paper we review some of the key advances in several areas of automatic speech recognition. We also illustrate, by examples, how these key advances can be used for continuous speech recognition of Bangla Language. Finally we elaborate the requirements in designing successful real-world applications and address technical challenges that need to be harnessed in order to reach the ultimate goal of providing an easy-to-use, natural, and flexible voice interface between people and machines.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

AF : Articulatory Features

ASR : Acoustic Model

ATR : Automatic Speech Recognition

BPF : Band Pass Filter

CMN : Cepstral Mean Normalization

DARPA: Defense Advanced Research Projects Agency

DCT : Discrete Cosine Transform

DPF : Distinctive Phonetic Featur

DCR : DPF Correct Rate

DSR : Distributed Speech Recognition

ETSI : European Telecommunications Standards Institute

EM : Expected Maximization

FFT : Fast Fourier Transform

GS : Gram-Schmidt

GMM : Gaussian Mixture Model

GI : Gender-Independent

HNN : Hybrid Neural Network

HMM : Hidden Markov Model

HL : Hidden Layer

In/En : Inhibition/Enhancement

BNAS : Bangla Newspaper Article Sentences

JEIDA : Japan Electronic Industries Development Association

KLT : Kahunen-Loeve Transform

LF : Local Feature

LM : Language Model

LR : Linear Regression

MFCC : Mel-Frequency Cepstral Coefficient

MLN : Multilayer Neural network

OL : Output Layer

OOV : Out-of-vocabulary

PCR : Phoneme Correct Rate

PRA : Phoneme Recognition Accuracy

RNN : Recurrent Neural Network

SPINE : Speech recognition In Noisy Environments

SNR : Signal-to-noise Ratio

CHAPTER 1
INTRODUCTION

1.1 Introduction

Speech is one of the oldest and most natural means of information exchange between human beings. We as humans speak and listen to each other in human-human interface. For centuries people have tried to develop machines that can understand and produce speech as humans do so naturally [1] [2] . Obviously such an interface would yield great benefits [3]. Attempts have been made to develop vocally interactive computers to realise voice/speech recognition. In this case a computer can recognize text and give out a speech output [3].

Voice/speech recognition is a field of computer science that deals with designing computer systems that recognize spoken words. It is a technology that allows a computer to identify the words that a person speaks into a microphone or telephone.

Speech recognition can be defined as the process of converting an acoustic signal, captured by a microphone or a telephone, to a set of words [4] [5]. Automatic speech recognition (ASR) is one of the fastest developing fields in the framework of speech science and engineering. As the new generation of computing technology, it comes as the next major innovation in man-machine interaction, after functionality of text-to-speech (TTS), supporting interactive voice response (IVR) systems.

The reason for the evolution of ASR, hence improved is that it has a lot of applications in many aspects of our daily life, for example, telephone applications, applications for the physically handicapped and illiterates and many others in the area of computer science. Speech recognition is considered as an input as well as an output during the Human Computer Interaction (HCI) design . HCI involves the design implementation and evaluation of interactive systems in the context of the users task and work [6].

The list of applications of automatic speech recognition is so long and is growing; some

of known applications include virtual reality, Multimedia searches, auto-attendants, travel information and reservation, translators, natural language understanding and many more applications [7]

Speech technology is the technology of today and tomorrow with a developing number of methods and tools for better implementation. Speech recognition has a number of practical implementations for both fun and serious works. Automatic speech recognition has an interesting and useful implementation in expert systems, a technology whereby computers can act as a substitute for a human expert. An intelligent computer that acts, responds or thinks like a human being can be equipped with an automatic speech recognition module that enables it to process spoken information. Medical diagnostic systems, for example, can diagnose a patient by asking him a set of questions, the patient responding with answers, and the system responds with what might be a possible disease.

1.2 Background of research

In the past few years a significant portion of the research in speech processing has gone into studying practical methods for automatic speech recognition (ASR). Much of this effort has been stimulated by the Advanced Research Project Agency (ARPA), formerly known as D(defense)ARPA, which has funded research on three large vocabulary recognition (LVR) projects, namely the Naval Resource Management (RM) task, the Air Travel Information System (ATIS) and the North American Business (NAB, previously known as the Wall Street Journal or WSJ) task. In addition, there is a worldwide activity in multi-lingual, large vocabulary speech recognition because of the potential applications to voice-interactive database access and management (e.g. ATIS RM), voice dictation (e.g. discrete word recognizer [8] and continuous speech recognition such as the NAB/WSJ task) and limited-domain spoken language translation.

The Philips SPICOS system and its extensions, the CSELT system (which is currently in trial) for Eurorail information services, the Cambridge University systems, and the LIMSI effort, are examples of the current activity in speech recognition research in Europe. In Japan, large vocabulary recognition systems are being developed based on the concept of *interpreting telephony* and telephone directory assistance. In Taiwan and China, syllable-

based recognizers have been designed to handle large vocabulary Mandarin dictation which is of practical importance because keyboard entry of Chinese text requires a considerable amount of effort and training. In Canada, the most notable research project is the INRS 86,000-word isolated word recognition system. In the United States, in addition to the research being carried out at ATT and IBM, most of the effort is sponsored by ARPA, encompassing efforts by BBN (the BYBLOS system), CMU (the SPHINX systems), Dragon, Lincoln Laboratory, MIT (the Summit system and its extensions), SRI (the DECIPHER system), and many others in the ARPA Human Language Technology Program. A brief history of automatic speech recognition research can be found in the textbook on speech recognition by Rabiner and Juang (1993).

Although we have learned a great deal about how to build practical and useful speech recognition systems, there remain a number of fundamental questions about the technology to which we have no definitive answers. It is clear that the speech signal is one of the most complex signals that we need to deal with. The speech signal being observed is different (even when produced by the same person) each time, even for multiple utterances of the same sequence of words. Part of the reason that automatic speech recognition by machine is difficult is due to this inherent signal variability. In addition to the vast inherent differences across different speakers and different dialects, the speech signal is influenced by the transducer used to capture the signal, the channel used to transmit the signal, and the speaking environment that can add noise to the speech signal or change the way the signal is produced (e.g. the *Lombard effect*) shown in very noisy environments.

There have been many attempts to find so called *distinctive features* of speech which are invariant to a number of factors. Certain distinctive (phonetic) features, such as nasality and voicing, can be used to represent the place and manner of articulation of speech sounds so that speech can be uniquely identified by detecting the acoustic-phonetic properties of the signal. By organizing such knowledge in a systematic manner, speech recognition can (in theory) be performed by first identifying and labeling the sequence of feature vectors and then identifying the corresponding sounds in the speech signal, followed by decoding the corresponding sequence of words using lexical access to a dictionary of words. This has been demonstrated in spectrogram reading by a human expert who can visually segment and identify some speech sounds based on knowledge of acoustic-phonetics of English.

Although the collection of distinctive features, in theory, offers a set of *invariant* features for speech recognition, it is not generally used in most speech recognitions systems. This is due to the fact that the set of distinctive features are usually difficult to identify in spontaneous continuous speech and the recognition results are generally unreliable.

1.3 Goal of the thesis

This thesis concentrates Bangla Phoneme Recognition for Different Acoustic Features:

(a) To critically review literature related to ASR.

(b) To identify speech corpus elements exhibited in Bangla language.

(c) Preparation Bangla speech corpus,

(d) To implement an isolated whole word phoneme recognizer that is capable of recognizing and responding to word .

(e) To train the above developed system in order to make it speaker independent.

(f) To validate the automatic phoneme recognizer developed during the study

1.4 Significance of the Study

The proposed research has theoretical, practical, and methodological significance:

i. The speech corpus developed will be very useful to any researcher who may wish to venture into Bangla language automatic phoneme recognition.

ii. By developing and training a speech recognition system in Bangla language, the semi illiterates would be able to use it in accessing IT tools. This would help bridge the digital divide.

iii. Since Speech technology is the technology of today and tomorrow, the results of this research will help many indigenous Bangla language speakers who are scattered all over the world to take advantage of the many benefits of ICT.

iv. The technology will find applicability in systems such as banking, telecommunications, transport, Internet portals, accessing PC, emailing, administrative and public services, cultural centres and many others.

v. The built system will be very useful to computer manufactures and software developers as they will have a speech recognition engine to include Bangla language in their applications.

vi. By developing and training a speech recognition system in Bangla language, it would mark the first step towards making ICT tools become more usable by the blind and elderly people with seeing disabilities.

CHAPTER 2
SPEECH RECOGNITION

2.1 Introduction

Speech is a natural way of communication among people. Speech communication refers to the processes associated with the production and perception of sounds used in spoken language. We learn to speak prior to write or type. Although there are many professional typists, it is not practical to employ them for all of our typing needs. Automatic speech recognition (ASR) is a useful alternative form of input. It has several advantages over using a computer keyboard. ASR can be used while someone is doing some work or looking somewhere. It also allows people with handicaps to use computers. ASR reduces injuries that arise from using computer keyboards for a long time or from concentrating on the computer screen. ASR can also aid in education; an example is its use as a language-learning tool.

For many languages speech recognition systems have been developed with some success. A notable example is Microsoft Office XP's speech recognition system. Other more specific ASR systems have also been used in different applications with continuing success. A general-purpose speech recognition system (a dictation system) can be used to encode documents into text. This may be especially useful for languages that have a very large character set. Large characters set languages are difficult to type using keyboards, as they require a number of keystrokes to type for a single character. For such languages typing is usually a very time consuming process. Amharic is an example of this type of languages.

This chapter introduces the basic of speech recognition system. Modern speech understanding systems merge interdisciplinary technologies from Signal Processing, Pattern Recognition, Natural Language, and Linguistics into a unified statistical framework.

2.2 Introduction to ASR Systems

Automatic speech recognition (sometimes referred to as just speech recognition, computer speech recognition or erroneously as voice recognition) is the process of converting speech signals uttered by speakers into a sequence of words, which they are intended to represent, by means of an algorithm implemented as a computer program. The recognized words can be the final results, as for applications such as data entry and dictation systems or the words so recognized can be used to trigger specific tasks as in command and control applications. Speech recognition systems can be categorized based on different parameters [9] , some of the more important of which are shown in Table 2.1.

Table 2.1: Typical parameters used to characterize the capability of speech recognition systems.

Parameters	**Range**
Speaking Mode	Isolated words vs. continuous speech
Speaking Style	Read Speech vs. Spontaneous Speech
Enrollment	Isolated words vs. continuous speech
Vocabulary Size	Small (less than 20 words identified) vs. Large (greater than 20,000 words identified)
Language Model	Finite-state vs. Context-sensitive
Signal to Noise Ratio	(SNR) High (>30dB) vs. Low (<10dB)

An isolated-word speech recognition system requires that the speaker pause briefly between words, whereas a continuous speech recognition system does not. Spontaneous speech is much more difficult to recognize than speech read from script. Some systems require speaker enrollment a user must provide samples of his or her speech before using them, whereas other systems are said to be speaker-independent, in that no enrollment is necessary. Some of the other parameters depend on the specific task.

2.3 Applications of ASR Systems

Speech recognition applications vary depending on the vocabulary size, the quality of the microphone, the number of users, and the tolerance for error of the users . A few examples include:

- Hands-free control of machinery
  - Small vocabulary
  - Speaker-independent
  - High-quality microphone (often a headset microphone)
  - Very low error tolerance (error tolerance can be increased with verbal feedback)
- Automatic telephone dialing
  - Small vocabulary
  - Mixture of speaker-independent (digits) and speaker-dependent (names)
  - Low-quality microphone (telephone handset)
  - Moderate error tolerance (if the system asks for confirmation before dialing)
- Telephone access to databases
  - Moderate vocabulary
  - Speaker-independent
  - Low-quality microphone
  - High error tolerance
- Word processing
  - Large vocabulary
  - Speaker-dependent
  - High-quality microphone
  - High error tolerance

2.4 Speech Recognition Basics

Speech recognition is the process by which a computer (or other type of machine) identifies spoken words. Basically, it means talking to your computer and having it correctly recognizes what you are saying. A general solution of speech recognition shows in Fig. 2.1 .

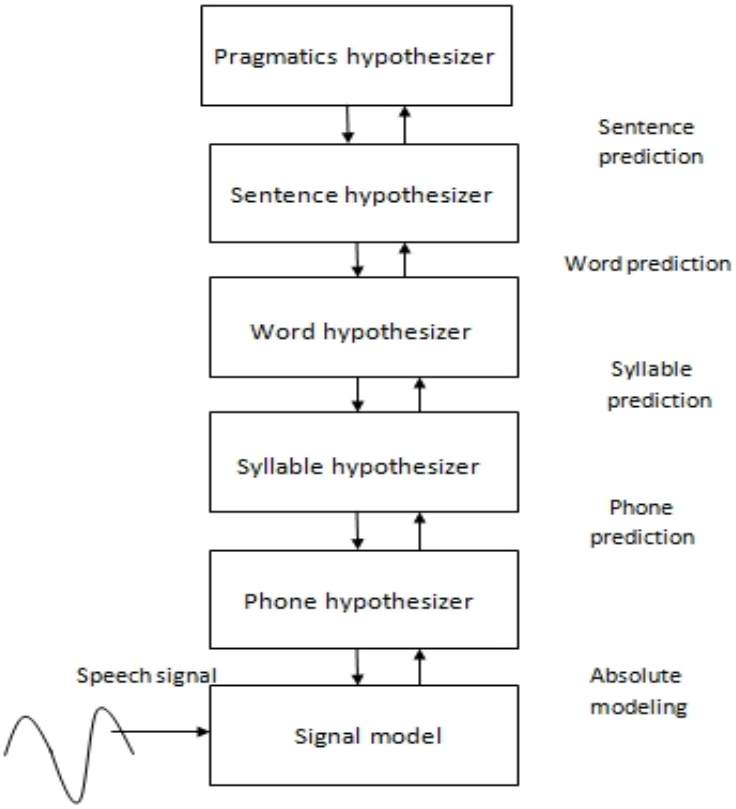


Figure 2.1: General Solution.

The following definitions are the basics needed for understanding speech recognition technology:

Utterance

An utterance is the vocalization (speaking) of a word or words that represent a single meaning to the computer. Utterances can be a single word, a few words, a sentence, or even multiple sentences.

Speaker Dependence

Speaker dependent systems are designed around a specific speaker. They generally are more accurate for the correct speaker, but much less accurate for other speakers. They assume

the speaker will speak in a consistent voice and tempo. Speaker independent systems are designed for a variety of speakers. Adaptive systems usually start as speaker independent systems and utilize training techniques to adapt to the speaker to increase their recognition accuracy.

Vocabularies

Vocabularies (or dictionaries) are lists of words or utterances that can be recognized by the SR system. Generally, smaller vocabularies are easier for a computer to recognize, while larger vocabularies are more difficult. Unlike normal dictionaries, each entry doesn't have to be a single word. They can be as long as a sentence or two. Smaller vocabularies can have as few as 1 or 2 recognized utterances (e.g."Wake Up "), while very large vocabularies can have a hundred thousand or more.

Accurate

The ability of a recognizer can be examined by measuring its accuracy or how well it recognizes utterances. This includes not only correctly identifying an utterance but also identifying if the spoken utterance is not in its vocabulary. Good ASR systems have an accuracy of 98% or more. The acceptable accuracy of a system really depends on the application.

Training

Some speech recognizers have the ability to adapt to a speaker. When the system has this ability, it may allow training to take place. An ASR system is trained by having the speaker repeat standard or common phrases and adjusting its comparison algorithms to match that particular speaker. Training a recognizer usually improves its accuracy.

Training can also be used by speakers that have difficulty speaking, or pronouncing certain words. As long as the speaker can consistently repeat an utterance, ASR systems with training should be able to adapt.

A Language Dictionary

Accepted Words in the Language are mapped to sequences of sound units representing pronunciation, sometimes includes syllabification and stress.

A Filler Dictionary

Non-Speech sounds are mapped to corresponding non-speech or speech like sound units.

Phone

Way of representing the pronunciation of words in terms of sound units. The standard system for representing phones is the International Phonetic Alphabet or IPA. English Language use transcription system that uses ASCII letters where as Bangla uses Unicode letters.

HMM

The Hidden Markov Model is a finite set of states, each of which is associated with a (generally multidimensional) probability distribution.Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are "hidden "to the outside; hence the name Hidden Markov Model.

Language Model

Language Model assigns a probability to a sequence of m words by means of a probability distribution. To model it we can use a regular grammar.

2.5 Overview of the Full System

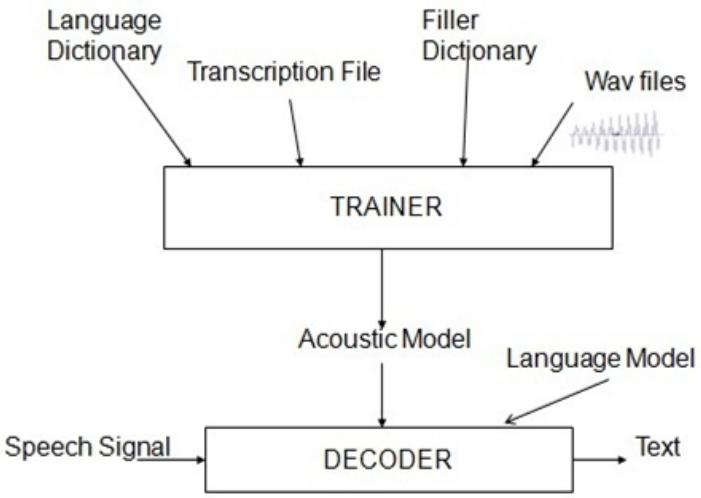


Figure 2.2: Overview of the Project

2.6 Types of Speech Recognition

Speech recognition systems can be separated in several different classes by describing what types of utterances they have the ability to recognize. These classes are based on the fact that one of the difficulties of ASR is the ability to determine when a speaker starts and finishes an utterance. Most packages can fit into more than one class, depending on which mode they're using.

Isolated Words

Isolated word recognizers usually require each utterance to have quiet (lack of an audio signal) on both sides of the sample window. It doesn't mean that it accepts single words, but does require a single utterance at a time. Often, these systems have "Listen/Not Listen "states, where they require the speaker to wait between utterances (usually doing processing during the pauses). Isolated Utterance might be a better name for this class.

Connected Words

Connect word systems (or more correctly 'connected utterances ') are similar to Isolated words, but allow separate utterances to be 'run together 'with a minimal pause between them.

Continuous Speech

Recognizers with continuous speech capabilities are some of the most difficult to create because they must utilize special methods to determine utterance boundaries. Continuous speech recognizers allow users to speak almost naturally, while the computer determines the content. Basically, it's computer dictation.

Spontaneous Speech

There appears to be a variety of definitions for what spontaneous speech actually is. At a basic level, it can be thought of as speech that is natural sounding and not rehearsed. An ASR system with spontaneous speech ability should be able to handle a variety of natural speech features such as words being run together, "ums "and "ahs ", and even slight stutters.

Voice Verification/Identification

Some ASR systems have the ability to identify specific users. This document doesn't cover

verification or security systems.

2.7 Conclusion

The recognition of spontaneous speech can be improved by taking into consideration the effects of the filled pauses while performing the recognition process by:

(1) Either omitting such pauses or by considering them as words to be added to the dictionary of the ASR system.

(2) Recognizing hesitations and restarts.

(3) Improving the model accuracy at both the acoustic level and at the language model level.

(4) Increasing the amount of training data and the lexicon size. This could reduce the error rate without increasing the search complexity.

A better understanding of the properties of human auditory perception that are relevant for decoding the speech signal and are likely to improve the performance of ASR in different environments is necessary for improving the performance of the existing recognizers. Also, using longer acoustic units (for example, syllables) instead of using short-term speech segments followed by post-processing techniques or using dynamic features is promising for the evolution of ASR. Moreover, rich prosodic cues (e.g., fundamental frequency (F0), energy, duration, etc.) that permit successful understanding, which are ignored by state-of-the-art ASR systems, must be considered for better performance. Also, the use of language-independent acoustic models and variable n-gram language models will enhance the performance further.

CHAPTER 3
HIDDEN MARKOV MODELS (HMMS)

3.1 Introduction

Speech Modern general-purpose speech recognition systems are based on Hidden Markov Models (HMMs). These are statistical models which output a sequence of symbols or quantities. HMMs are used in speech recognition because a speech signal can be viewed as a piecewise stationary signal or a short-time stationary signal. In a short-time (e.g., 10 milliseconds), speech can be approximated as a stationary process. Speech can be thought of as a HMM for many stochastic purposes.

Another reason why HMMs are popular is because they can be trained automatically and are simple and computationally feasible to use. In speech recognition, the hidden Markov model would output a sequence of n-dimensional real-valued vectors (with n being a small integer, such as 10), outputting one of these every 10 milliseconds. The vectors would consist of mel frequency cepstral coefficients (MFCCs), which are obtained by taking a Fourier transform of a short time window of speech and decorrelating the spectrum using a discrete cosine transform (DCT), then taking the first (most significant) coefficients. The hidden Markov model will tend to have in each state a statistical distribution that is a mixture of diagonal covariance Gaussians which will give likelihood for each observed vector. Each word, or (for more general speech recognition systems), each phoneme, will have a different output distribution; a hidden Markov model for a sequence of words or phonemes is made by concatenating the individual trained hidden Markov models for the separate words and phonemes.

3.2 Hidden Markov Models

Now we can introduce the application of HMMs in the field of ASR.It was mentioned above that the first step in the speech recognition process is the extraction of spectral features for each time frame. Now, for each two consecutive time frames, t and $t + 1$, the hmm-based recognizer is assumed to transition from state i to state j with probability aij , or stay in state i with probability aii, and emit an observation symbol o_t with probability density $bj(o_t)$.

The hidden Markov model is a statistical model that uses a finite number of states and the associated state transitions to jointly model the temporal and spectral variations of signals. It has been used extensively to model fundamental speech units in speech recognition because the HMM can adequately characterize both the temporal and spectral varying nature of the speech signal. Although many variants exist, perhaps the simplest subword model is a left-to-right HMM with only *self* and *forward* transitions. Within each state of the model there is an observation density function which specifies the probability of a spectral vector. This observation density can either be a *discrete density* (implying the use of one or more codebooks to discrete the input spectral vector), or *a continuous mixture density*, or a so-called *semi-continuous density* or *a tied-mixture density* which is a set of *common* continuous densities whose weights are chosen according to the model state. Tying can also be done at the HMM state level or at the state distribution level . *Stochastic segment modeling* , *dynamic system modeling* , *stochastic trajectory modeling* and *successive state splitting* have also been proposed to extend the HMM to handle intra-state, inter-state, and inter-sample correlations in a more precise manner. Some of the most often used acoustic modeling approaches include:

- ***Maximum Likelihood (ML) Estimation of HMM***: Estimation of HMM parameters is usually accomplished in a batch mode using the ML approach based on the EM (estimation-maximization) algorithm. Segmental ML approaches have also been extensively used. Although ML estimation has good asymptotic properties, it often requires a large size training set to achieve reliable parameter estimation. Smoothing techniques, such as *deleted interpolation* and *Bayesian smoothing*, have been proposed to circumvent some of the problems associated with sparse training data.

- ***Maximum Mutual Information (MMI) Estimation of HMM***: Instead of maximizing

the likelihood of observing both the given acoustic data and the transcription, the MMI estimation procedure maximizes the mutual information between the given acoustic data and the corresponding transcription. As opposed to ML estimation, which uses only class- -specific data to train the classifier for the particular class, MMI estimation takes into account information from data in other classes due to the necessary inclusion of all class priors and conditional probabilities in the definition of mutual information.

- ***Maximum A Posteriori (MAP) Estimation of HMM***: Perhaps the ultimate way to train subword units is to adapt them to the task, to the speaking environment, and to the speaker. One way to accomplish adaptive training is through Bayesian learning in which an initial set of seed models (e.g. speaker-independent or SI models) are combined with the adaptation data to adjust the model parameters so that the resulting set of subword models matches the acoustic properties of the adaptation data. This can be accomplished by maximum a posteriori estimation of HMM parameters and has been successfully applied to HMM-based speaker and context adaptation of whole-word and subword models. On-line adaptation, which continuously adapts HMM parameters and hyperparameters, has also been developed.

- ***Minimum Classification Error (MCE) Estimation of HMM and ANN***: One new direction for speech recognition research is to design a recognizer that minimizes the error rate on task-specific training data. The problem here is that the error probability is not easily expressed in a close functional form because the true probability density function of the speech signal is not known. An alternative is to find a set of model parameters that minimizes the recognition error based on a given set of application-specific, training or cross-validation data. Each training utterance is first recognized and then used for both positive and negative learning by adjusting the model parameters of all competing classes in a systematic manner. For HMM-based recognizers, a family of *generalized probabilistic descent* (GPD) algorithms has been successfully applied to estimate model parameters based on the minimum classification error criterion. The MCE/GPD approaches are also capable of maximizing the *separation* between models of speech units so that both discrimination and robustness of a recognizer can be simultaneously improved.

The observation symbol o_t is a 39-dimensional feature vector with real values. Each phoneme is typically modeled using anywhere between 3 to 12 HMM states, depending on the recognizer implementation. We will give an example of a very simple hmm in Fig. 3.1.

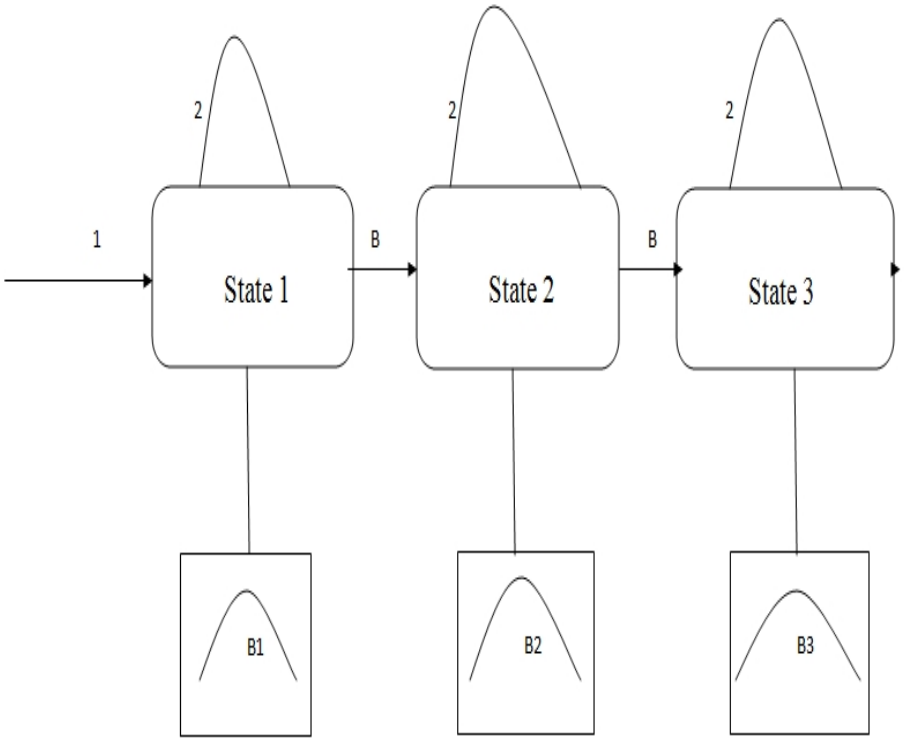


Figure 3.1: An hmm example state diagram. Three states with corresponding output distributions are interconnected by transitions with transitional probabilities architecture.

3.3 HMMs for Word Modeling

A vector sequence $\tilde{X}$ = { , ,....., which belongs a word w_v, can now be generated by the HMM V_v as follows (to simplify the notation, we skip the word dass index v): At each time t = (0,1 ... T_x-1) the HMM is in a state s_{θ_t}, and will generate a vector with the emission probability density $p\ (\vec{x_t} \mid s\theta_t)$ (the initial state probability u_θ gives the probability being in state s_θ at time t = 0). Then it will make a state transition from state s_{θ_t} to state s_{θ_t+1} with the transition probability $a_{\theta_t,\theta_{t+1}}$ and so on. A given vector sequence $\tilde{X}$ can thus be generated by running through a sequence of state indices $\Theta = \left\{\vec{x_0}, \vec{x_1}, ...\vec{x_{T_{x-1}}}\right\}$. However, given only the sequence $\tilde{X}$,one can not determine which sequence Θ was used to generate $\tilde{X}$ i.e., the state sequence is hidden from the observer.

The state transitions of the HMM reflect the sequence of quasi-stationary segments of speech contained in the utterance $\tilde{X}$. Therefore, one usually restricts the transition probability densities of the HMM to a "from left to right "structure, i.e., being in a certain state s_θ , only states with the same or a higher state index θ can be reached with nonzero probability. The initial state probability is usually set to one for the first state and to zero for all the other states, so the generation process always starts in state s_θ. Usually the state transi-tion probabilities are restricted to reach only the two next state indices :

$\alpha_{(\theta,\theta+\Delta)} > 0$, if $0 \leq \Delta \leq 2$

$\alpha_{(\theta,\theta+\Delta)} = 0$, else

This leads to a lefttoright HMM structure as shown for five states in Fig. 3.2

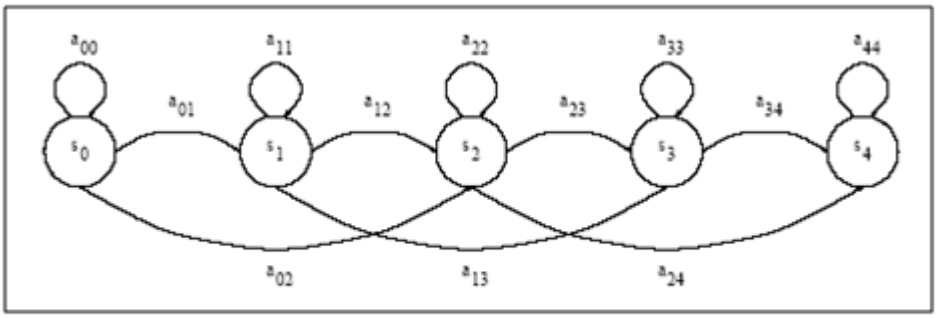


Figure 3.2: Typical lefttoright HMM for word recognition

Note that by choosing nonzero selftransitions probabilities $\alpha_{\theta,\theta}$, vector sequences of infinite length can generated, the shortest length of the sequence is defined by the shortest path, the shortest vector sequence must have three vectors.

3.4 Emission Probability Density of a Vector Sequence

We remember that the HMM will generate the vector sequence $\tilde{X}$ along a state index sequence = $\theta_0, \theta_1, \ldots \theta_{(T_X-1)}$. This is done by simply multiplying all the emission probability densities and transition probabilities along the state sequence :

$p\left(\tilde{X}, \Theta \mid \Lambda\right) = u_{\theta_0} . p(\vec{x}_0 \mid s_{\theta 0}) \cdot \prod_{t=1}^{T_X-1} (\alpha_{\theta(t-1)}, \theta_t$. p ()

Then, p $\left\{\tilde{X} \mid \Lambda\right\}$ can be computed as:

$$p(\tilde{X} \mid \Lambda) = \sum_{\Theta \varepsilon \Omega_{(N^{(T_X)})}} p(\tilde{X}, \Theta \mid \Lambda)$$

If we evaluate first one, we have to deal with product terms only, so that we can easily take the logarithm of the individual probability (density) values and replace the product by the sum of those logvalues. Since the probability density values may vary over several orders of magnitude, taking the logarithm has the additional advantage of reducing the dynamic range of those values. The logarithm of the probability density is often called the log score value.

Fig. 3.3 shows all the possible state sequences or paths (also called the trellis) for the HMM shown in Fig. 3.2. In this Figure, the HMM is rotated by 90 degrees and on the horizontal axis the vectors of the utterance X are shown. At every time index t, each possible path will go through one of the N states of the HMM. In other words, at every grid point shown in Fig. 3.3, a path recombination takes place. If we are using the log score values for the transition probabilities and the emission probability densities, the evaluation of first one is transformed to adding up all the log score values along a given path, in analogy to adding all the local distances along the path.

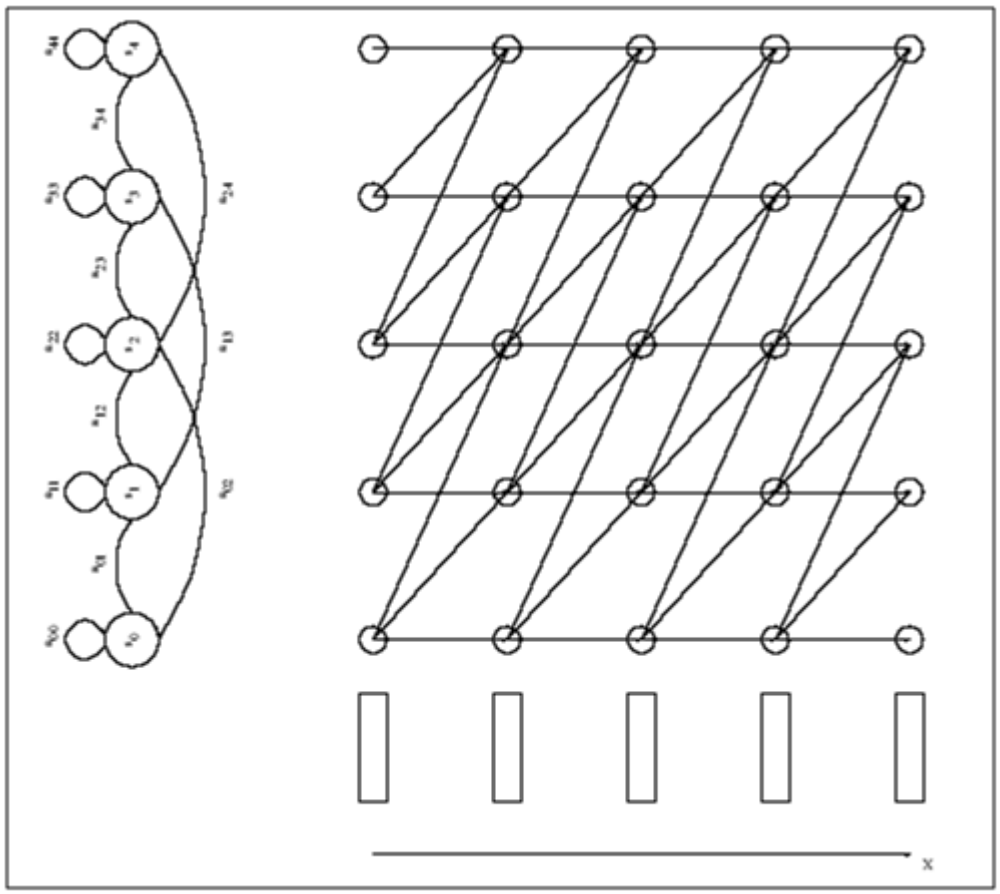


Figure 3.3: Possible state sequences through the HMM

3.5 The Viterbi Algorithm

Now, we can formulate the Viterbi (or DP) algorithm for computing the score $\log_{(p(\tilde{X},\Theta_{opt}|\Lambda))}$. To enable the backtracking of the state sequence Θ_{opt}, we will use the backtracking variable $\Psi(\theta$, t) in analogy.

- Initialization (t = 0):

  For the lefttoright structure we use, we define $u_0 = 1$. Therefore, all other initial state probabilities are zero, enforcing all paths to begin in state s_0 with the emission of the vector $_0$:

- Initialize first column: $\delta(s_\theta,0) = \begin{cases} b(0,s_0), \theta = 0 \\ -\infty, \theta = 1,2,\ldots\ldots,(N-1) \end{cases}$
- Initialize backtracking variable: $\Psi(\theta,0) = -1; \theta = 0,1,\ldots\ldots,(N-1)$
- Iteration: Perform the local path recombination as given by (7.14) for all states $\theta = 0,1,\ldots\ldots,(N-1)$ and all time indices t = 1, 2,$\ldots\ldots$, $(TX-1)$:

  △ Optimization step:

  $\delta(s_\theta,t) = b(t,s_\theta) + \max_{i\varepsilon\equiv(s\theta)}\{\, d(s_i,s_\theta) + \delta(s_i,t-1)$

  △ Backtracking:

  $\Psi(\theta,t) = \arg\max_{i\varepsilon\equiv(s\theta)}\{\, d(s_i,s_\theta) + \delta(s_i,t-1)$

- Termination: In our lefttoright model, we will consider only paths which end in the last state of the HMM, the state sN1. After the last iteration step is done, we get:

  $\log_{(p(\tilde{X},\Theta_{opt}|\Lambda))} = \delta(s_{N-1},(T_X)-1)$

- Backtracking procedure: Beginning with time index (T_X 1), go backwards in time through the backtracking variables:

  △ Initialization:

  t = $(T_X - 1); \theta_{(T_X-1)} = N-1$

  △ Termination:

  $\theta_0 = \Psi(\theta_1,1) \equiv 0$

3.6 Gaussian Probability Density Function

In our application, we measure feature vectors in a multidimensional feature space, so we will use a multivariate Gaussian PDF, which looks like this:

$$\phi(\vec{x}) = \frac{1}{\sqrt{(2\pi)^{DIM} \cdot |\tilde{C}|}} \cdot e^{(-\frac{1}{2}(\vec{x}-\vec{m})' \cdot \tilde{C}^{-1} \cdot (\vec{x}-\vec{m}))}$$

Where $\vec{m}$ denotes the mean vector and $\tilde{C}$ denotes the covariance matrix.

3.6.1 Continuous Mixture Densities

The Gaussian PDF equation can characterize the observation probability for vectors generated by a single Gaussian process. The PDF of this process has a maximum value at the position of the mean vector $\tilde{m}$ and its value exponentially decreases with increasing distance from the mean vector. The regions of constant probability density are of elliptical shape, and their orientation is determined by the Eigenvectors of the covariance matrix. However, for speech recognition, we would like to model more complex probability distributions, which have more than one maximum and whose regions of constant probability density are not elliptically shaped.

Here scatterplot of an emission process consisting of three Gaussian emission processes. The process parameters are:

$$c_1 = 0.3; \tilde{C_1} = \begin{bmatrix} 5 & 2 \\ 2 & 2 \end{bmatrix}; \vec{m_1} = [1, 1.5]'$$

$$c_2 = 0.4; \tilde{C_2} = \begin{bmatrix} 1 & -1 \\ 1 & 5 \end{bmatrix}; \vec{m_1} = [-3, -1]'$$

$$c_3 = 0.5; \tilde{C_3} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \vec{m_1} = [3, -3]'$$

The corresponding probability density function is visualized as a surface in Figure. 3.4.

The complexity of the PDF can be influenced by selecting the appropriate number of mixtures K and may be set individually for every state of the HMM. However, since a high number of mixtures also means that a high number of parameters has to be estimated from a

given training set of data, always some compromise between the granularity of the modeling and the reliability of the estimation of the parameters has to be found.

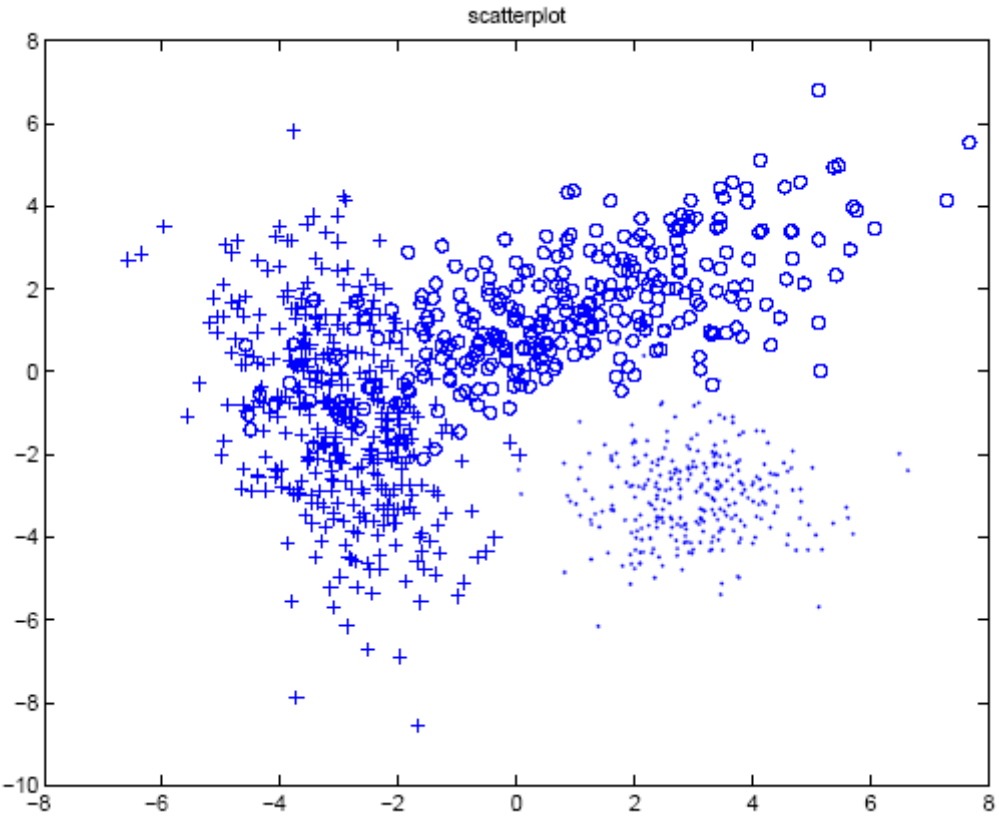


Figure 3.4: scatterplot of vectors emitted by a gaussian mixture density process

3.7 Conclusion

The core elements are the most common HMM-based approach to speech recognition. Modern speech recognition systems use various combinations of a number of standard techniques in order to improve results over the basic approach described above. A typical large-vocabulary system would need context dependency for the phonemes (so phonemes with different left and right context have different realizations as HMM states); it would use cepstral normalization to normalize for different speaker and recording conditions; for further speaker normalization it might use vocal tract length normalization (VTLN) for male-female normalization and maximum likelihood linear regression (MLLR) for more general speaker adaptation. The features would have so-called delta and delta-delta coefficients to capture speech dynamics and in addition might use heteroscedastic linear discriminant anal-

ysis (HLDA); or might skip the delta and delta-delta coefficients and use splicing and an LDA-based projection followed perhaps by heteroscedastic linear discriminant analysis or a global semitied covariance transform (also known as maximum likelihood linear transform, or MLLT). Many systems use so-called discriminative training techniques which dispense with a purely statistical approach to HMM parameter estimation and instead optimize some classification-related measure of the training data. Examples are maximum mutual information (MMI), minimum classification error (MCE) and minimum phone error (MPE).

Decoding of the speech (the term for what happens when the system is presented with a new utterance and must compute the most likely source sentence) would probably use the Viterbi algorithm to find the best path, and here there is a choice between dynamically creating a combination hidden Markov model which includes both the acoustic and language model information, or combining it statically beforehand (the finite state transducer, or FST, approach).

CHAPTER 4
BANGLA IPA TABLE

4.1 Introduction

The International Phonetic Alphabet (IPA) is an alphabetic system of phonetic notation based primarily on the Latin alphabet. It was devised by the International Phonetic Association as a standardized representation of the sounds of spoken language. The IPA is used by foreign language students and teachers, linguists, speech pathologists and therapists, singers, actors,lexicographers, artificial language enthusiasts (conlangers), and translators. The IPA is designed to represent only those qualities of speech that are distinctive in spoken language: phonemes, intonation, and the separation of words and syllables. To represent additional qualities of speech such as tooth gnashing, lisping, and sounds made with a cleft palate, an extended set of symbols called the Extensions to the IPA may be used.

IPA symbols are composed of one or more elements of two basic types, letters and diacritics. For example, the sound of the English letter t may be transcribed in IPA with a single letter, [t], or with a letter plus diacritics, [t^h], depending on how precise one wishes to be.Occasionally letters or diacritics are added, removed, or modified by the International Phonetic Association. As of 2008, there are 107 letters, 52 diacritics, and four prosodic marks in the IPA [10].

In 1886, a group of French and British language teachers, led by the French linguist Paul Passy, formed what would come to be known from 1897 onwards as the International Phonetic Association (in French, *l'Association phontique internationale*). Their original alphabet was based on a spelling reform for English known as the Romic alphabet, but in order to make it usable for other languages, the values of the symbols were allowed to vary from language to language. For example, the sound [$\int$] (the sh in shoe) was originally represented with the letter <c >in English, but with the letter <x >in French. However, in 1888, the

alphabet was revised so as to be uniform across languages, thus providing the base for all future revisions.

Since its creation, the IPA has undergone a number of revisions. After major revisions and expansions in 1900 and 1932, the IPA remained unchanged until the IPA Kiel Convention in 1989. A minor revision took place in 1993, with the addition of four letters for mid-central vowels and the removal of letters for voiceless implosives. The alphabet was last revised in May 2005, with the addition of a letter for a labiodentals flap. Apart from the addition and removal of symbols, changes to the IPA have consisted largely in renaming symbols and categories and in modifying typefaces. Extensions of the alphabet are relatively recent; "Extensions to the IPA" was created in 1990 and officially adopted by the International Clinical Phonetics and Linguistics Association in 1994 [10].

4.2 Bangla Script

The Bengali script (Bengali: bangla lipi or Bengali: b*ôgô*lipi) is the writing system for the Bengali language. It is also used, with some modifications, for Assamese, Meitei, Bishnupriya, Manipuri, Kokborok, Garo and Mundari languages. All these languages are spoken in the eastern region of South Asia. Historically, the script has also been used to write the Sanskrit language in the same region. From a classificatory point of view, the Bengali script is an abugida, i.e. its vowel graphemes are mainly realized not as independent letters like in a true alphabet, but as diacritics attached to its consonant graphemes. It is written from left to right and lacks distinct letter cases. It is recognizable by a distinctive horizontal line running along the tops of the letters that links them together, a property it shares with two other popular Indian scripts: Devanagari (used for Hindi, Marathi and Nepali) and Gurumukhi (used for Punjabi). The Bengali script is,however, less blocky and presents a more sinuous shaping [11].

The Bengali script evolved from the Eastern Nagari script, which belongs to the Brahmic family of scripts, along with the Devanagari script and other written systems of the Indian subcontinent. Both Eastern Nagari and Devanagari were derived from the ancient Nagari script.In addition to differences in how the letters are pronounced in the different languages, there are some minor typographical differences between the version of the script used

for Assamese and Bishnupriya Manipuri as well as Maithili languages, and that used for Bengali and other languages. For example, the letter r*ô* (Bengali র ; Assamese ৰ ; Bishnupriya Manipuri/Maithili র / ৰ) and w*ô* (Bengali second ব ; Assamese/Bishnupriya Manipuri/Maithili ৱ) have distinct variations depending on the language being written.

The Bengali script was originally not associated with any particular language, but was often used in the eastern regions of Medieval India. It was standardized and modernized by Ishwar Chandra under the reign of the British East India Company. The script was originally used to write Sanskrit, which for centuries was the only written language of the Indian subcontinent in addition to Tamil. Epics of Hindu scripture, including the Mahabharata or Ramayana, were written in older versions of the Bengali script or Mithilakshar/Tirhuta script in this region. After the medieval period, the use of Sanskrit as the sole written language gave way to Pali, and eventually to the vernacular languages we know now as Maithili, Bengali, and Assamese.Srimanta Sankardeva used it in the 15th and 16th centuries to compose his oeuvre in Assamese and Brajavali the language of the Bhakti poets. There is a rich legacy of Indian literature written in this script, which is still occasionally used to write Sanskrit today [11].

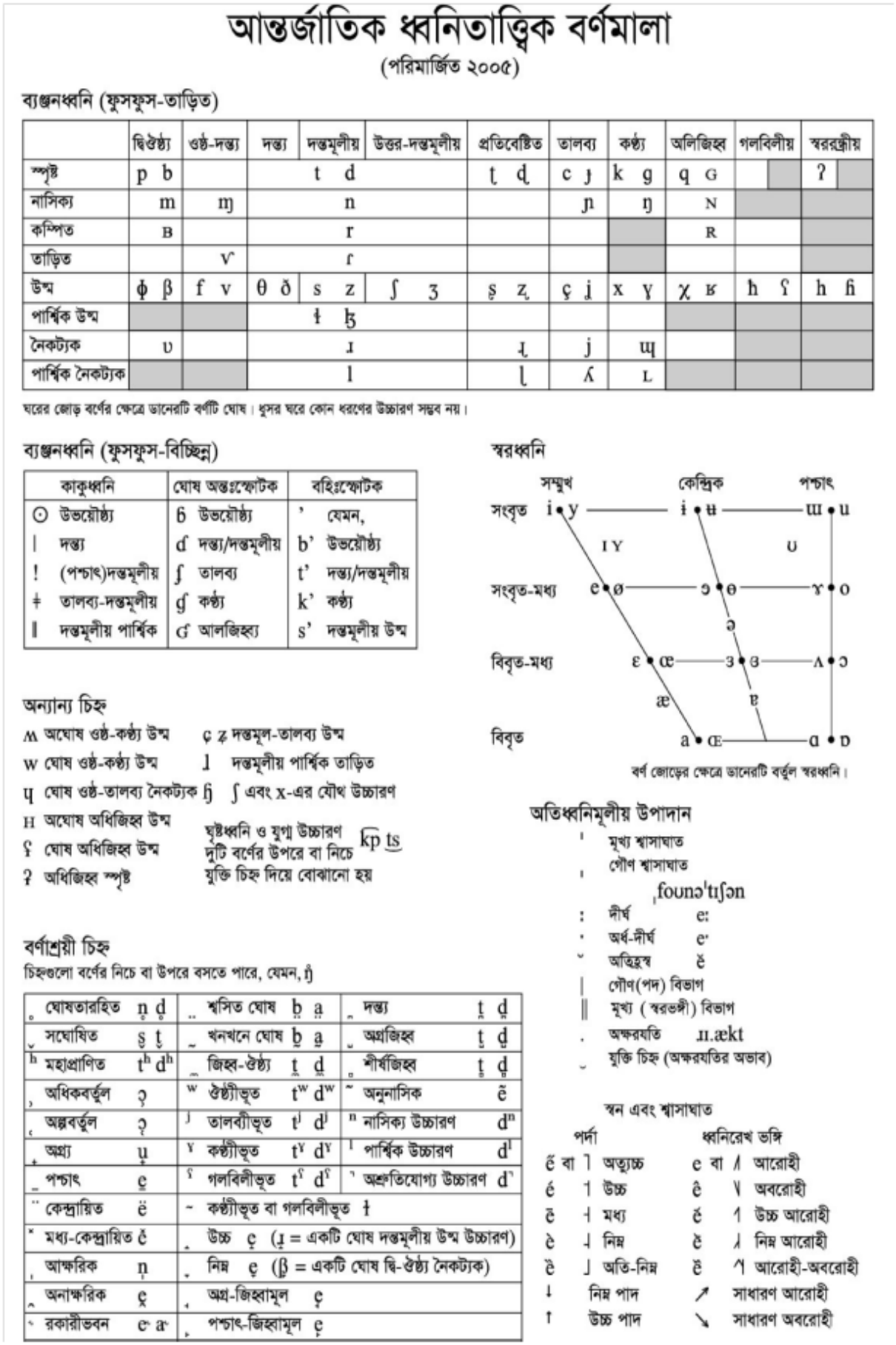
4.3 Bangla IPA Table

The first IPA chart was prepared in 1888 by the earliest form of the International Phonetic Association and it has gone through many changes since then. The 1888 chart was rather a list of symbols and their descriptions [12].

The Figure of 4.1 is the latest, revised 2005, in Bengali :

4.3.1 Vowels & Consonants

The phonemic inventory of Bangla consists of 29 consonants and 14 vowels, including the seven nasalized vowels. Several conventions exist for Romanizing Indic languages, including IAST (based on diacritics), ITRANS (uses upper case letters suited for ASCII keyboards), and the National Library at Calcutta Romanization. Bangla words are currently romanized on Wikipedia using a phonemic transcription, where the pronunciation is represented with no reference to the spelling. The Wikipedia Romanization is given in the



আন্তর্জাতিক ধ্বনিতাত্ত্বিক বর্ণমালা

(পরিমার্জিত ২০০৫)

ব্যঞ্জনধ্বনি (ফুসফুস-তাড়িত)

	দ্বিওষ্ঠ্য	ওষ্ঠ-দন্ত্য	দন্ত্য	দন্তমূলীয়	উত্তর-দন্তমূলীয়	প্রতিবেষ্টিত	তালব্য	কণ্ঠ্য	অলিজিহ্ব	গলবিলীয়	স্বররন্ধ্রীয়
স্পৃষ্ট	p b			t d		ʈ ɖ	c ɟ	k g	q G		ʔ
নাসিক্য	m	ɱ		n			ɲ	ŋ	N		
কম্পিত	ʙ			r					R		
তাড়িত		ⱱ		ɾ							
উষ্ম	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
পার্শ্বিক উষ্ম				ɬ ɮ							
নৈকট্যক	ʋ			ɹ		ɻ	j	ɰ			
পার্শ্বিক নৈকট্যক				l		ɭ	ʎ	ʟ			

ঘরের জোড় বর্ণের ক্ষেত্রে ডানেরটি বর্ণটি ঘোষ। ধূসর ঘরে কোন ধরণের উচ্চারণ সম্ভব নয়।

ব্যঞ্জনধ্বনি (ফুসফুস-বিচ্ছিন্ন)

কাকুধ্বনি	ঘোষ অন্তঃস্ফোটক	বহিঃস্ফোটক
ʘ উভয়ৌষ্ঠ্য	ɓ উভয়ৌষ্ঠ্য	’ যেমন,
ǀ দন্ত্য	ɗ দন্ত্য/দন্তমূলীয়	b’ উভয়ৌষ্ঠ্য
ǃ (পশ্চাৎ)দন্তমূলীয়	ʄ তালব্য	t’ দন্ত্য/দন্তমূলীয়
ǂ তালব্য-দন্তমূলীয়	ɠ কণ্ঠ্য	k’ কণ্ঠ্য
ǁ দন্তমূলীয় পার্শ্বিক	ʛ আলজিহ্ব্য	s’ দন্তমূলীয় উষ্ম

স্বরধ্বনি

বর্ণ জোড়ের ক্ষেত্রে ডানেরটি বর্তুল স্বরধ্বনি।

অন্যান্য চিহ্ন

ʍ অঘোষ ওষ্ঠ-কণ্ঠ্য উষ্ম
w ঘোষ ওষ্ঠ-কণ্ঠ্য উষ্ম
ɥ ঘোষ ওষ্ঠ-তালব্য নৈকট্যক
ʜ অঘোষ অধিজিহ্ব উষ্ম
ʢ ঘোষ অধিজিহ্ব উষ্ম
ʡ অধিজিহ্ব স্পৃষ্ট

ɕ ʑ দন্তমূল-তালব্য উষ্ম
ɺ দন্তমূলীয় পার্শ্বিক তাড়িত
ɧ ʃ এবং x-এর যৌথ উচ্চারণ

ঘৃষ্টধ্বনি ও যুগ্ম উচ্চারণ
দুটি বর্ণের উপরে বা নিচে
যুক্তি চিহ্ন দিয়ে বোঝানো হয়
k͡p t͡s

বর্ণাশ্রয়ী চিহ্ন

চিহ্নগুলো বর্ণের নিচে বা উপরে বসতে পারে, যেমন, n̥ d̥

̥ ঘোষতারহিত	n̥ d̥	̤ শ্বসিত ঘোষ	b̤ a̤	̪ দন্ত্য	t̪ d̪
̬ সঘোষিত	s̬ t̬	̰ খনখনে ঘোষ	b̰ a̰	̺ অগ্রজিহ্ব	t̺ d̺
ʰ মহাপ্রাণিত	tʰ dʰ	̼ জিহ্ব-ওষ্ঠ্য	t̼ d̼	̻ শীর্ষজিহ্ব	t̻ d̻
̹ অধিকবর্তুল	ɔ̹	ʷ ওষ্ঠ্যীভূত	tʷ dʷ	̃ অনুনাসিক	ẽ
̜ অল্পবর্তুল	ɔ̜	ʲ তালব্যীভূত	tʲ dʲ	ⁿ নাসিক্য উচ্চারণ	dⁿ
̟ অগ্র্য	u̟	ˠ কণ্ঠ্যীভূত	tˠ dˠ	ˡ পার্শ্বিক উচ্চারণ	dˡ
̠ পশ্চাৎ	e̠	ˤ গলবিলীভূত	tˤ dˤ	˞ অগ্রতিযোগ্য উচ্চারণ	d˞
̈ কেন্দ্রায়িত	ë	̴ কণ্ঠ্যীভূত বা গলবিলীভূত		ɫ	
̽ মধ্য-কেন্দ্রায়িত	e̽	̝ উচ্চ	e̝ (ɪ̝ = একটি ঘোষ দন্তমূলীয় উষ্ম উচ্চারণ)		
̩ আক্ষরিক	n̩	̞ নিম্ন	e̞ (β̞ = একটি ঘোষ দ্বি-ওষ্ঠ্য নৈকট্যক)		
̯ অনাক্ষরিক	e̯	̘ অগ্র-জিহ্বামূল	e̘		
˞ রকারীভবন	e˞ a˞	̙ পশ্চাৎ-জিহ্বামূল	e̙		

অতিধ্বনিমূলীয় উপাদান

ˈ মুখ্য শ্বাসাঘাত
ˌ গৌণ শ্বাসাঘাত
ˌfounəˈtɪʃən
ː দীর্ঘ eː
ˑ অর্ধ-দীর্ঘ eˑ
˘ অতিহ্রস্ব ĕ
| গৌণ(পদ) বিভাগ
‖ মুখ্য (স্বরভঙ্গী) বিভাগ
. অক্ষরযতি ɹ.ækt
‿ যুক্তি চিহ্ন (অক্ষরযতির অভাব)

সুর এবং শ্বাসাঘাত

পর্দা			ধ্বনিরেখ ভঙ্গি	
˩ বা ˥̀	অত্যুচ্চ	e̋ বা e᷄	˩˥	আরোহী
˥	উচ্চ	é	˥˩	অবরোহী
˧	মধ্য	ē	˧˥	উচ্চ আরোহী
˨	নিম্ন	è	˨˩	নিম্ন আরোহী
˩	অতি-নিম্ন	ȅ	˩˥˩	আরোহী-অবরোহী
↓	নিম্ন পাদ	↗		সাধারণ আরোহী
↑	উচ্চ পাদ	↘		সাধারণ অবরোহী

Figure 4.1: A chart of the full International Phonetic Alphabet

table below, along with IPA transcriptions above them.

An approximate phonetic scheme in IPA is given in Figure 4.3. In Figure 4.2, only the main 7 vowel sounds are shown, though there exists two more long counterpart of /i/ and /u/, denoted as /i:/ and /u:/, respectively. These two long vowels are seldom pronounced differently than their short counterparts in modern Bangla. There is controversy on the number of Bangla consonants.

	Front	Central	Back
Close	i i		u u
Close-mid	e e		o o
Open-mid	æ ê		
Open		a a	

Figure 4.2: Bangla phonetic scheme in IPA (Vowels)

4.4 Bangla Phoneme Schemes

4.4.1 Bangla Phoneme

The Phonetic inventory of Bangla consists of 14 vowels, including seven nasalized vowels, and 29 consonants. An approximate phonetic scheme in IPA is given in Figure 4.3. In Table 4.2, only the main 7 vowel sounds are shown, though there exists two more long counterpart of /i/ and /u/, denoted as /i:/ and /u:/, respectively. These two long vowels are seldom pronounced differently than their short counterparts in modern Bangla. There is controversy on the number of Bangla consonants.Native Bangla words do not allow initial

		Labial	Dental/ Alveolar	Retroflex	Palato-alveolar	Velar	Glottal
Nasal		m m	n n			ŋ ng	
Stop	tenuis	p~ɸ p	t̪ t	ʈ ṭ	tʃ~ts c	k~χ k	
	aspirated	pʰ~ɸ ph	t̪ʰ th	ʈʰ ṭh	tʃʰ~tsʰ ch	kʰ~χ kh	
	voiced	b~β b	d̪ d	ɖ ḍ	dʒ~dz j	g~ɣ g	
	murmured	bʰ~β bh	d̪ʰ~d̪ dh	ɖʰ~ɖ ḍh	dʒʰ~dz jh	gʰ~ɣ gh	
Fricative		(f) (f)	(s , z) (s, z)		ʃ sh		h~ɦ h
Approximant		(w)	l l		(j)		
Rhotic			r r	ɽ ṛ			

Figure 4.3: Bangla phonetic scheme in IPA (Consonants)

consonant clusters: the maximum syllable structure is CVC (i.e. one vowel flanked by a consonant on each side) [13]. Sanskrit words borrowed into Bangla possess a wide range of clusters, expanding the maximum syllable structure to CCCVC. English or other foreign borrowings add even more cluster types into the Bangla inventory.

4.5 Conclusion

In the Bengali script, clusters of consonants are represented by different and sometimes quite irregular characters; thus, learning to read the script is complicated by the sheer size of the full set of characters and character combinations, numbering about 350. While efforts at standardizing the script for the Bengali language continue in such notable centers as the Bangla Academies (unaffiliated) at Dhaka (Bangladesh) and Kolkata (West Bengal, India), it is still not quite uniform as yet, as many people continue to use various archaic forms of letters, resulting in concurrent forms for the same sounds. Among the various regional variations within this script, only the Assamese and Bengali variations exist today in the formalized system. It seems likely that the standardization of the script will be greatly influenced by the need to typeset it on computers. The large alphabet can be represented, with a great deal of ingenuity, within the ASCII character set, omitting certain irregular conjuncts. Work has been underway since around 2001 to develop Unicode fonts, and it seems likely that it will split into two variants, traditional and modern.

In this and other articles on Wikipedia dealing with the Bengali language, a Romanization scheme used by linguists specializing in Bengali phonology is included along with IPA transcription.

A recent effort by the government of West Bengal focused on simplifying Bengali spellings in primary school texts [11].

CHAPTER 5
OUR CONTRIBUTION

5.1 Introduction

There have been many literatures in automatic speech recognition (ASR) systems for almost all the major spoken languages in the world. Unfortunately, only a very few works have been done in ASR for Bangla (can also be termed as Bengali), which is one of the largely spoken languages in the world. More than 220 million people speak in Bangla as their native language. It is ranked seventh based on the number of speakers [14]. A major difficulty to research in Bangla ASR is the lack of proper speech corpus. Some efforts are made to develop Bangla speech corpus to build a Bangla text to speech system [15]. However, this effort is a part of developing speech databases for Indian Languages, where Bangla is one of the parts and it is spoken in the eastern area of India (West Bengal and Kolkata as its capital). But most of the natives of Bangla (more than two thirds) reside in Bangladesh, where it is the official language. Although the written characters of Standard Bangla in both the countries are same, there are some sound that are produced variably in different pronunciations of Standard Bangla, in addition to the myriad of phonological variations in non-standard dialects [16]. Therefore, there is a need to do research on the main stream of Bangla, which is spoken in Bangladesh, ASR.

In this paper, we build an ASR system for Bangla phoneme in a large scale. For this purpose, we first develop a medium size (compared to the exiting size in Bangla ASR literature) Bangla speech corpus comprises of native speakers covering almost all the major cities of Bangladesh. Then, Mel Frequency Cepstral Coefficients (MFCCs) are extracted from the input speech, and finally extracted features are inserted into the hidden Markov model (HMM) based classifier for obtaining the phoneme recognition performance.

5.2 Why is ASR hard to accomplish?

There are various reasons why this is a complicated task. Speech, captured in speech signals, has a rich variation between different speakers. Physiological aspects such as length of the vocal cords, shape of the mouth cavity, position of the teeth all influence the speech signal that a speaker produces. Other factors that differentiate speech include social aspects. Native speakers can hear differences between standard Bangla pronounced by speakers from different provinces in the Comilla and even between speakers from certain cities. An acquaintance from the city of Dhaka went to the US and surprisingly was recognized there as originating from Dhaka simply by his specific pronunciation of the letter r. Humans can also distinguish between male and female speech and even between speeches from different age categories. For example, when we answer the phone, we can tell whether it is a child or an elderly person calling. All these aspects influence the speech signal and complicate matters for a computer. An ASR system should be able to recognize all these different pronunciations of the same sounds and then conclude that they are in fact the same sounds.

But even when one speaker pronounces the same word twice, there will be differences in the speech signal. If a speaker has a cold, he sounds different. His emotions also influence the pronunciation. Speech from a certain speaker is even different at different times of the day. The duration of the different sounds that constitute the word in question, will also vary slightly. Because speaking involves a continuous motion of the articulators —tongue, teeth, lips— the sounds themselves even differ depending on which sounds precede and follow it. This effect is known as coarticulation.

Besides the influence of the speaker, there also are external factors that complicate matters. There will be different amounts of noise that obscure the speech signal and different input devices and encoding schemes that influence the speech signal. This is known as channel variability.

Apart from these more acoustical problems, there also are other complications. When looking at a transcript of spontaneous speech, it can readily be seen that there are many repetitions, filler words, mispronunciations, errors that are being corrected and other mistakes that complicate matters. Very often, spontaneous sentences will not be grammatically correct.

Another difficulty is the vocabulary. The vocabulary of a modern language changes continuously, with new words entering, old ones becoming obsolete and fashion words occur and disappear sometimes so fast that they do not even become an entry in the dictionary. And current speech recognizers rely on dictionaries to determine which sound sequences form a word in a specific language and which do not. Certain languages, such as Bangla, also allow the formation of a new word by combining two existing words, following certain rules. This makes it hard to capture a vocabulary of a language even without the constant changes.

Many of these problems can be categorized under the header of 'too much variation', on very different areas. During this research, we will not try to tackle all these problems simultaneously but focus on the acoustic variation between speakers. Currently, the acoustic variation between speakers of different characteristics is mostly ignored. Speech recognizers are trained on corpora of speech from a balanced group of speakers, using statistical averages calculated over the group as a whole to tune the acoustic speech models to. These models can then be used to recognize speech from all speakers. Limiting this variation by creating specialized acoustic speech models for different speaker groups could improve the performance of ASR systems. Most current ASR systems already include provisions for different genders. However, it is indicated in the literature of phonetics that age also influences the speech production organs and thus leads to differences in speech acoustics. Indeed, human listeners are able to differentiate between speech from speakers with different genders and even ages. A special ASR system was also created for Swedish children. However, the effects of these biological factors have not yet been researched for Bangla speakers. We will thus look at the acoustical differences between different age categories and genders specifically for the Bangla language, using the speech data from the recently finished Corpus (the spoken Bangla corpus). This will allow us to conclude whether different models for different age groups could be helpful for ASR of the Bangla language.

5.3 Preparation of Bangla Speech Corpus

There have been many literatures in automatic speech recognition (ASR) systems for almost all the major spoken languages in the world. Unfortunately, only a very few works have been done in ASR for Bangla (can also be termed as Bengali), which is one of the largely spoken languages in the world. More than 220 million people speak in Bangla as their native

language. It is ranked seventh based on the number of speakers [14]. A major difficulty to research in Bangla ASR is the lack of proper speech corpus. Some efforts are made to develop Bangla speech corpus to build a Bangla text to speech system [15]. However, this effort is a part of developing speech databases for Indian Languages, where Bangla is one of the parts and it is spoken in the eastern area of India (West Bengal and Kolkata as its capital). But most of the natives of Bangla (more than two thirds) reside in Bangladesh, where it is the official language. Although the written characters of Standard Bangla in both the countries are same, there are some sound that are produced variably in different pronunciations of Standard Bangla, in addition to the myriad of phonological variations in non-standard dialects [16]. Therefore, there is a need to do research on the main stream of Bangla, which is spoken in Bangladesh, ASR.

At present, a real problem to do experiment on Bangla phoneme ASR is the lack of proper Bangla speech corpus. In fact, such a corpus is not available or at least not referenced in any of the existing literature. Therefore, we develop a medium size Bangla speech corpus, which is described below.

Hundred sentences from the Bengali newspaper "Prothom Alo"[17] are uttered by 30 male speakers of different regions of Bangladesh. These sentences (30x100) are used for training corpus (D1). On the other hand, different 100 sentences from the same newspaper uttered by 10 different male speakers (total 1000 sentences) are used as test corpus (D2). All of the speakers are Bangladeshi nationals and native speakers of Bangla. The age of the speakers ranges from 20 to 40 years. We have chosen the speakers from a wide area of Bangladesh: Dhaka (central region), Comilla Noakhali (East region), Rajshahi (West region), Dinajpur Rangpur (North-West region), Khulna (South-West region), Mymensingh and Sylhet (North-East region). Though all of them speak in standard Bangla, they are not free from their regional accent.

Recording was done in a quiet room located at United International University (UIU), Dhaka, Bangladesh. A desktop was used to record the voices using a head mounted close-talking microphone. We record the voice in a place, where ceiling fan and air conditioner were switched on and some low level street or corridor noise could be heard.

Jet Audio 7.1.1.3101 software was used to record the voices. The speech was sampled at 16 kHz and quantized to 16 bit stereo coding without any compression and no filter is used on

the recorded voice.

5.4 Feature Extraction and Phonemes

When performing speech recognition, the first step is to analyze the given sound wave and extract relevant data from it. This signal processing starts by dividing the sound wave in time slices from which spectral features are extracted. Spectral features indicate how much energy is present in the signal at different frequencies.Sounds are first in .wav format and then it is converted to MFCC format.

WAV File Format

The WAV file format is a subset of Microsofts RIFF specification for the storage of multimedia files. A RIFF file starts out with a file header followed by a sequence of data "chunks". A WAV file is often just a RIFF file with a single "WAVE"chunk which consists of two sub-chunks - a "fmt"chunk specifying the data format and a "data"chunk containing the actual sample data. The WAV file header contains the following:

'RIFF' : RIFF file identification (4 bytes)
<length > : length field (4 bytes)
'WAVE' : WAVE chunk identification (4 bytes)
'fmt' : format sub-chunk identification (4 bytes)
flength : length of format sub-chunk (4 byte integer)
format : format specifier (2 byte integer)
chans : number of channels (2 byte integer)
sampsRate : sample rate in Hz (4 byte integer)
bpsec : bytes per second (4 byte integer)
bpsample : bytes per sample (2 byte integer)
bpchan : bits per channel (2 byte integer)
'data' : data sub-chunk identification (4 bytes)
dlength : length of data sub-chunk (4 byte integer)

Mel-frequency cepstral coefficients (MFCCs)

In sound processing, the mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum. This frequency warping can allow for better representation of sound, for example, in audio compression. MFCCs are commonly derived as follows:

1. Take the Fourier transform of (a windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows.
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosine transform of the list of mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

Later on, these features are used to determine which sounds were uttered. Instead of the general word 'sound 'we talk about a 'phoneme '. A phoneme is defined as:

Any of the minimal units of speech sound in a language that can serve to distinguish one word from another. For example, /p/ is a phoneme in the English language, distinguishing tap from tag. These spectral features are used in combination with hidden Markov models (HMMs) to determine the probabilities for the different phonemes of having been uttered, while taking the speech waveform into consideration. These calculations are explained in more detail in the next section.

5.5 Bangla Phonemes with Phonetic Symbols

Phonetic inventory of Bangla consists of 8 short vowels, excluding long vowels, and 29 consonants. Table 5.1 shows Bangla vowel phonemes with their corresponding International

Phonetic Alphabet (IPA) and our proposed symbols. On the other hand, the consonants, which are used in Bangla language, are presented in Table 5.2. Here, the table exhibits the same items for consonants like as Table 5.1. In the Table 5.2, the pronunciation of /শ / , /ষ / , /স / are same by considering the words বিশ (/bi∫/) and ডিস (/di∫/) .On the other hand, in the words জাম (/dʒam/) and যাক (/dʒak/) there is no difference of pronunciation of /জ / and /য / . Again , there is no difference of / ন / and / ণ / in the words হরিণ (/hrin/) and তিন (/tin/).Phonemes /ড় / / and /ঢ় / carry same pronunciation in the words পাহাড় (/pahar/) and আষাঢ় (/a∫ar/) .

Table 5.1: Bangla Vowels

Letter	IPA	Our Symbol
অ	/ɔ /*and*/*o*/	a
আ	/a/	aa
ই	/i/	i
ঈ	/i/	i
উ	/u/	u
ঊ	/u/	u
এ	/e/ and /æ/	e
ঐ	/oj/	oi
ও	/o/	o
ঔ	/ow/	ou

Table 5.2: Bangla Consonants

Letter	IPA	Our Symbol
ক	/k/	k
খ	/k^h/	kh
গ	/g/	g
ঘ	/g^h/	gh
ঙ	/ŋ/	ng
চ	/t∫ /	ch
ছ	/t$∫^h$ /	chh

জ	/dʒ/	j
ঝ	/dʒ^h/	jh
ট	/t/	ta
ঠ	/t^h/	th
ড	/d/	da
ঢ	/d^h/	dha
ণ	/n/	N
ত	/t̲/	t
থ	/t^h/	th
দ	/d̲	D
ধ	/d̲^{h}/	Dh
ন	/n/	N
প	/p/	p
ফ	/p^h/	ph
ব	/b/	B
ভ	/b^h/	Bh
ম	/m/	M
য	/dʒ/	J
র	/r/	R
ল	/l/	L
শ	/ʃ//s/	S
ষ	/ʃ/	S
স	/ʃ/ /s/	S
হ	/h/	H
ড়	/h̲/	Rh
ঢ়	/r̲/	Rh
য়	/e̲/	Y

Native Bangla words do not allow initial consonant clusters: the maximum syllable structure is CVC (i.e. one vowel flanked by a consonant on each side). Sanskrit words borrowed

into Bangla possess a wide range of clusters, expanding the maximum syllable structure to CCCVC. English or other foreign borrowings add even more cluster types into the Bangla inventory.

5.6 Bangla Words

Table 5.3 lists some Bangla words with their written forms and the corresponding IPA.From the table, it is shown that the same 'Av '(/a/) has different pronunciation based on succeeding phonemes 'g ', 'P 'and 'e'. These pronunciations are sometimes long or short. For long and short 'Av 'we have used two different phonemes /aa/ and /ax/, respectively. Similarly, we have considered all variations of same phonemes and consequently, found total 51 phonemes excluding beginning and end silence (/sil/) and short pause (/sp/).

Table 5.3: Some Bangla words with their orthographic transcriptions and IPA

Bangla Word	English pronounciation	IPA	Our Symbol
আমরা	AAMRA	/a m r a/	/aa m r ax/
আচরণ	AACHORON	/a t ʃ r n/	/ aa ch ow r aa n/
আবেদন	ABEDON	/a b æd̲ n/	/ax b ae d aa n/

5.7 HTK Tool

Much of the functionality of HTK is built into the library modules. These modules ensure that every tool interfaces to the outside world in exactly the same way. They also provide a central resource of commonly used functions.HTK tools are designed to run with a traditional command-line style interface. Each tool has a number of required arguments plus optional arguments [18] .

5.7.1 HCopy

This program will copy one or more data files to a designated output file, optionally converting the data into a parameterised form. While the source files can be in any supported

format, the output format is always HTK. By default, the whole of the source file is copied to the target but options exist to only copy a specified segment. Hence, this program is used to convert data files in other formats to the HTK format, to concatenate or segment data files, and to parameterise the result. If any option is set which leads to the extraction of a segment of the source file rather than all of it, then segments will be extracted from all source files and concatenated to the target.HCopy is used to convert the parameter kind of a file, for example from WAVEFORM to MFCC, depending on the configuration options. The allowable options to HCopy are as follows:

-C is used to specify a configuration file name and the option.

-S is used to specify a script file name.

-T HCopy supports the following trace options where each trace flag is given using an octal base:

00001 basic progress reporting.
00002 source and target file formats and parameter kinds.
00004 segment boundaries computed from label files.
00010 display memory usage after processing each file.

Trace flags are set using the -T option or the TRACE configuration variable.

5.7.2 HList

HList is invoked by typing the command line:
HList [options] file ...
This causes the contents of each file to be listed to the standard output. If no files are given and the source format is HAUDIO, then the audio source is listed. The source form of the data can be converted and listed in a variety of target forms by appropriate settings of the configuration variables, in particular TARGETKIND.HList also supports the standard options -A, -C, -D, -S, -T, and -V.

HList supports the following trace option where each trace flag is given using an octal base
00001 basic progress reporting.
Trace flags are set using the -T option or the TRACE configuration variable.

5.7.3 HCompV

This program will calculate the global mean and covariance of a set of training data. It is primarily used to initialise the parameters of a HMM such that all component means and all covariances are set equal to the global data mean and covariance.This might form the first stage of a *flat start* training scheme where all models are initially given the same parameters. Alternatively, the covariances may be used as the basis for Fixed Variance and Grand Variance training schemes. These can sometimes be beneficial in adverse conditions where a fixed covariance matrix can give increased robustness.

When training large model sets from limited data, setting a floor is often necessary to prevent variances being badly underestimated through lack of data. One way of doing this is to define a variance macro called varFloorN where N is the stream index. HCompV can also be used to create these variance floor macros with values equal to a specified fraction of the global variance. Another application of HCompV is the estimation of mean and variance vectors for use in cluster-based mean and variance normalisation schemes. Given a list of utterances and a speaker pattern HCompV will estimate a mean and a variance for each speaker.

The allowable options to HCompV are as follows:

-C is used to specify a configuration file name and the option.

-S is used to specify a script file name.

-M dir Store output HMM macro model files in the directory dir. If this option is not given, the new HMM definition will overwrite the existing one.

-m The covariances of the output HMM are always updated however updating the means must be specifically requested. When this option is set, HCompV updates all the HMM component means with the sample mean computed from the training files.

-v f This sets the minimum variance (i.e. diagonal elements of the covariance matrix) to the real value f (default value 0.0).

-T HCompV supports the following trace options where each trace flag is given using an octal base:

00001 basic progress reporting.

00002 source and target file formats and parameter kinds.

00004 segment boundaries computed from label files.

00010 display memory usage after processing each file. Trace flags are set using the -T option or the TRACE configuration variable.

5.7.4 HRest

HRest performs basic Baum-Welch re-estimation of the parameters of a single HMM using a set of observation sequences. HRest can be used for normal isolated word training in which the observation sequences are realisations of the corresponding vocabulary word.

Alternatively, HRest can be used to generate seed HMMs for phoneme-based recognition. In this latter case, the observation sequences will consist of segments of continuously spoken training material. HRest will cut these out of the training data automatically by simply giving it a segment label.

The allowable options to HRest are as follows:

-C is used to specify a configuration file name and the option.

-S is used to specify a script file name.

-I mlf This loads the master label file mlf. This option may be repeated to load several MLFs.

-L dir Search directory dir for label files (default is to search current directory).

-M dir Store output HMM macro model files in the directory dir. If this option is not given, the new HMM definition will overwrite the existing one.

-H mmf Load HMM macro model file mmf. This option may be repeated to load multiple MMFs.

-v f This sets the minimum variance (i.e. diagonal element of the covariance matrix) to the real value f. This is ignored if an explicit variance floor macro is defined. The default value is 0.0.

-w f Any mixture weight or discrete observation probability which falls below the global constant MINMIX is treated as being zero.

-i N This sets the maximum number of re-estimation cycles to N (default value 20).

-m N Sets the minimum number of training examples to be N. If fewer than N examples are supplied then an error is reported (default value 3).

-T HRest supports the following trace options where each trace flag is given using an octal base:

000001	basic progress reporting.
000002	output information on the training data loaded.
000004	the observation probabilities.
000010	the alpha matrices.
000020	the beta matrices.
000040	the occupation counters.
000100	the transition counters.
000200	the mean counters.
000400	the variance counters.
001000	the mixture weight counters.
002000	the re-estimated transition matrix.
004000	the re-estimated mixture weights.
010000	the re-estimated means.
020000	the re-estimated variances.

Trace flags are set using the -T option or the TRACE configuration variable.

5.7.5 HERest

This program is used to perform a single re-estimation of the parameters of a set of HMMs, or linear transforms, using an *embedded training* version of the Baum-Welch algorithm. Training data consists of one or more utterances each of which has a transcription in the form of a standard label file (segment boundaries are ignored). For each training utterance, a composite model is effectively synthesised by concatenating the phoneme models given

by the transcription. Each phone model has the same set of accumulators allocated to it as are used in HRest but in HERest they are updated simultaneously by performing a standard Baum-Welch pass over each training utterance using the composite model.

HERest is intended to operate on HMMs with initial parameter values estimated by HInit/HRest. HERest supports multiple mixture Gaussians, discrete and tied-mixture HMMs, multiple data streams, parameter tying within and between models, and full or diagonal covariance matrices. HERest also supports tee-models , for handling optional silence and non-speech sounds. These may be placed between the units (typically words or phones) listed in the transcriptions but they cannot be used at the start or end of a transcription. Furthermore, chains of tee-models are not permitted.

HERest includes features to allow parallel operation where a network of processors is available. When the training set is large, it can be split into separate chunks that are processed in parallel on multiple machines/processors, consequently speeding up the training process.

The allowable options to HERest are as follows:

-C is used to specify a configuration file name and the option.

-S is used to specify a script file name.

-I mlf This loads the master label file mlf. This option may be repeated to load several MLFs.

-L dir Search directory dir for label files (default is to search current directory).

-H mmf Load HMM macro model file mmf. This option may be repeated to load multiple MMFs.

-M dir Store output HMM macro model files in the directory dir. If this option is not given, the new HMM definition will overwrite the existing one.

-v f This sets the minimum variance (i.e. diagonal element of the covariance matrix) to the real value f (default value 0.0).

-w f Any mixture weight which falls below the global constant MINMIX is treated as being zero.

-t f [i l] Set the pruning threshold to f. During the backward probability calculation, at each time t all (log) β values falling more than f below the maximum β value at that time are ignored. During the subsequent forward pass, (log) α values are only calculated if there are corresponding valid α values. Furthermore, if the ratio of the αβ product divided by the total probability (as computed on the backward pass) falls below a fixed threshold then those values of α and β are ignored. Setting f to zero disables pruning (default value 0.0). Tight pruning thresholds can result in HERest failing to process an utterance. if the i and l options are given, then a pruning error results in the threshold being increased by i and utterance processing restarts. If errors continue, this procedure will be repeated until the limit l is reached.

-T HERest supports the following trace options where each trace flag is given using an octal base:

00001 basic progress reporting.
00002 show the logical/physical HMM map.
00004 list the updated model parameters. of tied mixture components.
Trace flags are set using the -T option or the TRACE configuration variable.

5.7.6 HHEd

HHEd is a script driven editor for manipulating sets of HMM definitions. Its basic operation is to load in a set of HMMs, apply a sequence of edit operations and then output the transformed set. HHEd is mainly used for applying tyings across selected HMM parameters. It also has facilities for cloning HMMs, clustering states and editing HMM structures.

Many HHEd commands operate on sets of similar items selected from the set of currently loaded HMMs. For example, it is possible to define a set of all final states of all vowel models, or all mean vectors of all mixture components within the model X, etc. Sets such as these are defined by item lists using the syntax rules given below. In all commands, all of the items in the set defined by an item list must be of the same type where the possible types are:
s - state

t - transition matrix

p - pdf

w - stream weights

m - mixture component

d - duration parameters

u - mean vector

x - transform matrix

v - variance vector

i - inverse covariance matrix

h - HMM definition

The allowable option to HHEd for our experiment is:

-H mmf Load HMM macro model file mmf. This option may be repeated to load multiple MMFs.

5.7.7 HParse

The HParse program generates word level lattice files (for use with e.g. HVite) from a text file syntax description containing a set of rewrite rules based on extended Backus-Naur Form (EBNF). The EBNF rules are used to generate an internal representation of the corresponding finite-state network where HParse network nodes represent the words in the network, and are connected via sets of links. This HParse network is then converted to HTK V2 word level lattice. The program provides one convenient way of defining such word level lattices.

The lattice produced by HParse will often contain a number of !NULL nodes in order to reduce the number of arcs in the lattice. The use of such !NULL nodes can both reduce size and increase efficiency when used by recognition programs such as HVite.

5.7.8 HVite

HVite is a general-purpose Viterbi word recogniser. It will match a speech file against a network of HMMs and output a transcription for each. When performing N-best recognition a word level lattice containing multiple hypotheses can also be produced.

Either a word level lattice or a label file is read in and then expanded using the supplied dictionary to create a model based network. This allows arbitrary finite state word networks and simple forced alignment to be specified.

The allowable options to HVite are as follows:

-C is used to specify a configuration file name and the option.

-S is used to specify a script file name.

-t f [i l] Enable beam searching such that any model whose maximum log probability token falls more than f below the maximum for all models is deactivated. Setting f to 0.0 disables the beam search mechanism (default value 0.0). In alignment mode two extra parameters i and l can be specified. If the alignment fails at the initial pruning threshold f, then the threshold will by increased by i and the alignment will be retried. This procedure is repeated until the alignment succeeds or the threshold limit l is reached.

-s f Set the grammar scale factor to f. This factor post-multiplies the language model likelihoods from the word lattices. (default value 1.0).

-i s Output transcriptions to MLF s.

-w s Perform recognition from word level networks. If s is included then use it to define the network used for every file.

-H mmf Load HMM macro model file mmf. This option may be repeated to load multiple MMFs.

-T HVite supports the following trace options where each trace flag is given using an octal base:

0001 enable basic progress reporting.
0002 list observations.
0004 frame-by-frame best token.
0010 show memory usage at start and finish.
0020 show memory usage after each utterance.

Trace flags are set using the -T option or the TRACE configuration variable.

5.7.9 HResults

HResults is the HTK performance analysis tool. It reads in a set of label files (typically output from a recognition tool such as HVite) and compares them with the corresponding reference transcription files. For the analysis of speech recognition output, the comparison is based on a Dynamic Programming-based string alignment procedure. For the analysis of word-spotting output, the comparison uses the standard US NIST FOM metric. When used to calculate the sentence accuracy using DP the basic output is recognition statistics for the whole file set in the format

```
------------------------ Overall Results --------------------
SENT:  %Correct=13.00 [H=13, S=87, N=100]
WORD:  %Corr=53.36, Acc=44.90 [H=460,D=49,S=353,I=73,N=862]
================================================================
```

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 accuracy based on the DP matches between the label files and the transcriptions. In this second line, H is the number of correct labels, 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 labels in the defining transcription files. The percentage number of labels correctly recognised is given by

$$\%\ Correct = \frac{H}{N} * 100\%$$

and the accuracy is computed by

$$Accuracy = \frac{H - I}{N} * 100\%$$

The allowable options to HResult are as follows:

-I mlf This loads the master label file mlf. This option may be repeated to load several MLFs.

-L dir Search directory dir for label files (default is to search current directory).

These are the speech commands that are required for our experiment of phoneme recognition.

5.8 Steps of the Experiment(Data preparation)

1. Make a folder Experiment (any name can be given) into the drive H (any drive can be used but this drive name is used in the command).

2. We have to put Train.scp (we have to make it according to our input voice and we can use any name instead of Train but extension should be .scp), config.txt, HCopy.exe files are into the folder Experiment.

3. In Train.scp (in this experiment we used 3000 input speech which contains in 30 different folders and each folder has got 100 inputs voice) we have to type the informations. In this file left part is the input and right part is the output.Train.scp looks like following :

"H:\\Experiment\\Recorded_Audio_0.wav""H:\\Experiment\\Recorded_Audio_0.mfcc "
"H:\\Experiment\\Recorded_Audio_1.wav""H:\\Experiment\\Recorded_Audio_1.mfcc "
"H:\\Experiment\\Recorded_Audio_2.wav""H:\\Experiment\\Recorded_Audio_2.mfcc "
"H:\\Experiment\\Recorded_Audio_3.wav""H:\\Experiment\\Recorded_Audio_3.mfcc "
"H:\\Experiment\\Recorded_Audio_4.wav""H:\\Experiment\\Recorded_Audio_4.mfcc "
"H:\\Experiment\\Recorded_Audio_5.wav""H:\\Experiment\\Recorded_Audio_5.mfcc "

4. The first stage of data preparation is to parameterise the raw speech waveforms into sequences of feature vectors. HTK support both FFT-based and LPC-based analysis. Here Mel Frequency Cepstral Coefficients (MFCCs), which are derived from FFT-based log spectra, will be used. Coding can be performed using the tool HCopy configured to automatically convert its input into MFCC vectors. To do this, a configuration file (config) is needed which specifies all of the conversion parameters. Reasonable settings for these are as follows :

# Coding parameters
SOURCEFORMAT = WAV
# SOURCERATE = 625
TARGETKIND = MFCC_D_A_0
TARGETRATE = 100000.0
SAVECOMPRESSED = T

SAVEWITHCRC = T

WINDOWSIZE = 250000.0

USEHAMMING = T

PREEMCOEF = 0.97

NUMCHANS = 26

CEPLIFTER = 22

NUMCEPS = 12

ENORMALISE = F

Some of these settings are in fact the default setting, but they are given explicitly here for completeness. In brief, they specify that the target parameters are to be MFCC using C0 as the energy component, the frame period is 10msec (HTK uses units of 100ns), the output should be saved in compressed format, and a crc checksum should be added. The FFT should use a Hamming window and the signal should have first order preemphasis applied using a coefficient of 0.97. The filterbank should have 26 channels and 12 MFCC coefficients should be output. The variable ENORMALISE is by default true and performs energy normalisation on recorded audio files. It cannot be used with live audio and since the target system is for live audio, this variable should be set to false.To run HCopy, a list of each source file and its corresponding output file is needed.

Files containing lists of files are referred to as script files and by convention are given the extension scp (although HTK does not demand this). Script files are specified using the standard -S option and their contents are read simply as extensions to the command line. Thus, they avoid the need for command lines with several thousand arguments.Assuming that the above script is stored in the file Train.scp, the training data would be coded by executing

HCopy -T 1 -C config.txt -S Train.scp

This is illustrated in figure 5.1 :

5. Now we have to make source.mfc (we can use any name instead of source but extension should be .mfc) into the folder practice and type the informations.The informations are taken from the right part of the step 3 but here “”will not be used. source.mfc looks like following :

H:\\Experiment\\Recorded_Audio_0.mfcc

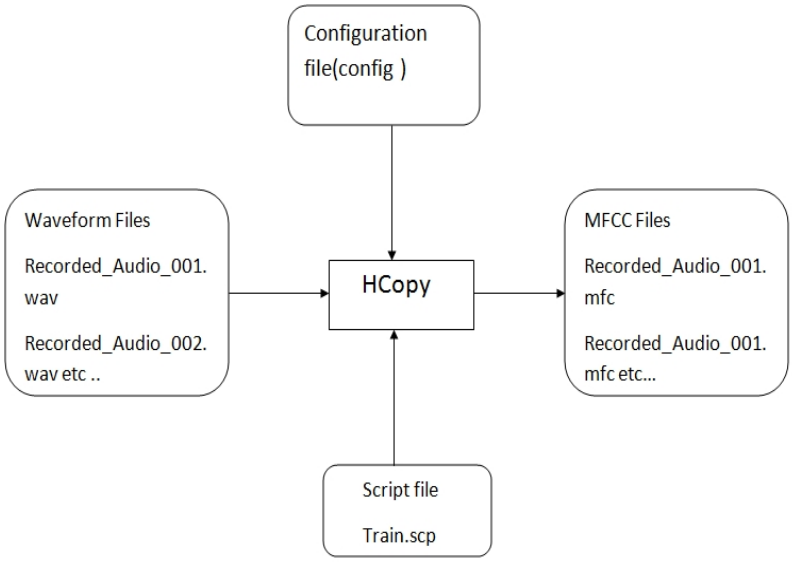


Figure 5.1: HCopy Command

H:\\Experiment\\Recorded_Audio_1.mfcc

H:\\Experiment\\Recorded_Audio_2.mfcc

H:\\Experiment\\Recorded_Audio_3.mfcc

H:\\Experiment\\Recorded_Audio_4.mfcc

H:\\Experiment\\Recorded_Audio_5.mfcc .....

6. Now we Put HList.exe into the folder Experiment.

7. Then we Put Batch_HList.m into the folder Experiment and open this file using matlab.

8. After executing Batch_HList.m, the following file will be generated

- dest.mfc into the folder Experiment.
- Output and genoutput folder will be created into every folder of the input voice and in the output folder we will get automatically the following files:

  Recorded_Audio_0.shk, Recorded_Audio_1.shk, ...
- But genoutput folder will not contain any information at that time.

9. Now we have to Put Vector_Formatter.cpp file into the folder Line-Cutter (we can use any name) which we have to create into the folder Experiment.

10. Put dest.mfc (which we created in the last step#8) into the folder Line-Cutter.

11. Run Vector_Formatter.cpp using the visual studio 6.0. after executing this program the following information will be generated :

- Out_vec.test into the folder Line-Cutter. We open the file Out_vec.test and remove last blank line and save it.
- In genoutput folder we will get automatically the following files:

  Recorded_Audio_0.gen, Recorded_Audio_1.gen, ...

12. Make a folder Converted (we can use any name) into the folder Experiment. Put convert.c into the folder Converted. Also put Out_vec.test which was created in the last step into the folder Line-Cutter into the folder Converted.

13. Make another folder Train (we can use any name) into the folder Converted. Now open and run convert.c using visual studio software. After executing this program the following files will be generated:

- Out_vec.scp will be generating into the folder Converted.
- file1.vec, file2.vec, file3.vec ..... files will be created into the folder Train.These vector files contain garbage data.

5.9 Steps of the Experiment(Training)

1. The first step in HMM training is to define a prototype model. The parameters of this model are not important, its purpose is to define the model topology. For phone-based systems, a good topology to use is 3-state left-right with no skips such as the following
∼o <VecSize >39 <MFCC_0_D_A >
∼h “proto ”
<BeginHMM>
<NumStates>5
<State>2

<Mean>39

0.0 0.0 0.0 ...

<Variance>39

1.0 1.0 1.0 ...

<State>3

<Mean>39

0.0 0.0 0.0 ...

<Variance>39

1.0 1.0 1.0 ...

<State>4

<Mean>39

0.0 0.0 0.0 ...

<Variance>39

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>

where each ellipsed vector is of length 39. This number, 39, is computed from the length of the parameterised static vector (MFCC 0 = 13) plus the delta coefficients (+13) plus the acceleration coefficients (+13). The HTK tool HCompV will scan a set of data files, compute the global mean and variance and set all of the Gaussians in a given HMM to have the same mean and variance. Hence, assuming that a list of all the training files is stored in train.scp, the command :

HCompV -T 1 -C HTK_39dpf.config -S Train.scp -m -v 0.01 -M Model proto_5states_39dim

will create a new version of proto_5states_39dim in the directory Model(the directory created by us in the Experiment directory)in which the zero means and unit variances above have been replaced by the global speech means and variances. Note that the prototype

HMM defines the parameter kind as MFCC_0_D_A. This means that delta and acceleration coefficients are to be computed and appended to the static MFCC coefficients computed and stored during the coding process described above. To ensure that these are computed during loading, the configuration file config should be modified to change the target kind, i.e. the configuration file entry for TARGETKIND should be changed to

TARGETKIND = MFCC_0_D_A

Under Model directory two files are created :

- vfloors
- proto_5states_39dim

2.Now we rename vloors to macro .And we copy first 3 lines from proto_5states_39dim to macro .

3.Now we run the following command:
HRest -T 1 -C HTK_39dpf.config -S Train.scp -I Train.MLF -L Label -M Model1 -H macro -w 3 -v 0.05 -i 40 -m 1 proto_5states_39im

Here Label is the folder where label files of the input data are located.The extension of the label files is .lab . Model1 is the folder created by us in the Experiment folder . Under Model1 directory two files are created :

- macro
- proto_5states_39dim

4.Now copy & paste proto_5states_39dim and rename it as hmmdefs.

5. Given this new prototype model stored in the directory Model, a Master Macro File (MMF) called hmmdefs containing a copy for each of the required monophone HMMs is constructed by manually copying the prototype and relabeling it for each required monophone (including "sil"). The format of an MMF is similar to that of an MLF and it serves a similar purpose in that it avoids having a large number of individual HMM definition files.Now we copy monophones and paste it into the folder Experiment. This file contains 53 elements.

6.Now open hmmdefs and copy the codes from ∼h “proto_5states_39dim ”(4th line) to <ENDHMM >(last line) and paste it 52 times. So total will be 53.Now remove ∼h “proto_5states_39dim”(53 times) and write aa, ch, ... (total 53 word from monophones file).

Now we run HERest commnd for several mixtures. For every mixtures we run the command 10 times for 10 Model directory.Each time HERest is run it performs a single re-estimation. Each new HMM set is stored in a new directory.The process leading to the initial set of monophones in the directory Experiment is illustrated in Figure 5.2 .

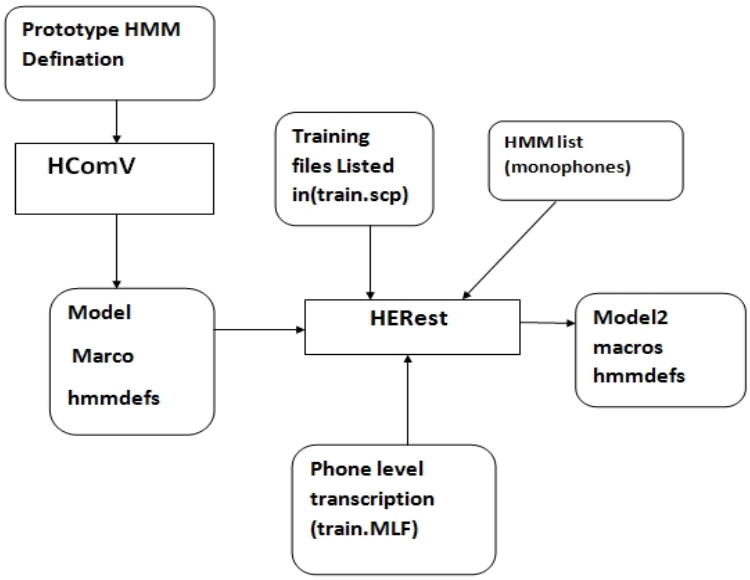


Figure 5.2: HERest Command

5.9.1 Mixture_1

1.Now we run the following command after creating Model2 directory:
HERest -T 1 -C HTK_39dpf.config -S Train.scp -I Train.MLF -L Label -M Model1 -H macro -w 3 -v 0.05 -t 250.0 150.0 1000.0 -H Model1\hmmdefs -H Model1\macro -M Model2 monophones .

Under Model2 directory two files are created :

- macro
- hmmdefs

2.Now we run the following command after creating Model3 directory:
HERest -T 1 -C HTK_39dpf.config -S Train.scp -I Train.MLF -L Label -M Model2 -H macro -w 3 -v 0.05 -t 250.0 150.0 1000.0 -H Model2\hmmdefs -H Model2\macro -M Model3 monophones .

Under Model2 directory two files are created :

- macro
- hmmdefs

3. Now we have to repeat the above step until Model10 directory is created .

5.9.2 Mixture_2

1.Now we run the following command after creating Mix2 directory and Model1 directory under Mix2 directory:
HHEd -H Model10\macro -H Model10\hmmdefs -M Mix2\Model1 Hed\mix2.hed monophones .

Under Model1 directory two files are created :

- macro
- hmmdefs

2. Now we have to repeat the above step until Model10 directory is created .

5.9.3 Mixture_4

1.Now we run the following command after creating Mix4 directory and Model1 directory under Mix4 directory:
HHEd -H Mix2 \Model10\macro -H Mix2\Model10\hmmdefs -M Mix4\Model1 Hed\mix4.hed monophones.

Under Model1 directory two files are created :

- macro
- hmmdefs

2. Now we have to repeat the above step until Model10 directory is created .

5.9.4 Mixture_8

1.Now we run the following command after creating Mix8 directory and Model1 directory under Mix8 directory:
HHEd -H Mix4 \Model10\macro -H Mix4\Model10\hmmdefs -M Mix8\Model1 Hed\mix8.hed monophones.

Under Model1 directory two files are created :

- macro
- hmmdefs

2. Now we have to repeat the above step until Model10 directory is created .

5.9.5 Mixture_16

1.Now we run the following command after creating Mix16 directory and Model1 directory under Mix16 directory:
HHEd -H Mix8 \Model10\macro -H Mix8\Model10\hmmdefs -M Mix16\Model1 Hed\mix16.hed monophones.

Under Model1 directory two files are created :

- macro
- hmmdefs

2. Now we have to repeat the above step until Model10 directory is created .

5.9.6 Mixture_32

1.Now we run the following command after creating Mix32 directory and Model1 directory under Mix32 directory:
HHEd -H Mix16 \Model10\macro -H Mix16\Model10\hmmdefs -M Mix32\Model1 Hed\mix32.hed monophones.

Under Model1 directory two files are created :

- macro
- hmmdefs

2. Now we have to repeat the above step until Model10 directory is created .

5.10 Steps of the Experiment(Testing)

1. Now we run the following speech command :
HParse Grammer.grm lattice.lat
here lattice.lat is the output file.

2. HVite -T 1 -C HTK_39dpf.config -S Train.scp -t 300.0 -s 5.0 -i Result\out1.txt -w lattice.lat -H Model10\hmmdefs dictionary.dic monophones

HVite -T 1 -C HTK_39dpf.config -S Train.scp -t 300.0 -s 5.0 -i Result\out2.txt -w lattice.lat -H Model10\hmmdefs dictionary.dic monophones

HVite -T 1 -C HTK_39dpf.config -S Train.scp -t 300.0 -s 5.0 -i Result\out4.txt -w lattice.lat -H Model10\hmmdefs dictionary.dic monophones

HVite -T 1 -C HTK_39dpf.config -S Train.scp -t 300.0 -s 5.0 -i Result\out8.txt -w lattice.lat -H Model10\hmmdefs dictionary.dic monophones

HVite -T 1 -C HTK_39dpf.config -S Train.scp -t 300.0 -s 5.0 -i Result\out16.txt -w lattice.lat -H Model10\hmmdefs dictionary.dic monophones

HVite -T 1 -C HTK_39dpf.config -S Train.scp -t 300.0 -s 5.0 -i Result\out32.txt -w lattice.lat -H Model10\hmmdefs dictionary.dic monophones

3. By running the following commands we check the correct rate and accuracy rate .

HResults -I Train.MLF -L Label monophones Result\out1.txt >>out1.log

HResults -I Train.MLF -L Label monophones Result\out2.txt >>out2.log

HResults -I Train.MLF -L Label monophones Result\out4.txt >>out4.log

HResults -I Train.MLF -L Label monophones Result\out8.txt >>out8.log

HResults -I Train.MLF -L Label monophones Result\out16.txt >>out16.log

HResults -I Train.MLF -L Label monophones Result\out32.txt >>out32.log

In out.log file the result of the experiment is shown – The correct rate and the accuracy rate .

5.11 Conclusion

The HTK tools are best described by going through the processing steps involved in building a sub-word based continuous speech recogniser. As shown in Figure 5.3, there are 4 main phases: data preparation, training, testing and analysis.

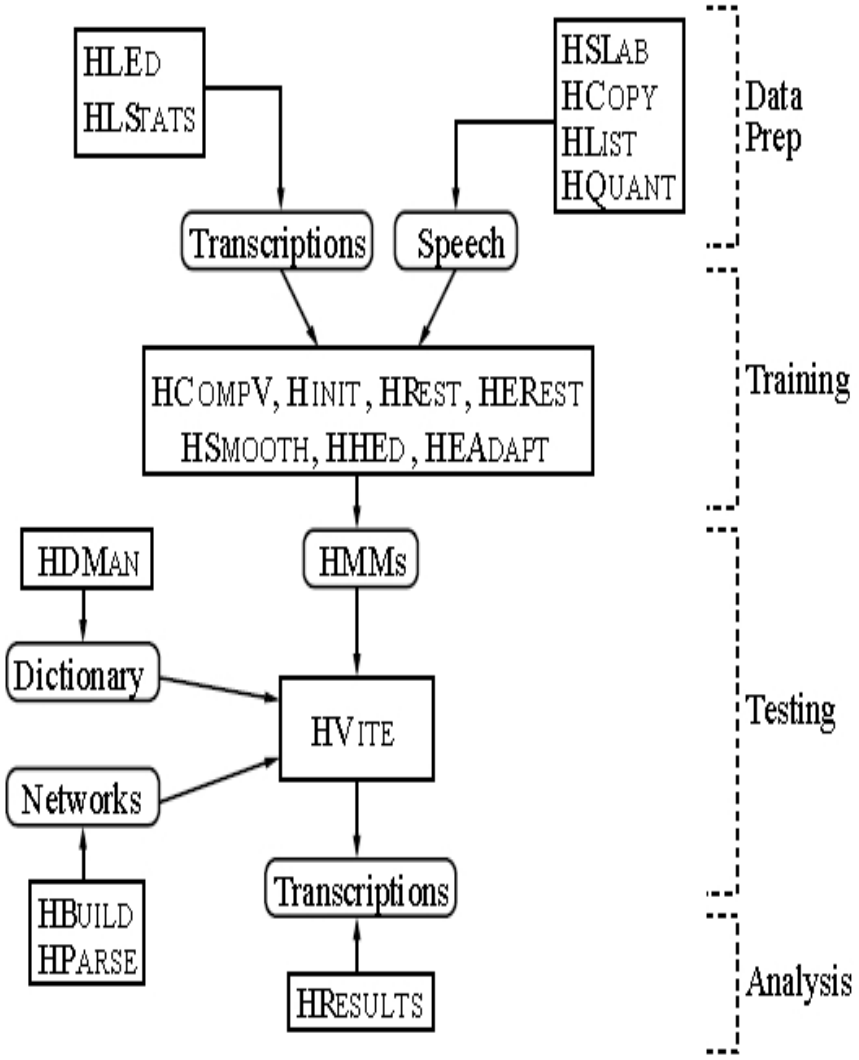


Figure 5.3: HTK Processing Stages

CHAPTER 6
CONCLUSIONS AND FUTURE WORKS

6.1 Conclusion

In this project, the main task was to develop an automatic phoneme recognizer for Bangla language. This system is aimed at improving on the current Human-computer interface by introducing a voice interface, which has proved to have so many advantages to the traditional I/O methods. Users naturally know how to speak so this would provide an easy interface which does not necessarily require special training which is normally the case when we are using various ICT tools for the first time.

This paper showed a preparation of Bangla speech corpus and provided some experiments to obtain phoneme recognition performance. This short introduction covered the basics of speech recognition up to the principles of isolated word recognition.

The following conclusions are drawn from the experiments:

i) The MFCC-based system provides tremendous improvement of Bangla phoneme recognition accuracy for both training and test data.

ii) A higher Bangla phoneme correct rate for training and test data is also obtained by the MFCC-based system.

iii) We have learned how the basic algorithm for isolated word recognition with HMMs works.

iv) If we have large vocabularies, we will not have enough training samples for the creation of word models.

v) We have learned how “n-best” word sequences instead of the best word sequence only can be searched .

vi) We have learned how we can integrate additional knowledge sources like finite state grammars or stochastic language models into the recognition process.

vii) We have learned how we can reduce the computational complexity of the search algorithm so that we are able to meet real time constraints.

6.2 Future Works

Some issues are beyond of this thesis. In future, we would like to do the following works :

(a) In the proposed canonicalization process, DPF extractor is implemented by a single MLN, which cannot solve coarticulation problems and hence, shows some misclassifications. Some experiments should be done using the proposed DPF extractor, MLN+MLN+In/En+GS in the canonicalization process.

(b) Some authors incorporated multiple noise reduction techniques, instead of single one, in their system and obtained better word accuracy for high and mid level SNR in multicondition training. On the other hand, the proposed method in this thesis used two-stage Wiener filtering technique and showed higher word accuracy in low SNR conditions. Therefore, some experiments by embedding multiple noise reduction techniques in the proposed method are needed to get better word accuracy for all SNR conditions in multicondition training.

Some word recognition experiments by incorporating language models will be evaluated in near future by incorporating the proposed DPF extractor.We would like to do further experiments for obtaining phoneme recognition performance after inserting both features into the neural network based systems.

REFERENCES

[1] S. Pinker, "The language instinct," *HarporCollins*, 2010.

[2] D. Neeraj, A. Ganapathiraju, and J. Picone, "Hierarchical search for large vocabulary conversational speech recognition," *IEEE Signal Processing Magazine*, 1999.

[3] S. Kandasamy, "Speech recognition systems," *SURPRISE Journal*, 1995.

[4] V. Zue and C. Ronald, "Speech recognition.survey of the state of the art in human language technology," 1996.

[5] Z. Mengjie, "Overview of speech recognition and related machine learning techniques," *Technical Report*, December 2004.

[6] A. Dix, J. Finlay, G. Abowd, and R. Beale, *Human-Computer Interaction*. Prentice Hal, Englewood Cliffs and NJ and USA, 1998.

[7] R. John, W. Y. Wong, C. Chung, and D. K. Kim, "Automatic speech recognition for generalised time based media retrieval and indexing," *sixth ACM International Conference on Multimedia*, 1998.

[8] Jelinek, "The development of an experimental discrete dictation recognizer," *IEEE,73*, 1985.

[9] R. Cole, J. Mariani, H. Uszkoreit, A. Zaenen, and V. Zue, *Survey of the State of the Art in Human Language Technology*. Cambridge University Press, 1996.

[10] http://en.wikipedia.org/wiki/International_Phonetic_Alphabet, [Last visited 21/12/2013].

[11] http://en.wikipedia.org/wiki/Bengali_alphabet, [Last visited 21/12/2013].

[12] http://banglahelp.blogspot.com/2007/05/ipa-chart-in-bengali.html, [Last Visited 3/12/2013].

[13] C. Masica, *The Indo-Aryan Languages*. Cambridge University Press, 1991.

[14] http://en.wikipedia.org/wiki/List_of_languages_by_total_speakers, [Last visited 21/12/2013].

[15] A. Black, R. Kumar, and R. Sangal, "Experiments with unit selection speech databases for indian languages," *National seminar on Language Technology Tools*, 2003.

[16] http://en.wikipedia.org/wiki/Bengali_phonology, [Last visited 21/12/2013].

[17] http://www.prothom-alo.com/, [Last visited 21/12/2013].

[18] S. Young and G. Evermann, *The HTK Book*, 2006.

APPENDIX A
BATCH_HLIST.M

Here is the code of an MatLab file which converts the mfcc file to shk files in the output folder:

```
%%Input Script File
fin = fopen('H:\\Experiment\\source.mfc');
%%Output Script File
fout= fopen('H:\\Experiment\\dest.mfc ','w ');
%%HTK Command
htk='H:\\Experiment\\HList    -h ';
%%HList Command Generated File Extension
extn = '.shk ';

tline = fgetl(fin);

while  ischar(tline)
disp(tline)
toutline = tline;

pathstr,name,ext,versn
= fileparts(tline); % Get File Path, Name, Extension and Version
disp(pathstr);
genpath=sprintf('%s',pathstr);
pathstr=strcat(pathstr,'\output '); % Insert Another Level of Folder Named output
```

```
genpath=strcat( genpath,‘\genoutput ’);
%disp(pathstr);
%disp(genpath);
a = exist (pathstr, ‘dir’);%Check If The Directory Exists
disp (a);

if(a =7)
status =mkdir(pathstr) %If Not Exist Create Directory
disp(status);
end

a=exist(genpath,‘dir ’); %Check If The Directory Exists

if(a~=7)
mkdir(genpath) %If Not Exist Create Directory
end

str=strcat(pathstr,‘\ ’,name,extn); %Concat New File Extension
a=exist( str,‘file’);

if(a==2)
delete(str) %If HList Generated File Already Exist Then Remove Old File
end

fprintf(fout, ‘%s ’,str); %Write The File Name(Path+File) Into Output Script File
fprintf(fout,‘\r\n ’);

cmd=sprintf( ‘%s %s≫%s’,htk,tline,str); %Build Command String

%HList
status = dos(cmd); % Run Command
tline = fgetl(fin);
```

```
end

fclose(fin); %Close Opened File Stream
fclose(fout);
```