Implementation of Statistical Speech to Text Recognition System for Punjabi Language

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Software Engineering

Submitted By

Shama Mittal

(Roll No. 801431026)

Under the supervision of

Ms. Rupinderdeep Kaur

Lecturer, CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2016

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Implementation of Statistical Speech to Text Recognition System for Punjabi Language*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Ms. Rupinderdeep Kaur* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Shama Mittal)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Rupinderdeep Kaur)

Lecturer, CSED

Countersigned by

(Dr. Maninder Singh)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. S. Bhatia)
Dean (Academic Affairs)
Thapar University
Patiala

ACKNOWLEDGEMENTS

The completion of thesis work mainly concerned with many people, who contributed directly or indirectly through their constructive criticism in the evolution and preparation of this work. It would not be fair on my part, if I don't say a word of thanks to all those whose sincere advice made this period a real educative, enlightening, pleasurable and memorable one.

First of all, a special debt of gratitude is owed to my supervisor **Ms. Rupinderdeep Kaur**, for her gracious efforts and keen pursuits, which has remained as a valuable asset for the successful completion of research work. Her dynamism and diligent enthusiasm has been highly instrumental in keeping my spirit high. The flawless and forthright suggestions blended with an innate intelligent application have crowned my task a success.

I am equally grateful to **Dr. Maninder Singh** Head Department of Computer Science and Engineering and **Dr. Rupali Bhardwaj** P.G. Coordinator Department of Computer Science and Engineering, for their support and motivation that encouraged me for the dissertation work.

I also like to offer my sincere thanks to all faculty members, teaching and non-teaching staff of Department of Computer Science and Engineering (CSED) and staff of central library, Thapar University, Patiala for their assistance.

I would also like to thank to my parents and friends for their constant encouragement during the entire course of my work.

Above all, I owe my reverence to Almighty for the kindness who blessed me at finish of whole work.

(Shama Mittal)

(801431026)

ABSTRACT

In order to take advantage of a machine's facilities, it is very important for human beings to interact with them adequately. Thus, this interaction of human beings with the machines fascinating most of the researcher's intentness towards it these days. The process of recognizing and translating the spoken words, using the technologies which uses computerized devices, such as smart technologies and robotics, is known as Automatic Speech Recognition (ASR). Here, the implementation of the statistical speech to text recognition system is explained and language chosen for this implementation is Punjabi because it is a highly prosodic language and work done on any prosodic language is very less. In development of this speech recognition system, MFCC is used as a feature extraction technique and these features are classified with Hidden Markov Model (HMM). HMM has been implemented using HTK Toolkit. First step for this work is data collection. Here a total of 7 hours data is collected in read speech mode, lecture speech mode and conversational speech mode. After the completion of data collection, the whole data is transcribed using International Phonetic Alphabet (IPA) chart. As a result of transcription, prosodic database consists a vocabulary of 266 unique words including 39 mono-syllable words, 43 bi-syllable words, 110 tri-syllable words and 74 multi- syllable words. Second step is data preparation, in which hmmlist, grammar and dictionary files are created using this vocabulary. Once the data is prepared, 75% of it is used for training with the help of HTK Toolkit and remaining 25% data is used for testing. The experimental results depicts that the accuracy of the recognition system increases from 61.84% to 69.95% in read speech mode, 41.45% to 53.32% in lecture speech mode and 20.24% to 21.00% in conversational speech mode. The accuracy of the system increases with the increase in number of mixtures from 29 to 33 (excluding silence) as well as increase in data from 3 hours to 7 hours. When a part of this data is trained and tested for word level speech recognition system, the system shows an accuracy of 57.54 % using triphone models.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Expanded Form
ASR	Automatic Speech Recognition
HMM	Hidden Markov Model
IPA	International Phonetic Alphabet
C	Constants
V	Vowels
HTK	Hidden markov Tool Kit
DAG	Directed Acyclic Graph
ROC	Receiver Operating Curve
FOM	Figure Of Merit
EBNF	Extended Backus-Naur Form
LPC	Linear Predictive Coding
HSMM	Hidden Semi Markov Model
MFCC	Mel Frequency Cepstral Coefficient
CDM	Chinese Dictation Machine
LVCSR	Large Vocabulary Continuous Speech Recognition
MMIE	Maximum Mutual Information Estimation
LFPC	Log Frequency Power Coefficients
LPCC	Linear Prediction Cepstral Coefficient
CMU	Carnegie Mellon University
ANN	Artificial Neural Network
DTW	Dynamic Time Wrap
ASCII	American Standard Code for Information Interchange

CHAPTER 1

Introduction

The ability of human beings to create communication between machines and computers with the help of keyboards or some other external devices is more cumbersome and slow too. A speech can be considered as a significant component to make this communication intelligible and rapid [1]. The process of recognizing and translating the spoken words, using the technologies which uses computerized devices, such as smart technologies and robotics, is known as Automatic Speech Recognition (ASR).

All of us are living in a digital world where digital gadgets like desktops, laptops, tablets, phones, music players, digital-watches, *etc*. are playing a vital role in our lives. In current scenario, our life seems incomplete without the presence of these gadgets. All of these gadgets are the result of technological advancements and intelligence of human brain. In the early years of digital era, interaction with computers is done using hardware components along with their compatible software program. All these interactions made tasks like mathematical calculations and computations easy.

Nowadays, all of these tasks are not possible without being comfortable and proficient in digital technology. As our society is getting increasingly dependent on these digital devices, there is an imminent requirement of making these devices hand-free and voice-controlled. Making the digital devices speech-controlled will improve the output of our work a great deal. The necessity of such type of speech-controlled devices and software programs, which are able to direct the digital machines, has increased in recent years. It has happened due to the requirement to make the current tasks easier for every person of the society and due to this fact the research in this domain is at its peak and is touching new heights every day.

A real life example that we can be considered is of a disabled person, who cannot type on a manual keyboard. It would be very difficult for a blind person to use computer. In order to empower such persons to use technology independently and live a dignified and progressive life, it is necessary to have an application which will make controlling computers and digitally assisted devices, easy in their own native language.

Here, language plays an important role as not every person is fluent or proficient enough in English, German, French or any other standard language to use the computers. Many studies have proved that

India and other Asian countries contribute the highest number of digital technology users and most of them prefer to work in own their native language [2]. Another reason of using their native language in work is to save and propagate their heritage and language for their next generations to come. In this digital era, if regional languages and heritage is left behind because of technological trends, it will not take long for these languages to be classified as 'endangered'. This requirement enforces the researchers to explore the domain of speech recognition for Punjabi and other such languages.

In an Automatic Speech Recognition system, an articulation of signals of speech (controlling with the help of a telephone or microphone *etc*.) is taken as an input and is converted into as much as promising to the text sequence of the data spoken by the user. Main problem in development of ASR system is because of the variety of the speaking style of different human beings and along with this environmental disturbances are the main reason too [14]. So the main objective of ASR system is to revolutionize the signals based on speech into the signals based on text which are not dependent on device, surroundings or the speaker in an efficient and more accurate way as possible.

Both Speech recognition and speech synthesis require phonetic transcription. In speech synthesis, firstly, in preprocessing stage, text is assigned the phonetic transcription and then front end divides and marks the text into prosodic units. Symbolic linguistic representation consists two elements, Phonetic transcription and prosody information. Then, synthesizer converts symbolic linguistic representation into sound. In speech recognition, speech is provided as an input to system and then corresponding phonetic transcription is generated by the system as output.

There are a variety of approaches to automatic speech recognition, namely:

- Acoustic phonetic
- Statistical pattern recognition
- Artificial intelligence approaches

Out of these approaches, the statistical approach is the fundamental method employed in HMM.

1.1 Terminologies

The brief introduction of the terms either related to arts, science, or any particular subject, is known as terminology. The basic terminologies that are relevant to this speech recognition system and used in this work are briefly explained below.

1.1.1 Phoneme

Every language has a smallest and most fundamental unit of sound, named as phoneme, combined with other phonemes, they make meaningful units such as words. There is a distinction between phone

and phoneme. A phone is basically a representation of a sound unit only and corresponding to sound units, there are infinite number of phones are present. It is not necessary that combining different phones would produce some meaningful unit. It can simply be a noise, word, animal cry, *etc*. However, phoneme always produces a meaningful unit. For example, words 'madder' and 'matter', both composed of different phonemes but in American English, both words sounds same when pronounced, *i.e.,* both words have same phones but different phonemes. If a particular person makes different sounds, phones may be same in all languages but are written differently in different languages and for that different phonemes are used.

1.1.2 Acoustic Model

An acoustic model embraces the demography illustration Hidden Markov Models (HMMs) of different sounds which, when combined, makes a meaningful unit such as word [6]. Phonemes are assigned to each statistical representation of sound. For creating an acoustic model, take the database containing speech in it and the corresponding transcriptions which are given as input to some speech recognition software, and that in return provides the statistical representation of different sounds.

1.1.3 Language Model

This model involves the grammar and dictionary used by the speech recognition software which helps in recognizing the phoneme of unknown utterances. After several experiments, it has been observed that the language model depends on the recognizer being used [3]. If text is to be produced, then the language model may reasonably be constructed by processing examples of corresponding written materials.

1.1.4 Speech Recognition

Speech can be considered as a significant component to accomplish the interaction between the human beings and machine. In Speech Recognition system, an articulation of signals of speech is taken as an input and is converted into as much as promising to the text sequence of the data spoken by the user.

1.1.5 Word Level Speech Recognition

Word level speech recognition system, is a system in which, the text is taken in the form of a word. From the speech, the data is taken word by word and recognized with the help of word level recognition system. When a speaker speaks out a word that need to be recognized, recognizer will listen the distinct sounds and looks for matching HMMs of each sound and then, determine the sequence of phones that make up a particular word based on training given to it and then these

sequence of phonemes, *i.e.*, the pronunciation, are checked in dictionary. If entry exist then, the word mentioned against it, is picked up. Word level recognition is much better than phone level, because a full word is recognized here instead of single phone of a word.

1.1.6 Tri-phones

The unique phones are tied up into a combination of three phones together to make it a tri-phone. As, we got a combination of three phones together so it will help in making a unique word, that is further used in word recognition. These are represented in the combination as L-X+R. To create a single tri-phone from different phones, the 'L' (Left-Hand) phone is preceded by 'X' (Center) phone which is then followed by the 'R' (Right-Hand) phone. For example, if we want to recognize kamedian using tri-phones than it can be written as follows.

kamedian [kamedian] k ae m ey d ee n

kamedian [kamedian] k+ae k-ae+m ae-m+ey m-ey+d ey-d+ee d-ee+n ee-n

The first line shows the recognition on phone level using phones and second line shows the recognition on word level using tri-phone.

1.1.7 Punjabi Language

Punjabi language is a very popular language and this can be proved by viewing its position in world's widely spoken language list where it is placed at 10^{th} position [4]. It is used in Punjab in India, as well as, Punjab in Pakistan. Punjabi language is syllabic in nature. Punjabi language consists of 41 consonants, 9 vowels and 3 symbols are used as supporting vowels. Punjabi language is chosen for this work, because, very less work is done on this Prosodic language. Most of the work done in this field is on English, Arabic, French *etc*. languages.

1.2 Hidden Markov Models (HMM)

HMM is a Markov method which is divided into two components:

- Observable states
- Unobservable states (hidden states).

It is a doubly stochastic method because underlying stochastic method is hidden (not noticeable) but through some another set of stochastic method (that generated the consecutive flow of observed symbols) can be observed. For example, suppose in a room there is a barrier (say, curtains) between two persons and one person cannot see what is happening on the other side. Firstly, person on one side of curtain is flipping a single coin (or multiple coins) and telling the results to second person on the other side of curtain but not exactly telling what he is doing. Thus, second person can only observe

the results without knowing what is exactly going on the other side.

In general, to build up transitions of HMM that can represent the spoken words (that actually deals with signal), it is necessary to find out the parameters needed to represent the speech and their corresponding transcription. After finding out the parameters, HMM transitions are assigned to these parameters, and the process is called the training of HMM For training of HMM, Baum-Welch algorithm is used which is a well-known forward backward algorithm [5]. The Viterbi and forward algorithms are used for decoding the speech signals [6]. These algorithms helps in solving the HMM's problem (problem of decoding and training). Hidden Markov Model Toolkit is a toolkit used for creating and shaping the HMM. In early stages, HTK was designed for creating speech processing tool based upon HMM. For its execution, it mainly requires the UNIX Operating System and contains a set of library module which are written using ANSI C language. Now a days, it can run not only on UNIX Operating System but also on other Operating System environments. There after HTK tool has been developed gradually. There are many versions of HTK are present these days. The latest version of HTK was HTK 3.4.1 which is being used to develop a recognition system. In this study, it has been used for recognition of speech in Punjabi language.

1.2.1 Classification of HMM Observation

HMMs can be divided in two categories depends on the observations done by it. These observations are:

- Discrete Observation HMM.
- Continuous Observation HMM.

Discrete Observation HMM

HMMs are used for recognition purpose because words are made up of several distinct combinations of elements and these distinct elements in sequence are made up of phonemes. Suppose there is a word say, ਜਿਹੜੀ (ʤéɽi) which needs to be recognized. We need some learning, which is expressive enough to capture the sounds well, such as, first it sounds like 'je' for a shorter time, then it is 'hhhh' for a longer time, then it is 'ri' for a longer time. HMMs are good enough to capture these sounds that can be represented using states, probability transition matrix and distribution associated with each state from which observations are drawn.

While using speech recognition system in real world scenario, the phoneme model themselves as left-to-right HMMs (*e.g.,* to model the HMMs for phoneme, a stationary part is sandwiched between two

transition parts). After concatenation of smaller phonetic HMMs, larger HMMs are produced which represent the words.

In Discrete Observation HMM, observation sequences are drawn from discrete distribution associated with each state and in Continuous Observation HMM, observation sequences are drawn from continuous distribution associated with each state and these observations are scalars or vectors [6].

Continuous Observation HMM

In Continuous Observation HMM, all the notations used in Discrete Observation HMM denote the similar meaning except the observation sequences. In this, $b_j(o_p)$'s are computed as some probability density functions or mixtures of them. Some restrictions must be placed in so that the re-estimation of the parameters of Probability Density Function (PDF) is done. The only restriction present here is, this PDF can be only log-concave or elliptically symmetric density [6].

1.2.2 Classification of HMM Structure

HMM structures are classified into 2 categories based on the network topologies that are inherently possessed in them. These topologies are:

- Ergodic topology
- Left-to-Right Topology (or Bakis Model)

Ergodic Topology

In this topology, no pattern or sequence is followed [7]. Here, all states are approachable to each other. Thus, this topology is not suitable for speech recognition system.

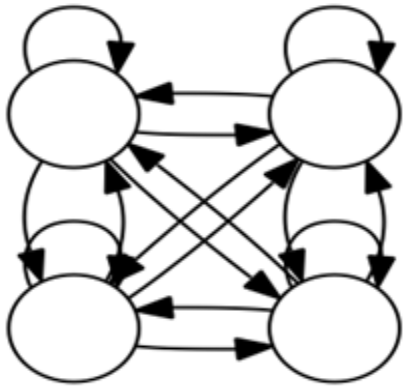


Figure 1.1: Ergodic Topology [7]

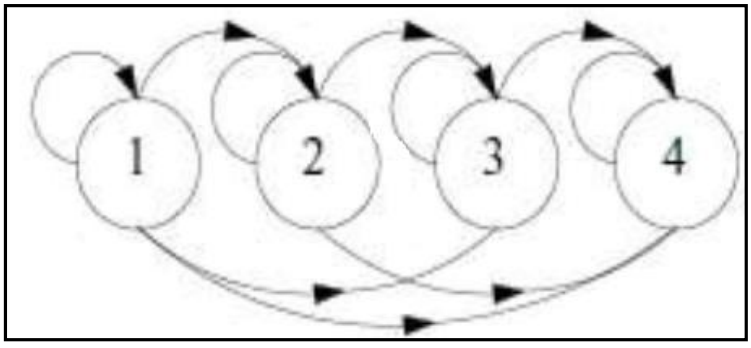


Figure 1.2: Left-to-Right Topology [7]

Because, speech signals needs to follow a particular sequence of sounds and process according to that order only. The concept of Ergodic topology is shown in figure 1.1.

Left-to-Right Topology (or Bakis Model)

In this topology, an ordered sequence is followed by the states [7]. Every state is reached through an

ordered sequence. While following a particular sequence, if any state is left, then it cannot be reached again. Here, the states are reached only in one direction, *i.e.* from left to right. Thus, this topology is well suitable for speech recognition. The overall working of left-to-right topology is shown in figure 1.2. The transition probability matrix for this topology can be represented by the matrix 1 as shown.

$$A = \begin{bmatrix} a11 & a12 & 0 & 0 \\ 0 & a22 & a23 & 0 \\ 0 & 0 & a33 & a34 \\ 0 & 0 & 0 & a44 \end{bmatrix} \quad (1)$$

1.2.3 Basic Issues in HMM

There are some problems that are associated with HMM, out of which three problems are of main interest, which are arising while applying the HMM model for recognition task. These are:

- Learning Problem
- Decoding Problem
- Evaluation Problem

Learning Problem

In learning problem, a number of hidden and visible states and some observation sequences are gathered from training the data. Here, HMM parameters which are best fit for training the data, need to be determined from the observations given prior. Let's suppose:

$O = o_1, o_2, \ldots, o_p$ are the given observations, which are general structure of HMM.

$M = (A, B, \pi)$ are the HMM parameters need to be determined

To resolve this issue, Baum-Welch Re-Estimation algorithm [6] is used.

Decoding Problem

In decoding problem, some observation sequences and the HMM parameters are given. From these two, the sequence of hidden states need to be concluded, which can produce this observed sequence. Let's suppose:

$M = (A, B, \pi)$, parameters of HMM

$O = o_1, o_2, \ldots, o_p$, observation sequence

Now, the most likely sequence of hidden states need to be computed which can produce this observed sequence. To resolve this issue Viterbi algorithm [6] is used.

Evaluation Problem

In evaluation problem, it is evaluated that the model M has produce the sequential arrangement which in turn computed from the sequence of the observations given.

HMM $M = (A, B, \pi)$ and,

$O = o_1, o_2, \ldots, o_p$, observation sequence

Here, the chances that model *M* has produced this sequential arrangement is to be computed. To resolve this issue forward-backward algorithm [6] is used.

1.2.4 Elements of HMM

These are finite number of states '*N*' such that signal possesses distinct properties within each state [6]. At every clock time 't' a new state is entered, therefore the probability distribution depends mainly on its previous state. Based on probability distribution depending only on the current state, an observation output symbol is generated after each transition is made. Thus, there are '*N*' observational probability distributions.

1.2.5 Uses of HMM

One of the most common usage of HMM is for speech recognition where the speech audio waveform is the observed data and the spoken text is the hidden state, *i.e.*, HMMs are used to recognize spoken utterances in the speech given sequence of observations [6].

$O = o_1, o_2, \ldots, o_p$. Utterances may be a word, phoneme or a sentence.

In Speech Recognition, given a observation's sequence, the probably corresponding states sequence would be estimated using Viterbi algorithm, the probability of the sequence of observations would be computed using forward algorithm, and the Baum–Welch algorithm would estimate the initial probabilities, probability of transition, and the observation's function of a HMM, *i.e.*, parameters (A, B, π)

1.2.6 Principles of HMM

A speech signal is generally a realization of some encoded messages which are following a sequential arrangement of one or more symbols, this can be shown from figure 1.3. To recognize the sequence of symbols, a spoken utterance is given [6]. For this, first of all continuous speech waveform is converted in to an equally distributed discrete parameter vectors. An exact representation of speech waveform is created from the above sequenced parameter vectors. Typically duration of speech waveform if 5ms by a single vector, thus the speech waveform can be considered as a stationary one.

The parametric representations commonly used are smoothed spectra also known as linear prediction coefficients and certain other depictions derived from these. The speech vector's sequence and the underlying symbol sequences are effectively mapped through the recognizer. The two challenges that

are to be faced in this are really hard. The first issue is no one to one mapping from symbols to speech. Different symbols can result in similar speech sounds. The second major issue is there is no wall between the speech waveform and symbols that makes it hard to make recognition between the two. Therefore, it is hard to treat the speech waveform as a series of concatenated static patterns. The second problem can be overcome by limiting the work to isolated word recognition.

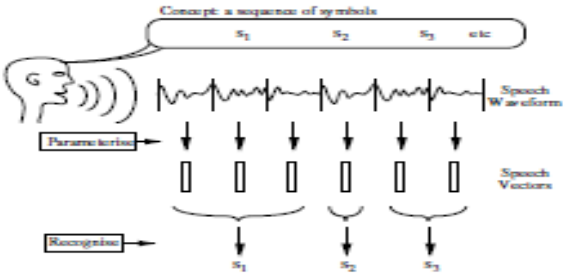


Figure. 1.3: Message Encoding/Decoding [6]

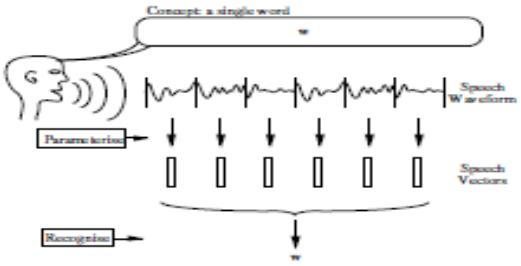


Figure 1.4: Isolated Word Problem [6]

As presented in figure 1.4, this specifies that the speech waveform correlates with an underlying symbol that is chosen from the particular fixed vocabulary.

1.3 Julius

Julius is a large vocabulary continuous speech recognition (LVCSR) engine. It is a high performance decoder software which is used by speech related researchers and developers [13]. It incorporates major search techniques like tree lexicon, enveloped beam search, Gaussian selection, Gaussian pruning *etc.* It is not only used for searching process, but also used for modularizing the model structures in a way that independent model structures are obtained. Along with this various HMM types such as tied-mixture models and shared-state triphones are supported too and with any number of phones, mixtures or states.

The main platform for working of Julius is Linux and other Unix Operating Systems. It can also perform well on Windows Operating system.

1.4 Transcription Using IPA Chart

The visual representation of speech sounds (phones) is called the phonetic transcription. IPA (International Phonetic Alphabet) is the most common type of phonetic transcription. Suppose there is a word ‘ਮੁੱਖ ਗੱਲ’in Punjabi, this can be transcribed as ‘mʊkkʰ gəll’ using the IPA chart. Phonetic transcription deals with the sound of phones used in words, *i.e.,* it tells us about the pronunciation of the words. Another advantage of transcribing words using IPA chart, is that all the languages can be

transcribed into IPA format, thus it becomes easier to make computer understand utterances of the language used.

Transcription can be done at phoneme level, word level or at syllable level. In this paper, main focus has been put on the transcription at phoneme level. Some of the examples of phonetic transcription examples are shown in table 1.1 below:-

Table 1.1: Symbolic Form Representation of Spoken Utterances

Punjabi	Transcription
ਸਾਰਾ ਮਨੁੱਖੀ ਪਰੀਵਾਰ ਆਪਕੀ ਮਹਿਮਾ	sara mənʊkʰi pərɪvar apki məhɪma
ਜੀ ਆਇਆਂ ਨੂੰ	ʤɜ ɑɑeːiaŋ ŋʊʰ
ਸੱਜੇ ਹਥ ਮੁੜ ਜਾਣਾ	sɜʤʤɜe hɜtʰ mʊɳ ʤaɳa

[10] For converting text to IPA certain steps need to be followed. The architecture representing the above explained steps is shown in figure 1.5.

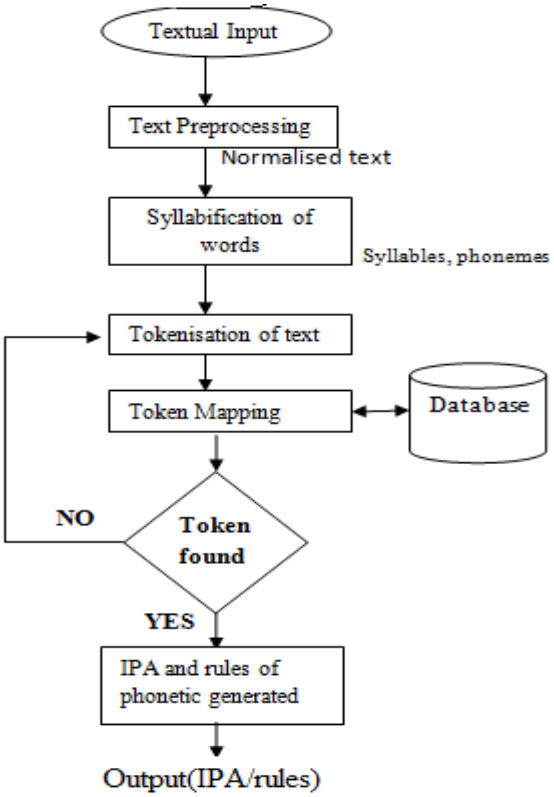


Figure 1.5: Architecture for text to IPA conversion [8]

For Punjabi language, the conversion system which will convert the text to IPA depends on syllables. First of all, analysis of the text is done which is entered by the user and then normalization of the text is done. This normalized text is further transferred to the syllabification module, where segmentation

of text into syllables is performed [9]. The combination of different vowels (V) and consonants (C) pair, make a syllable. The pairs can be like, V, VC, VCC, CV, CVC, CVCC and CCVC. In Punjabi all the combinations are accepted except the CVCC combination.

Now, the generated syllables are further transferred to the tokenizer which will break the syllables into tokens. Further, the token mapper module will help in mapping these generated tokens with their associated IPA representation [10].

The output of the conversion system is the result of the phonetic representation of every letter which can be represented with the help of IPA chart [11].

Now, the speech synthesizer module is used to convert the IPA symbols into sounds and various techniques are used for this purpose [12].

The main purpose of using IPA, is that it uses only a single letter to represent each distinguish sound *i.e.* distinguish sound segment. Thus, it does not make combinations of different letters to represent a single sound.

The IPA chart used in this work for transcription purpose is given below which is divided into parts.

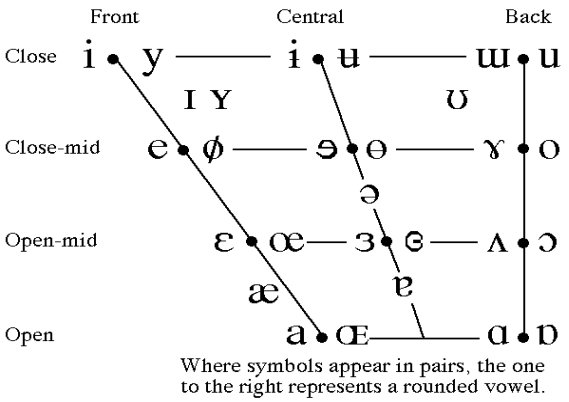


Figure 1.6: Vowels representation of IPA

CONSONANTS (NON-PULMONIC)

Anterior click releases (require posterior stops)	Voiced implosives	Ejectives
ʘ Bilabial fricated	ɓ Bilabial	ʼ *Examples:*
ǀ Laminal alveolar fricated ("dental")	ɗ Dental or alveolar	pʼ Bilabial
ǃ Apical (post)alveolar abrupt ("retroflex")	ʄ Palatal	tʼ Dental or alveolar
ǂ Laminal postalveolar abrupt ("palatal")	ɠ Velar	kʼ Velar
ǁ Lateral alveolar fricated ("lateral")	ʛ Uvular	sʼ Alveolar fricative

Figure 1.7: Consonants of IPA

DIACRITICS Diacritics may be placed above a symbol with a descender, e.g. n̊ŋ̊

◌̥	Voiceless	n̥ d̥	◌̤	Breathy voiced	b̤ a̤	◌̪	Dental	t̪ d̪
◌̬	Voiced	s̬ t̬	◌̰	Creaky voiced	b̰ a̰	◌̺	Apical	t̺ d̺
◌ʰ	Aspirated	tʰ dʰ	◌̼	Linguolabial	t̼ d̼	◌̻	Laminal	t̻ d̻
◌̹	More rounded	ɔ̹	◌ʷ	Labialized	tʷ dʷ	◌̃	Nasalized	ẽ
◌̜	Less rounded	ɔ̜	◌ʲ	Palatalized	tʲ dʲ	◌ⁿ	Nasal release	dⁿ
◌̟	Advanced	u̟	◌ˠ	Velarized	tˠ dˠ	◌ˡ	Lateral release	dˡ
◌̠	Retracted	e̠	◌ˤ	Pharyngealized	tˤ dˤ	◌̚	No audible release	d̚
◌̈	Centralized	ë	◌̴	Velarized or pharyngealized	ɫ			
◌̽	Mid-centralized	e̽	◌̝	Raised	e̝	(ɹ̝ = voiced alveolar fricative)		
◌̩	Syllabic	n̩	◌̞	Lowered	e̞	(β̞ = voiced bilabial approximant)		
◌̯	Non-syllabic	e̯	◌̘	Advanced Tongue Root	e̘			
◌˞	Rhoticity	ə˞ a˞	◌̙	Retracted Tongue Root	e̙			

Figure 1.8: Diacritics of IPA

Figure 1.6 represents the vowels of IPA chart, figure 1.7 represents the consonants of IPA chart and figure 1.8 represents the diacritics of IPA chart.

1.5 The HTK

HTK is a consolidated suite of software tools that are being used for creating and molding continuous density HMM. It consists of a number of tools along with the library modules set [1].

HTK tool was developed by Steve Young in 1989 for research related to speech recognition. He developed it at Cambridge University under the Speech Vision and Robotics Group. The training and testing of the system is done by the HTK which manipulates the HMM parameters [1].

1.5.1 Fundamentals of HTK

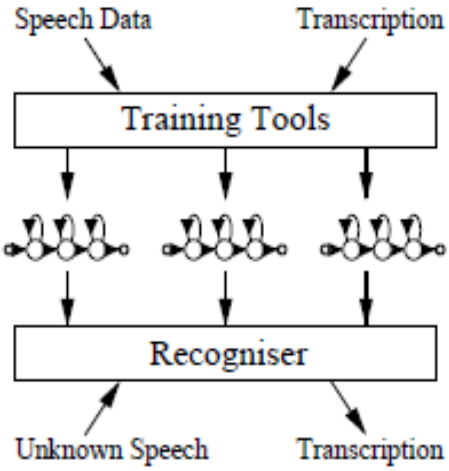


Figure 1.9: Working of HTK

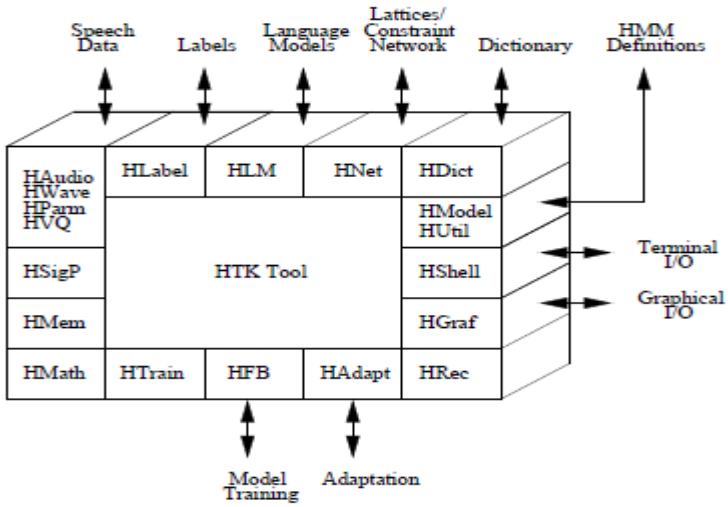


Figure 1.10: Software Architecture [6]

HMM is used to create models depending on different time series and the main purpose of HTK is the same. Primarily, HTK is used to build speech processing tools based on HMM models. The figure 1.9 shows that HTK has two major processing stages, namely training and testing (recognizing). In the first stage, HTK training tools use utterances of training data and their corresponding transcriptions to estimate HMMs parameters. In former stage, with the help of HTK recognition tools, unknown utterances are transcribed [6].

1.5.2 HTK Software Architecture

The library modules contains the major functionality of HTK [6]. The interaction of every HTK tool to the outer world is exactly the same way it is inside the HTK, this is ensured by the library modules. Figure 1.10 illustrates the software architecture of HTK tool and also represents its input/output interfaces.

HShell [6] is a library module which is used to control all the input/output operations performed by the users and the communication of the users with Operating System. *HMem* module is used to control all the memory management. *HMath* module provides the support to Mathematical functionality and *HSigP* controls the operations of signal processing which are desired for speech analysis. Each and every file type which is recommended by HTK has a committed consolidate module. The consolidation for label files are provided by *HLabel*, the model files used for language uses the HLM tool, *HNet* for the lattices and networks, *HDict* is used for the dictionaries and *HModel* is used for the HMM definitions.

All the input and output functions of speech, at waveform level are carried out *via HWave* tool and at parameterized level carried out *via HParm* tool. To import data from one system to another, *HWave* and *HLabel* tools are used because they provide a consistent interface as well as support multiple file formats. *HAudio* is used to support direct audio processing. At last, the main recognition process is taken out *via HRec* tool.

1.5.3 The Toolkit

HTK tool kit is a statistical tool to build HMM models [6]. The main objective for designing this toolkit is to build HMM-based Speech Processing tools, specifically recognizers. It is mainly concerned with HMMs, whose probability distribution of observations is personified *via* Gaussian mixture density [33]. It consists of two major processing stages.

Using training utterances (recorded speech used for training purpose) and their corresponding transcriptions, the HTK training tools compute the parameters of set of HMMs. Recognition tools of HTK are used to transcribe the Unknown utterances (recorded speech, transcription of which is to be done). Working of speech recognition system is shown in figure 1.11.

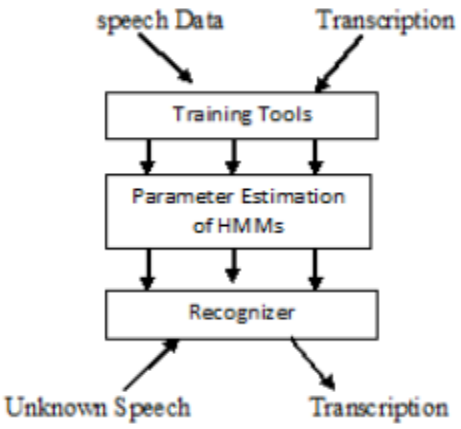


Figure 1.11: Working of speech recognition system

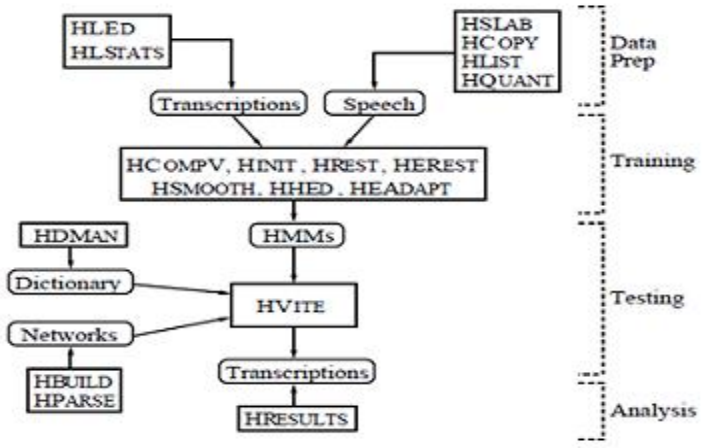


Figure 1.12: HTK Processing Stages [6]

In order to introduce the HTK tools, the best way is to follow the steps for processing which are involved in creating the continuous speech recognizer based on sub-words. As shown in figure 1.12, four main phases are involved in this procedure:

- Data preparation
- Data training
- Data testing
- Data analyzing

Data Preparation Tools

A bunch of speech files and the corresponding transcription files are needed to create a set of HMMs. Before it is used for training, it is necessary to convert it into a convenient parametric form. Thus, any corresponding transcription file need to be transformed such that appropriate format is created and the recommended phone or word labels can be used properly.

For Transcription:-

HLEd: This is an editor which is constructed for making the appropriate changes to the label files and the results are sent to a single Master Label File (MLF). This MLF file is commonly much commodious for handling the subsequent files.

HLStats: The gathering and the presentation of statistics data on the label files and where ever required is done by this tool.

Training Tools

In second step of system development, prototype definition is written for each HMM, so that the required topology for each HMM can be defined. The only purpose of defining a prototype is to enumerate the overall characteristics and topology of HMM. Here, no actual parameters are evaluated. The transitions came out from any state, made equally likely with the help of these probabilities.

The actual training process takes place in stages. First of all, an initial set of modules is created. The same can be used as a bootstrap data if some of the speech data is achievable for which sub-word's (*i.e.* phone) boundaries have been marked. In such cases, *HInit* and *HRest* tools use fully labelled bootstrap data to give provision for an isolated style training. The training data is uniformly segmented on the first cycle. Here, corresponding data segments are matched with each model state, so that, means and variances can be computed. Gaussian models mixtures are trained to use a mutated version of k-means clustering. Successively, on the second cycle, the uniform segmentation has to be replaced by Viterbi alignment [6]. First of all, *HInit* computes the initial parameters and then through

HRest further re-estimation is done. Now, segmental *k*-means procedure is replaced with Baum-Welch re-estimation [6] procedure using once again the same bootstrap data.

Table 1.2 displays the brief summary of all the commands used in different tools at different stages.

Table 1.2: Brief Summary of Commands of HTK

Sr. No.	**Command**	**Phase**	**Description**
1	HLEd	Data Preparation	Label editor designed for making the appropriate changes to the label files and store results in a single Master Label File (MLF).
2	HLStats	Data Preparation	Gather and display statistics on label file
3	HSLab	Data Preparation	Record and manually annotate the speech
4	HCopy	Data Preparation	Main objective is to copy the data of one or many source files and sends the data in to an output file.
5	HList	Data Preparation	Check contents of speech and check results of conversions
6	HQuant	Data Preparation	VQ codebook is built for discrete probability HMM
7	HInit	Data Training	Reads all of the data used in bootstrap training and all the required phones examples are cut out.
8	HRest	Data Training	Further re-estimate the parameter values computed by HInit
9	HCompV	Data Training	When all the phone models discovered, are identically initialized to have the state variances and means equal to global variances and means.
10	HERest	Data Training	Has the capacity to operate large databases, thus used for performing the embedded training by using the whole training data set
11	HSmooth	Data Training	Used to smoothing the distributions when data insufficiency problem occurs due to discrete probability systems and fully tied mixture systems

12	HHed	Data Training	Used as HMM definition editor
13	HEadapt	Data Training	Adaptation of HMMs set is performed using MLLR, MAP, or both. By default MLLR is chosen.
14	HDMan	Recognition	Used for preparing a pronunciation dictionary from different sources. From the given scrip file a list of editing commands are read out and the results of modified and joined copies of one or more dictionaries came out as outputs.
15	HVite	Recognition	Used to evaluate the lattices and models using language.
16	HLRescore	Recognition	Used for manipulating lattices.
17	HDecode	Recognition	Similar to HVite command, transcription of speech files is done using a set of HMM model and a dictionary
18	HBuild	Analysis	Used to convert language model input files into a standard HTK lattice output file.
19	HParse	Analysis	Word level lattice files are generated using a syntax description text file which contains a bunch of modified rules based on the extended Backus-Naur Form (EBNF).
20	HResults	Analysis	In order to assess the performance of the recognizer based on HMM, pre transcribe some test sentences which are pre-recorded. And then, match the output obtained from recogniser with the actual reference transcriptions.

Recognition Tools

HVite is an HTK recognition tool that allows the evaluation using language lattices and models. A tool named *HLRecsore*, used for generating the lattices with the help of HVite *(*or HDecode*)* to be manipulated. *HDecode* is to be distributed under the more restrictive license agreement.

Analysis Tools

As soon as the recogniser based on HMM is built, it is compulsory to assess its performance. Mainly, this is carried out by transcribing some of pre-recorded sentences used for testing and then match the outcome generated by the recogniser with the original transcriptions. For this purpose, *HResults* tool is used which will adjust both the transcriptions using dynamic programming, and thus counting of replaced, inserted and deleted errors is done. *Hresults* provides a speaker-by-speaker breakdowns, transcriptions aligned timely and confusion matrices for the sake of global performance measures.

CHAPTER 2

Literature Review

This section will imitates the literature review similar to the presented work.

Rabiner *et. al.* (1989) [14] proposed connected digit recognition system based on HMMs. Proposed system was trained and tested with three different kinds of data: speaker trained, multi-speaker and speaker independent. Evaluation was done on three databases: database widely distributed through National Bureau of Standards, 225 adult talker and 50 talker connected digit database. 0.78, 2.85 and 2.94 were the error rates corresponding to string that was observed from all the three modes.

Lee *et. al.* (1989) [15] proposed speech recognition system using SPHINX tool. This tool depends upon discrete HMMs and LPC which are used as feature vectors. In order to train 48 context independent phonetic HMMs, 4200 sentences were used spoken by 105 speakers. Phone HMMs were concatenated to build word HMMs which further concatenated to build large sentence HMM. It was observed that when words occur in cluster it becomes difficult to recognize them. Training was carried out in two different stages: in first one, training of 48 context independent phonetic HMMs were done and in second stage trained models from first stage initialized context dependent phone models. This system was evaluated on 150 sentences spoken by 15 speakers. With word-pair grammar, word recognition accuracy of 96.0% was obtained and with null grammar word recognition accuracy was 82.0%.

Lee *et. al.* (1989) [16] proposed speaker independent phone recognition system which was based on discrete HMMs. This system was recognized with the help of TIMIT database which consists 6300 sentences from 630 different speakers. In this system, HMMs are trained using sentences based on TIMIT which was collected from 357 speakers and was recognized with 160 TIMIT sentences collected from 20 speakers. LPC was used as a feature extraction technique. A new novel smoothing algorithm which smooth out the HMM output parameters was also introduced in this work. For 39 English phones, 64.0% was the accuracy rate with context-independent phone models and with context-dependent phone models accuracy rate was 73.8%.

Lamel *et. al.* (1992) [17] proposed a phoneme recognition system, which was speaker independent and used for continuous speech. The work was proposed for French language. For training to HMMs, data was collected from 43 speakers. For testing, data from new 19 speakers was collected using large

read speech corpus BREF. For 35 context independent phoneme models, 60.0% of phone accuracy was obtained and with 428 context dependent models, 68.6% of phone accuracy was obtained.

Ratnayake *et. al.* (1992) [18] proposed a phone recognition system which is speaker independent based on hidden semi Markov models (HSMMs). In this work, HSMMs were used to overcome the limitation of HMMs. Instead of parametric distributions, non-parametric distributions were used. The technique used for feature extraction in this system is LPC. As a result, it was observed that with HSMMs phoneme recognition accuracy was 53.7% and with HMMs it was 48.4%. One drawback that was observed with HSMMs was that it has high computational complexity as compared to HMMs.

Woodland *et. al.* (1995) [19] proposed speech recognition system based on HTK toolkit. This system was based on tied-state context dependent continuous density HMMs, *i.e.*, Gaussian mixture HMMs. MFCCs was used as a feature extraction technique which included twelve cepstrum's, normalized log energy, acceleration and delta coefficients, *i.e.*, first and second order derivatives of parameters. In this work, vocabulary consisting of 65464 words, four-gram language model was used.

Leggetter *et. al.* (1995) [20] proposed a linear regression technique which is meant for speaker adaptation using continuous density approach, *i.e.*, Gaussian mixtures HMMs. Modeling of new speaker was improved using an initial speaker independent system by updating the HMM parameters. Experiments were performed on databases of ARPA RM1 using HMMs having continuous density mixtures output distribution and cross word tri-phones. It was observed that with supervised adaptation 37.0% error reduction was achieved and with unsupervised adaptation 32.0% of error reduction was achieved using 40 adaptation utterances.

Mari *et. al.* (1996) [21] proposed a second order HMM for continuous speech recognition used for phones as well as words. In this work, it was shown that second order HMM yield better performance than first order HMM. Data was collected from speech telephone corpus and experiments were done on spelled names over telephone. More than 4000 people were asked to spell their first and last name with and without pauses over telephone and their voice were recorded. This was the speaker independent system. For training purpose 1200 calls and for testing purpose 491 calls were selected. It was observed that the second order HMMs can achieve more than 69.0% of accuracy.

Ming *et. al.* (1998) [22] proposed phone recognition system for continuous speech signal based on Bayesian trip hone models. Here, a new statistical framework was introduced for building tri-phone models using models of less context dependency. This method somewhat was different from the

previous models as it was based on the Bayesian principle not on the heuristic method. This system was used for the evaluation of 39 phones based on TIMIT database. Performance of proposed system was tested on two test set: core and complete test set. With core test case accuracy was 74.4% and with complete test case accuracy was 75.6%.

Zheng *et. al.* (1999) [23] proposed an Easytalk application, *i.e.*, Chinese dictation machine (CDM).This application was developed for recognizing the continuous speaker-independent Chinese speech having a large vocabulary. CDM engine included automation of merging based syllable detection, frame synchronous search algorithms based on statistical knowledge, methods for rejecting and accepting the decisions, critical area percentage and syllable synchronous network search. LPC was used as a feature extraction technique. In this application, it was observed that Cepstrum was not the best feature. CMD achieved 98.0% accuracy for in-vocabulary commands and 95.0% accuracy for out-of-vocabulary commands.

Young (1999) [24] presented an acoustic model for large vocabulary continuous speech recognition (LVCSR). The main purpose of LVCSR was to transcribe speech given as input into an orthographic transcription. It was assumed that input speech consists a sequence of words and using the language model probability of any specific sequence of words. This was an N-gram model. MFCCs were used as a feature extraction technique. It was observed that to get the good phonetic discrimination, for each different context HMMs required to be trained and the most common context was tri-phone. Cross-word tri-phones provided the best modeling accuracy but too many parameters required to be computed. To overcome that, state-tying context was used. This involved the concept of mixture splitting.

Pruthi *et. al.* (2000) [25] proposed the implementation of speaker dependent real time recognizer for Hindi language for isolated words. This recognizer was named as Swaranjali. This system was based on HMMs. Data had been collected from two different male speakers who had been asked to utter Hindi digits from shoonya (0) to nau (9) two times means using total 20 tokens HMMs were trained. After training, evaluation was performed on the proposed system to check the accuracy. Some errors had been recognized too due to the existence of plosives at the beginning as well as at the ending of some of the words. On an average 84.5% was the accuracy of system for speaker 1 and for speaker 2 it was 84.3%.

Woodland *et. al.* (2002) [26] proposed a framework for continuous density HMMs. The proposed work provides the discriminative training to the large vocabulary speech recognition systems. The

maximum mutual information estimation (MMIE) method was used to train the HMMs. For conversational telephone speech transcription, 265 hours of data was used for training the HMM models. In this, tri-phone and quin-phone HMM parameters were estimated which led to reduction in word error rate for the transcription of conversational telephone speech. Also, a scheme which reduced the danger of over training was also shown. This scheme was based on linear interpolation of MMIE and MLE objective functions.

Nwe *et. al.* (2003) [27] proposed a method which is text independent for classification of the emotions of speech. This method depends on discrete HMMs which was used as classifier and log frequency power coefficients (LFPC) was used to depict the speech signals. In this system, emotions were classified into six categories fear, disgust, anger, joy, surprise and sadness. Data was collected from twelve speakers, each of which was asked to utter 60 emotional utterances. A comparison of LFPC feature parameters was made with Linear Prediction Cepstral Coefficients (LPCC) and MFCC feature parameters. It was observed that LFPC as feature parameter showed better performance than other traditional feature parameters. Proposed system showed 78.0% average accuracy on evaluation.

Hasan *et. al.* (2004) [28] proposed speaker recognition system. This is a security system based on speaker identification. MFCCs was used as a feature extraction technique. This system was implemented using Matlab 6.1 in windows XP environment. Data set prepared for this particular system which consists of 21 speakers (13 male speakers and 8 female speakers). Identification rate was used as a measure for estimating the realization of the system. Identification rate is the ratio of total number of identified speakers to the total numbers of speakers tested. Identification rate was measured using three windows triangular, rectangular and hamming on two scales one was linear frequency scale and another was Mel-frequency scale. When linear frequency scale was used it was observed that identification rate was directly proportional to codebook size, as the codebook size increased identification rate was also increased and when the codebook size was 16 identification rate was 100.0% for both triangular and hamming windows. When Mel-frequency scale was used, relation between identification rate and codebook size was same as it was in the case of linear frequency scale but in this case 100.0% identification rate was observed when size of codebook was 4 and hamming window was used.

Satori *et. al.* (2007) [29] proposed a novel approach for building an automated Speech recognition System for Arabic language. For building this system, utilities of Sphinx-4 engine were used which is the open source from Carnegie Mellon University (CMU). This tool is based on discrete HMMs.

Difficulties that were faced in developing this system for Arabic language was that non-diacritized content was in larger amount, huge variety of dialectal and lastly, morphological complexity. This system was designed for recognizing the ten Arabic digits and this system was named as Hello_Arabic_Digit application. Data corpus prepared particularly for this system consisted of six male speakers who were asked to utter all ten digits five times. For training purpose, all 300 utterances were used. For checking the performance of trained system three different male speakers were asked to utter all ten Arabic digits. Mean recognition ratio for each one was computed for speaker 1 it was 86.7%, for speaker 2 it was 86.7% and for speaker 3 it was 83.3%.

Bhuriyakorn *et. al.* (2008) [30] presented phoneme recognition of continuous speech of Thai language. In this work, an approach of estimating HMM topology was proposed, whole process was divided into two stages: by combining different objective functions and topology generation methods a set of suitable topologies were constructed and a genetic algorithm was used as the topology selection algorithm considering global fitness. As a result, about 4.4% of error reduction in well-trained topologies was observed over already defined left-to-right HMM models.

Elshafei *et. al.* (2008) [31] proposed speaker independent natural Arabic speech recognition system. This system was based on HMMs and was developed using Sphinx tools. This system was tri-phone based acoustic model using five states HMMs in which first and last state was non-emitting and all other states were emitting ones. It used continuous density of eight Gaussian mixture distributions. Total 5.4 hours of data was used for training as well as for testing purpose out of which 4.3 hours of data was used for the training purpose and the remaining data of 1.1 hours was used for testing purpose. In pronunciation dictionary 14,232 words were defined and language model contained both bi-grams and tri-grams. After testing the system, word error rate was observed to be 9.0%.

Alotaibi (2008) [32] proposed Arabic digit recognition system and did a comparative study of HMM and artificial neural network (ANN). Proposed system was implemented using HMM and was a recognizer based on isolated word phonemes. After evaluating the performances of both recognizers it was observed that ANN based recognizer obtained 99.5% accuracy in multi-speaker mode and 94.5% accuracy in speaker independent mode while HMM based recognizer obtained 98.1% accuracy in multi-speaker mode and 94.8% accuracy in speaker independent mode.

Azmi *et. al.* (2008) [33] proposed automatic speech recognition for Arabic speech using syllables and did a comparison of mono-phone, tri-phone and word based recognition with syllable based recognition. Proposed system was based on HMMs and was implemented using HTK toolkit. For

training and testing purpose, data was collected from forty four speakers. MFCCs were used as feature extraction technique. It was observed that syllable based recognition overcome the performance of other recognizers as recognition rate of mono-phone based recognizer, tri-phone based recognizer, word based recognizer was 90.7%, 92.2%, 91.6% and of syllable based-recognizer, it was 93.4%.

R.Kumar (2010) [35] implemented a real time, speaker independent isolated word recognizer for Punjabi language. The work had been further extended to compare the performance of speaker dependent isolated spoken words for small vocabulary using the Hidden Markov Model (HMM) and Dynamic Time Warp (DTW) technique.

Al-Qatab *et. al.* (2010) [36] implemented an automatic speech recognition engine based upon Arabic language using HTK. The system was built to recognize both isolated words as well as continuous speech in Arabic language. MFCC was used as the feature extraction technique. To compute HMM parameters training were based on tri-phones. A manually built Arabic dictionary from speech-sounds of thirteen speakers was used for the developed system and a vocabulary of 33 words was used for the same. For this work data has been collected from thirteen Arabian native speakers and then further divided into two categories: speaker dependent and speaker independent. The data of thirteen speakers was divided as ten for speaker dependent and three for speaker independent. For both Continuous and isolated kinds of speech, it was observed that the system was able to recognize the speech for speaker dependent and speaker independent categories. The thorough performance of system was 90.60% for sentence correction, 98.00% for word correction and 97.90% word accuracy.

S. Mandal *et. al.* (2010) [37] implemented a Speech To Text (STT) system named as Shruti-II for Bengali language and this system is used for E-mail applications. The development of the system was done with the help of SPHINX3 and was developed for the people who are visually challenged.

R.Kumar *et. al.* (2011) [38] developed an isolated word ASR system for Punjabi language for speaker independent data and in real time environment. For evaluating the system Dynamic Time Warping (DTW) and Vector Quantization approaches were used. This system has been developed for vocabulary of small isolated words.

Kumar *et. al.* (2011) [39] proposed Speech Recognition System for isolated words of Hindi language. MFCC was used as a feature extraction technique. This system was based on HMMs and developed using HTK toolkit on Linux platform. A vocabulary of size of thirty words was used for training purpose. This training data was collected from eight speakers. At the time of performance evaluation which was conducted in room environment, the overall accuracy of the system that had been observed

was 94.60% and word error rate was 5.40%.

K.Kumar *et. al.* (2012) [40] implemented a speech recognition system for Hindi language of connected words. The system was prepared with the help of HTK. A vocabulary of 102 words were used in this system. Thus, the system was trained for recognizing any sequence of words. The data had been collected from both male and female speakers. For training purpose data had been collected from 12 speakers and for the testing data had been collected from five speakers. These collected data were used to analyze the performance of the recognizer. MFCCs features were extracted from the speech signal. Once extraction of features is done, the system was trained for computing HMM parameters using word level acoustic models. Many experiments were done in different environments such as open space, lab room, room environment, and classroom and in market. It was observed that with the increase of noise level, the system's performance started to degrade and the thorough accuracy of system was 87.00%.

Choudhary *et. al.* (2013) [41] proposed automatic speech recognition for Hindi language using HMM. Proposed system was developed for recognizing the isolated and connected words of Hindi speech and implemented using HTK toolkit. System was trained for hundred different isolated words and each word was uttered ten times.

Saini *et. al.* (2013) [42] proposed Hindi Speech Recognition System for isolated words based on HMMs using HTK toolkit. This system recognizes one hundred and thirteen Hindi isolated words and three different states (6, 8, and 10) HMM topology was used. For recognizing the speech, word model was used. MFCC features were extracted for data parameterization. Data was collected from 6 speakers for training purpose. This system was developed on Linux environment. In this system, recognition involves two cases: recognition by speakers involved in both training and testing using three different states in HMM topology and recognition by speakers involved only in testing using three different states in HMM topology. In first case, with 6 states in HMM topology accuracy was 93.9%, with eight states in HMM topology accuracy was 91.7%, with 10 states in HMM topology accuracy was 96.6%. In second case, with 6 states in HMM topology accuracy was92.7%, with 8 states in HMM topology accuracy was 91.2%, with 10 states in HMM topology accuracy was 95.5%.

Sarma *et. al*. (2013) [43] proposed Phonetic engine development for Assamese language and discussed some issues related to it. Proposed work was based on HMMs and is implemented with the help of HTK toolkit. It was a phoneme based recognizer. Here, speech data was recorded in three different modes: read speech mode, lecture speech mode and conversational speech mode. Read

speech data was used to train HMMs and performance accuracy was achieved in all three modes was 47.3% in read speech mode, 45.3% in lecture speech mode and 36.1% in conversation speech mode. Mankala *et. al.* (2014) [44] proposed automatic speech recognizer for Telugu language. This automatic speech recognizer was implemented using HTK toolkit. This was developed for recognizing isolated words and acoustic word model is used for this. Data was collected from nine Telugu speakers for training purpose and system was trained using 113 isolated Telugu words. The overall accuracy of the system that was observed was in the range of 95.46% and 96.64%.

The overall description of all these papers are described in table 2.1.

Table 2.1: Brief Summary of all literature review

Sr. No.	**Author Name**	**Year**	**Tools/ Technique Used**	**Language**	**Results**	**Further Work**
1.	Rabiner *et. al.*	1989	HMM	----	0.78, 2.85 & 2.94 string error rate for 3 different modes	Connected digit recognition system Train & Test system in three different modes
2.	Lee *et. al.*	1989	SPHINX	English	96.0%, 82.0% with word pair grammar & null grammar respectively	HMM & LPC as feature vectors Phone concatenate to form word, word concatenate to form large sentence
3.	Lee *et. al.*	1989	HMM	English	64.0%, 73.8% with context independent & dependent	Novel smoothing algorithm was introduced Smooth out the HMM output parameters
4.	Lamel *et. al.*	1992	HMM	French	60.0%, 68.6% with phone independent	Speaker independent phoneme recognition

					and phone dependent data	Data collected using read speech corpus BREF
5.	Ratnayake *et. al.*	1992	HSMM(Hidden Semi markov Model)	---	53.7%, 48.4% accuracy with HSMM & HMM recognition respectively	Non-parametric distribution were used LPC as a feature extraction technique
6.	Woodland *et. al.*	1995	HTK	English	---	MFCC as a feature extraction technique
7.	Leggetter *et. al.*	1995	HMM	---	37.0%, 32.0% error reduction with supervised and unsupervised adaptation	Update HMM parameters to improve modeling of new speaker using an initial speaker independent system Experiments were performed on ARPA RM1 database using HMM
8.	Mari *et. al.*	1996	HMM	---	69.0% accuracy	Data was collected from speech telephone corpus Experiments were done on spelled names over telephone
9.	Ming *et. al.*	1998	Bayesian Principle	----	75.6% accuracy	Tri-phone models are built using models of

			for triphones			less context dependency
10.	Zheng *et. al.*	1999	LPC, HMM	Chinese	98.0% & 95.0% accuracy for in-vocabulary & out-of-vocabulary commands	Recognized speaker independent large vocabulary Transcribe input speech into an orthographic transcription
11.	Young	1999	HMM, MFCC	----	----	Transcribe speech given as input into an orthographic transcription.
12.	Pruthi *et. al.*	2000	HMM	Hindi	84.5%, 84.3% accuracy for speaker1 & 2 respectively	Data was collected from 2 different male speakers Total 20 tokens are taken(digits from 0 to 9 were uttered twice in Hindi)
13.	Woodland *et. al.*	2002	HMM, MMIE	----	---	Discriminative training of vocabulary speech is done Tri-phone and Quin0phone parameters were estimated
14.	New *et. al.*	2003	HMM, LFPC, MFCC	---	78.0% average accuracy rate	HMM was used as classifier

						LFPC was used to represent speech signals Comparison of LFPC feature parameters with LPCC and MFCC feature parameters was done
15.	Hasan *et. al.*	2004	MFCC, Matlab 6.1	---	100.0% identification rate	MFCC used as feature extraction technique Identification rate is used as a measure to analyze system's performance
16.	Satori *et. al.*	2007	Sphinx-4, Discrete HMM	Arabic	86.7%, 86.7% & 83.3% for speaker1, 2 & 3 respectively	Used to recognize 10 Arabic digits Data is collected from 6 male speakers Data is collected by uttering all 10 digits 5 times For testing 3 different male speakers asked to utter all 10 digits
17.	Bhuriyakorn *et. al.*	2008	HMM	Thai	4.4% error reduction	Different objective functions and topology generating methods were combined to create a set of suitable topologies

						Genetic algorithm was used as the topology selection algorithm
18.	Elshafei *et. al.*	2008	HMM, Sphinx	Arabic	9.0% word error rate	Continuous density of 8 Gaussian Mixture were used. 4.3 hours of data is used for training 1.1 hours of data is used for testing
19.	Alotaibai *et. al.*	2008	HMM	Arabic	99.5% accuracy with ANN recognizer, 98.1% accuracy with HMM recognizer	Comparative study of HMM and ANN was done
20.	Azmi *et. al.*	2008	HTK(Hidden Markov Tool Kit) tool	Arabic	90.7% Accuracy	Comparison of mono-phone and tri-phone MFCC features extracted
21.	R.Kumar *et. al.*	2010	HMM(Hidden Markov Models) and DTW(Dynamic Time	Punjabi	---	Recognize speaker independent isolated words Comparison of performance of speaker dependent with speaker independent using small vocabulary

			Wrap) techniques			
22.	Al-Qatab *et. al.*	2010	HTK	Arabic	90.6%, 98.0% & 97.9% for sentence correction, word correction and word accuracy respectively	Recognize isolated words Recognize continuous speech Both with a vocabulary of 33 words
23.	S.Mandal *et. al.*	2010	SPHINX3	Bengali	---	E-mail application Developed for visually challenged people
24.	R.Kumar *et. al.*	2011	Vector Quantization and DTW technique	Punjabi	---	Developed a speaker independent isolated word ASR(Automatic Speech Recognition)
25.	Kumar *et. al.*	2011	HMM, HTK, Linux	Hindi	94.6% accuracy and 5.4% word error rate	MFCC used as a feature extraction technique Training data was collected from eight different speakers
26.	K.Kumar *et. al.*	2012	HTK	Hindi	87.0% Accuracy	Recognize connected words with a vocabulary of 102 words
27.	Choudhary *et. al.*	2013	HMM	Hindi	---	Recognize isolated and connected words

						System was trained for hundred individual isolated words Each word has been uttered ten times
28.	Saini *et. al.*	2013	HMM, HTK	Hindi	92.7%, 91.2% & 95.5% with 6, 8 & 10 states respectively	Recognize isolated words with 3 different states HMM topology MFCC features are extracted Developed on Linux environment
29.	Sarma *et. al.*	2013	HTK, HMM	Assamese	47.3%, 45.3% & 36.1% in read ,lecture and conversation speech mode	Phone based recognizer Data was collected in 3 different modes
30.	Mankala *et. al.*	2014	HTK	Telugu	In range of 95.46% to 96.64%	Isolated word recognizer using acoustic model Data was collected from 9 native speakers for training For testing 113 isolated words were used

CHAPTER 3

Problem statement

3.1 Research Gap

Punjabi is an Indo-Aryan language. The spoken Punjabi language, relies on Sanskrit vocabulary. Punjabi language is different among the Indo-Aryan group of languages, because of its tonal nature. Punjabi is the most widely spoken language all over the world, thus ranked 10^{th} in the world's most widely spoken language list [4]. It has more than hundred and thirty million native speakers all over the world. Majority of the people, who are speakers of Punjabi language, lives in Punjab region of India as well as of Pakistan. Punjabi language can be written in two different scripts: Shahmukhi and Gurmukhi.

Punjabi language has many dialects, which eventually creates a continuous series, represented as Hindi in India and as Sindhi in Pakistan. The main dialects of Punjabi language in India are: Majhi, Malwai, Doabi and Pwadhi. Whereas, main dialects of Punjabi language in Pakistan are: Hindko, Majhi, Multani and Pothohari. Out of these dialects, Majhi is the most important and esteemed dialect of Punjabi language, due to its standard formalization for writing in Punjabi language [4].

3.2 Problem Statement

Inspite of having tonal nature, variation arises from the emotional stress in pronunciation and this deviates the sense of speech. Gurmukhi script is written left-to-right and spelled phonetically. There are 25 consonants, 10 vowels, 3 auxiliary signs, and 7 diphthongs in this script [4]. Tonal features are phonetic and segmental in nature. Phonemes are the smallest building blocks of every word. Since, phonemes are language dependent thus changes with the pronunciation. So, it is not a good idea to use phonemes in speech recognition system, because it will make the system more and more confusing. That is why phonetics is used for recognition system instead of phonemes. These phones are than used for word level recognition system, where a combination of these phones are built and placed in a model to identify a complete word as a whole.

Punjabi language has gained wide acceptability in the fields of communication and media so it deserves to get a proper pace in the growing field of ASR. ASR has already been explored successfully for a number of other Indian as well as foreign languages. In the field of Punjabi language, some work has been done for isolated word as well as connected words.

In the proposed work, a speech recognition system for Punjabi language is developed. This work is carried out for isolated words. In development of this speech recognition system, features are extracted using MFCC and are classified with the help of HMM. HMM has been implemented using HTK toolkit. Here, a total of 7 hours data is collected from three different modes of speech. After the collection of data, this data is transcribed using IPA chart and as a result a prosodic database is created. From the total data, 75% of the data is used for training purpose and rest 25% of data is used for testing the performance of the system. The result of this system can be shown in the form of a matrix known as confusion matrix, which is a comparison of actual transcriptions (*i.e.* system generated) of the data with the expected transcriptions (*i.e.* manual) of the data. This system made to work with 29 phones and 33 phones. The phones added in the previous list to make it 33 from 29 are: 'ae', 'an', 'chh', 'oun'. The main objective of increasing these phones is to increase the accuracy as well as correctness of the system.

Once the system is trained and tested for phone level recognition, it is than trained and tested for word level speech recognition. So, for this work a total of 2 hours of data is taken out of which 90 minutes of data is used for training and rest 30 minutes of data for testing. With the help of phone list (created in phone level recognition) and transcription file, mlf file for word level is generated from which a tri-phone list is generated corresponding to each word present in the word level mlf file. That list of triphones are than used for generating hmm definitions and macros corresponding to each proto definition of different combinations of phones. After training is done these files are used for creating a mlf file which is further compared with the actual transcription (*i.e.* manual) file of word level and gives results on the terminal window.

3.3 Objectives

- To collect data in 3 different modes ,that are as follows:
  - Read Speech Mode
  - Lecture Speech Mode
  - Conversation Speech Mode
- To explore, understand and install Hidden Markov Toolkit (HTK) and Julius.
- To prepare data in a form usable by HTK and Julius.
- To train and test the system for phone level recognition.
- To train and test the system for word-level recognition.
- To develop front end and improve the performance of system for phone level.

CHAPTER 4

Data Collection and Data Preparation of Proposed System

4.1 Data Collection

Speech is a natural way for exchanging the information among human beings. It can solve a lot of problems, if it is used as a medium for interaction between human being and the computer. For this purpose, some speech interfaces are required, such as:

- Speech synthesizer
- Speech recognizer

Phonetic transcription is required for both recognizing the speech as well as synthesis the speech. In speech recognition, input is provided in the form of speech and then corresponding phonetic transcription is generated as output through the system. Automatic Speech Recognition (ASR) is such a module that uses the information of acoustic phone which is present in speech signal for converting the speech signal into symbolic form. This symbolic form is nothing more than the units of sound present in the spoken utterances of speech signal. International Phonetic Alphabet (IPA) transcription standard is used to represent these basic sound units in symbolic form. Acoustic phonetic information means that the ASR will use the sounds of phones of spoken utterances and these sounds are represented in the symbolic form.

To accomplish this task, Punjabi speech data has been collected in three different modes of speech, namely, read speech mode, lecture speech mode and conversational speech mode.

4.1.1 Read Speech Mode

In read speech mode, speech is uttered from the books by the speakers. Here, speakers just need to read the books, magazines *etc*. In this mode chances of external noise is less. To collect the data, Zoom h4n tool is used. A total of 3.77 hours (*approx.*) data was collected for read speech mode and this data is collected in studio where no external noise and disturbance exists. Here data has been collected for both genders (*i.e.* male and female) separately. For this task, speech is recorded by two male speakers and two female speakers. The training and testing was performed individually for each speaker as well as on whole data too.

4.1.2 Lecture Speech Mode

In lecture speech mode, speech is collected in the natural form of speech, just like a person is delivering the lecture. In this mode, no pre-defined script was given to the speakers, only topic was given to them. Here, speech was collected in the natural form of speech. In lecture speech mode, as lecture has to be delivered in a class room, so the chances that noise can be added in the speech increases. In this mode, tool used for collecting data is same as that of read speech mode *i.e.* Zoomh4n. Here, a total of 2.28 hours (*approx.*) data was collected. This data has been collected in a classroom. The data has been collected from 4 transcribers (who transcribe the data) and 3 speakers. The training and testing was performed individually for all of them as well as on the whole data too.

4.1.3 Conversation Speech Mode

In conversation speech mode, speech is recorded in the natural way of group discussion. The speech contains conversation and dialogues of the multiple speakers. Here data is recorded in normal room environment as well as open room environment using a microphone channel. That is why chances of noise to be added are more in this mode. Zoom h4n tool is used to record the conversation of different speakers. Here, a total of 35.45 minutes (*approx.*) of data is recorded. As data is collected in normal and open room environment so chances of addition of noise in the speech increases.

4.2 Data Preparation

ASR involves acoustic modeling (that contains the statistical representation of distinct sounds) and language modeling (that contains a very large list of words and their probability occurrence in a pre-defined sequence). Each and every statistical representation is assigned with a label known as 'phoneme'. Here, performance of a Punjabi (Gurmukhi script) isolated automated transcription is discussed. An automated transcription process is done for all the three speech modes. In the proposed work, 29 and 33 unique phonemes excluding silence has been used. From the whole data 75% of data is used for training and 25% for testing.

For building a set of HMMs, a bunch of speech data files and their corresponding transcription files are demanded. The transcription of files is generated by using IPA. Before using data for training, it is necessary to convert it into the desired parametric form and the corresponding transcriptions need to be transformed to the correct format and then use the recommended phone or word labels. In order to get good accuracy, all the recorded speech files need to be cut down in to a maximum length of 5seconds each. Figure 4.1 shows the overall training procedure and the files needed for training the system and the files generated after training process is completed.

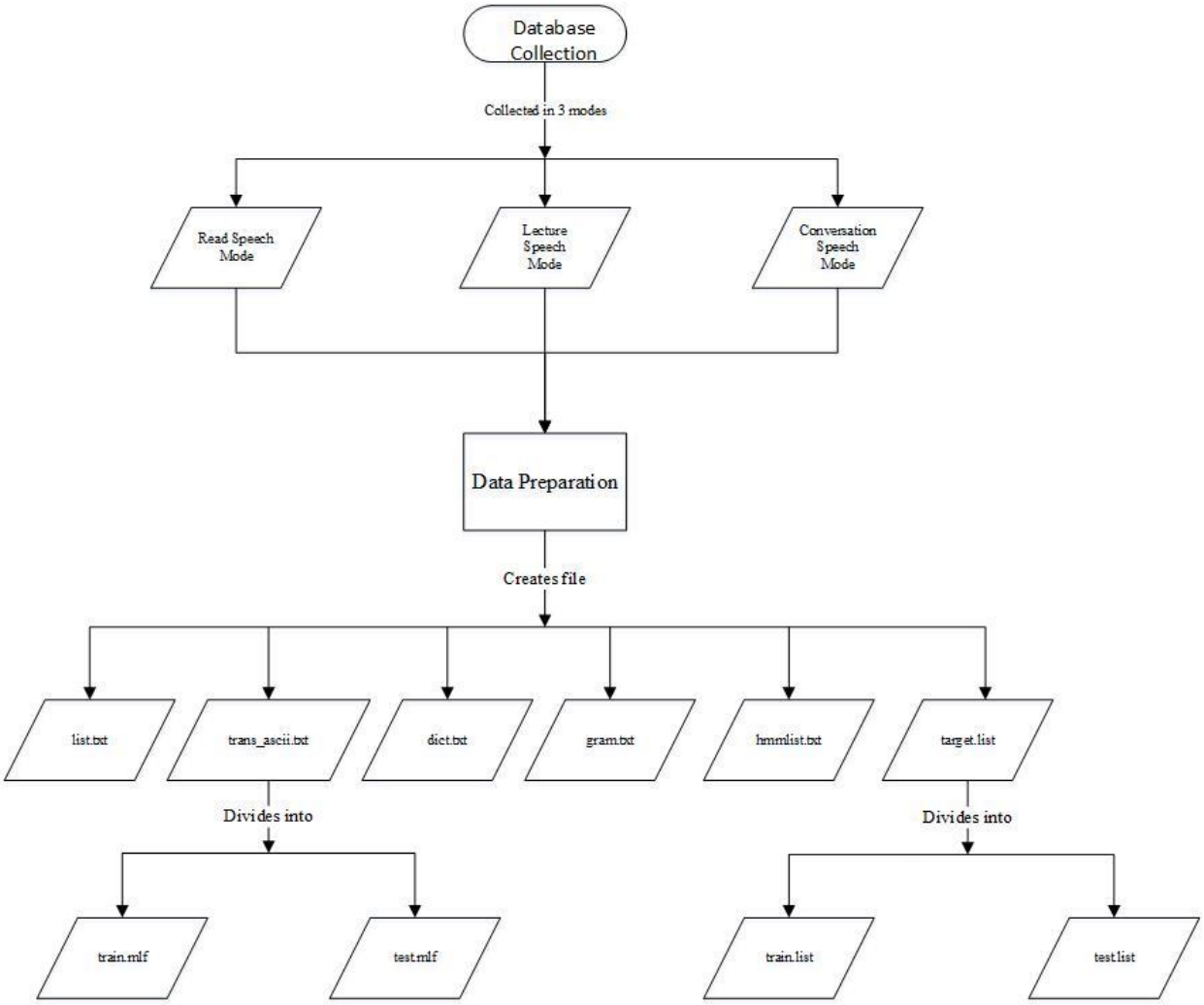


Figure 4.1: Data Preparation Phase

Following steps are performed in this phase:

1. Transcription

Transcription of read speech, lecture speech, and conversational speech has been done using International Phonetic Alphabet (IPA) chart. There are 36 IPA symbols including Vowels, Semi vowels, and Consonants. Apart from this, diacritics, tone and word accents, and suprasegmentals were also used in transcription.

Figure 4.2 depicts the transcription of a lecture speech mode. Wave Surfer, a standalone tool is used for cutting the speech wave files and also for its transcription. A particular segment of particular length is selected, then in transcription pane, transcription of that segment has been noted down using IPA chart.

2. Save whole transcription (IPA) in transcription_all.txt file and put all the speech wave files separately in a single folder.

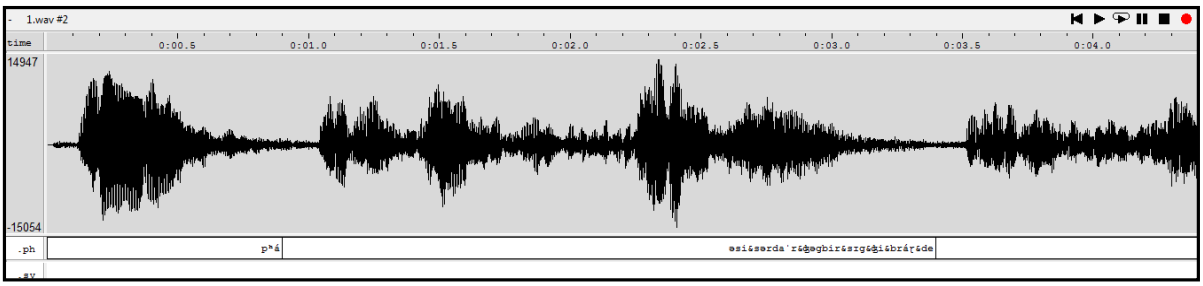


Figure 4.2: Transcription of a lecture speech file

Prepare a single transcription file at word level for each wav file such that it includes both wav file name and its corresponding transcription. These transcriptions demonstrates spoken utterances in the speech audio file. IPA symbols defined in transcription file need to be converted such that all IPA symbols get replaced with their corresponding ASCII characters. This is done so because HTK understands only the ASCII characters. This file does not differ for developing engine with 29 and 33 phone category. Figure 4.3 represents the transcription file that has been used in this work.

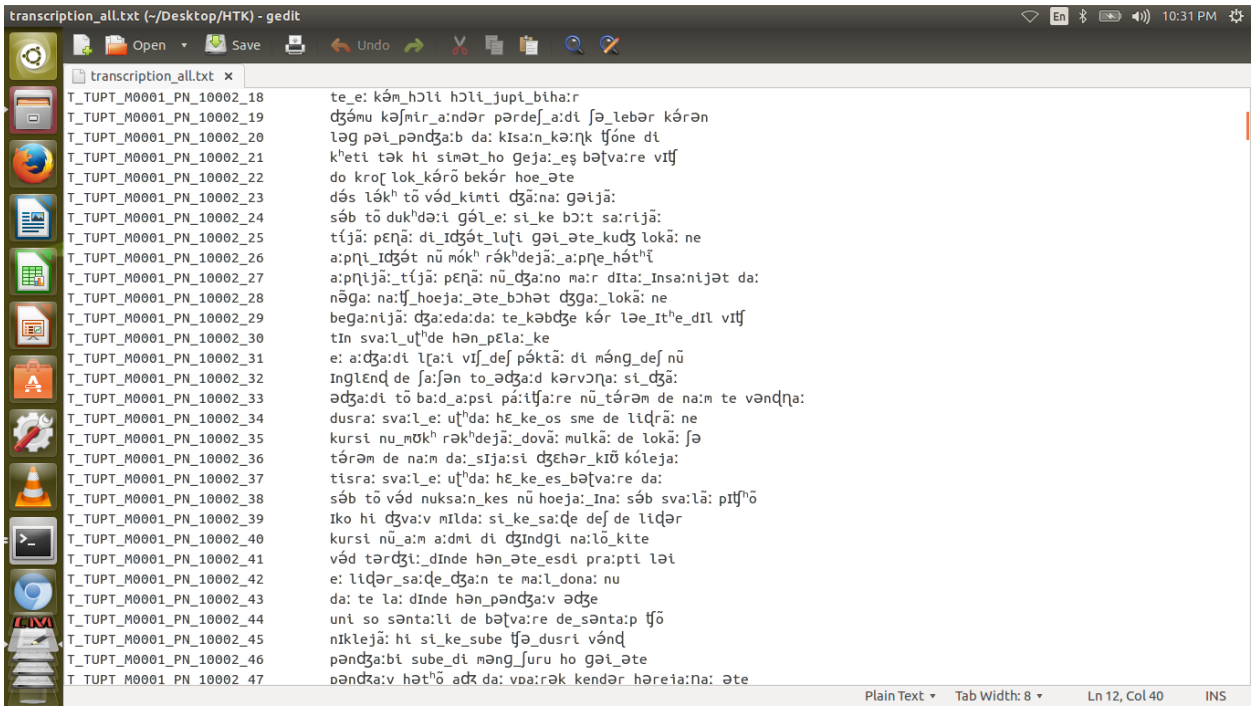


Figure 4.3: Sample transcription_all file

First column shows speech audio file names and second column shows their corresponding transcribed spoken utterances.

Some of the phonetic alphabets are substituted by the same ASCII character for getting the good accuracy as for a particular phone, many examples needs to be provided to HMM at the time of training and examples related to phonetic alphabets with diacritic were short. So, phonetic alphabets with diacritics are substituted with phonetic alphabets without diacritics resembling to same sound.

To develop the ASR for Punjabi with 33 phones, 4 new phones has been introduced in comparison to the proposed system with 29 phones. These phones are ‘ae’, ‘an’, ‘chh’, ‘oun’.

3. Create a new file list.txt in which all the IPA and their corresponding ASCII are written.
4. Install HTK in Ubuntu OS, for this just copy the binary files in usr/local/bin.
5. HTK takes input only in ASCII form not in IPA form, so next step is to convert IPA to ASCII *i.e.* need to convert all the transcription written in transcription_all.txt into ASCII. For IPA to ASCII conversion follow the following steps:
   i. Copy only the transcription part from transcription_all.txt file to another file *i.e.*trans_all.txt.
   ii. At the beginning of transcription of each wav file introduce ‘%’ symbol and at the end of each transcription introduce ‘@’ symbol.
   iii. Introduce single space after each character using command on terminal:-
   sed 's/./& /g' trans_all.txt> trans_space.txt
   iv. Introduce file name of their corresponding transcription in trans-space.txt.
   v. Replace ‘%’ symbol with .lab.
   vi. Run convert.sh script on terminal. This script will convert all IPA in trans_space.txt to their corresponding ASCII as given in list.txt and save it another file trans_ascii.txt.
   vii. Now find $ in trans_ascii.txt and replace it with /n and @ with /n.

Figure 4.4, depicts the ASCII value corresponding to each IPA.

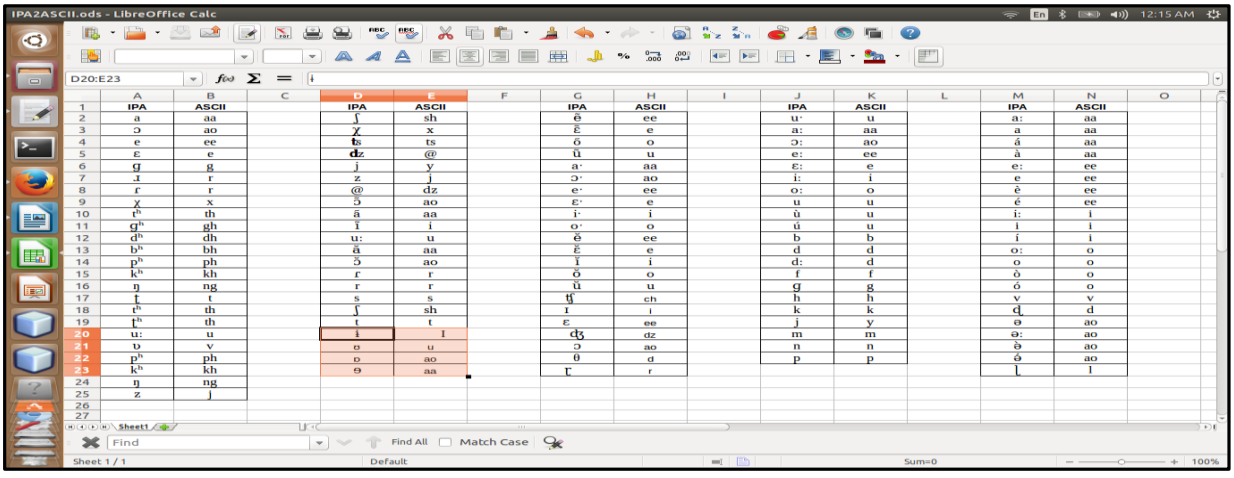


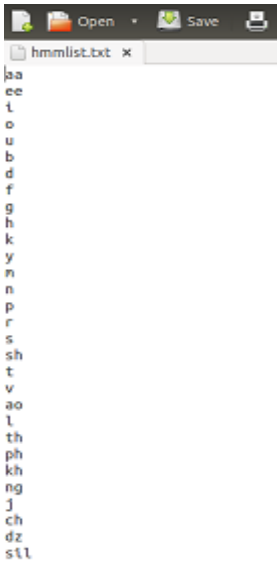
Figure 4.4: IPA to ASCII

6. Now check the file for all symbols, if any symbol not converted then convert it manually.
7. Now need to create two mlf files. For this:
   i. Introduce #!MLF!# at the beginning in trans_ascii.txt and rename it as train.mlf. This file is for training purpose.

ii. Now create another file test.mlf and cut the last (say 25% of the total files) from train.mlf and save it in test.mlf with #!MLF!# at the beginning. This file is for testing purpose.

Thus transcription of all wav files are converted into ASCII form to be given as input to HTK toolkit.

8. Copy all the ASCII symbols in two columns and name the file as dict.txt.
9. Write all the symbols in a separate file and symbols should be in concatenated form and name the file as gram.txt.
10. Now copy all the unique ASCII codes in single row and save it as hmmlist.txt.
11. Now create a file target.list which contains the path of all wave (training + testing) files and correspondingly path of .mfc files that are to be generated.
12. Now copy the path of .mfc files for only those wave files which are listed in train.mlf and name the file as train.list.
13. Now copy the path of .mfc files for only those wave files which are listed in test.mlf and name the file as test.list.
14. List of Models (HMM List)



(a)

(b)

Figure 4.5: HMM list for (a) 29 phones (b) 33 phones

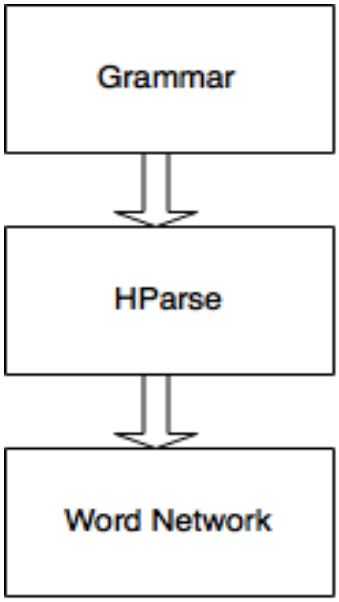


Figure 4.6: Working of HParse

Here a list of all those phonemes whose HMM models are to be built, is prepared both for 29 and 33 phones (excluding Silence) category. Both the lists has been prepared separately to develop the proposed system. Figure 4.5 (a) depicts a list of 29 different phonemes and figure 4.5 (b) depicts a list of 33 different phones that have been used to build HMM models.

In HMM list, every defined HMM has two different types of name: physical and logical name. The physical name identifies definition on disk and the logical name expresses the role of the model. Both physical and logical names are identical by default.

15. Grammar

HTK basically requires a word network to get each word to word transition and each word instance. For this, a grammar definition language is supported by the HTK for specifying the simple task grammar. This grammar is processed by HTK and HTK creates a word network for itself. Grammar consists of some variable definitions followed by regular expressions. For processing this grammar, HTK uses its HParse tool. Figure 4.6 shows the working of HParse tool. This tool takes as input the 'grammar' and using this input it creates a word network. Figure 4.7 (a) displays the grammar for engine to be built with 29 phones (excluding silence) and figure 4.7 (b) displays the grammar for engine with 33 phones (excluding silence).

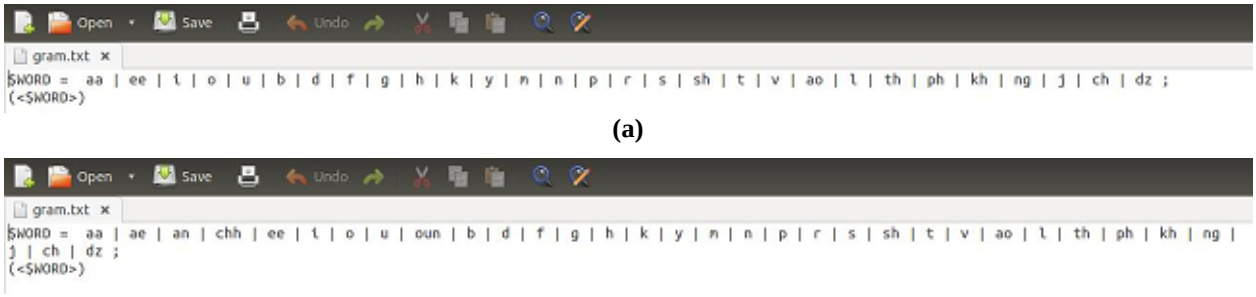


Figure 4.7: Grammar for (a) 29 phones (b) 33 phones

Where, vertical bars (*i.e.* |) specify the alternatives and angle braces (*i.e.* < >) specifies one or many reoccurrences. This full grammar is converted by HTK into a network.

If talk about word level speech recognition, instead of HTK Julius toolkit is used, which uses the acoustic model created by HTK toolkit along with its own grammar definition. In phone level recognition system grammar, it will recognize words only by listening a sequence of phones. For example, if we want to recognize 'vaakh' it will actually listening to sequence of phones as “v”, “aa”, “kh”. Opposite to HTK acoustic model, recognition grammar of julius is divided into two parts:

- word.grammar
- word.voca

word.grammar:- This file defines a bunch of rules which are used for determine the words that are expected to be recognise. Here, instead of listing out all the words, only word categories are defined.

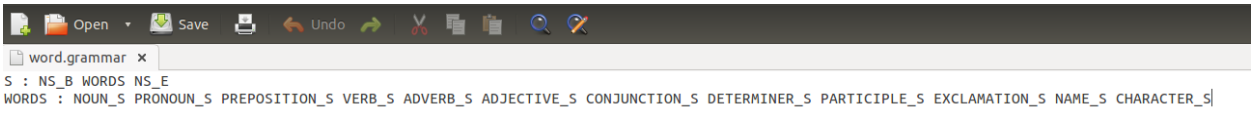


Figure 4.8: word.grammar file

Figure 4.8 shows the grammar file used by Julius. Here only the word definitions are mentioned in this file whose corresponding words are defined in word.voca file.

word.voca:- This file defines the words under each and every word category along with their pronunciation. Figure 4.9 represents the vocabulary file used for word level recognition. Here, in this file each word is represented under specific word category and their pronunciation is also mentioned here.

After these two files are created now the corresponding dictionary file need to be generated. For this the Julius script mkdict.jl is run in the terminal using the command:

julia mkdict.jl word

Here in this command Julia represents the name of Julia programming language, mkdict.jl is the Julian script which is used to create a dictionary file which gives the information about the dictionary created for this work and word is the prefix of dictionary file which will be created.

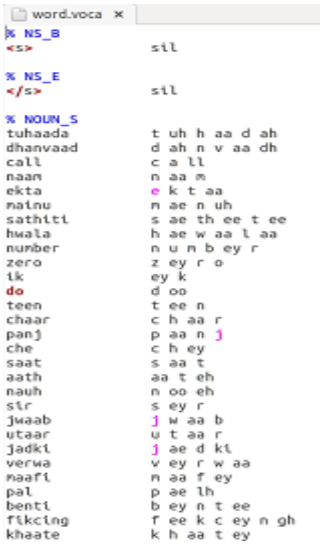


Figure 4.9: Sample word.voca file

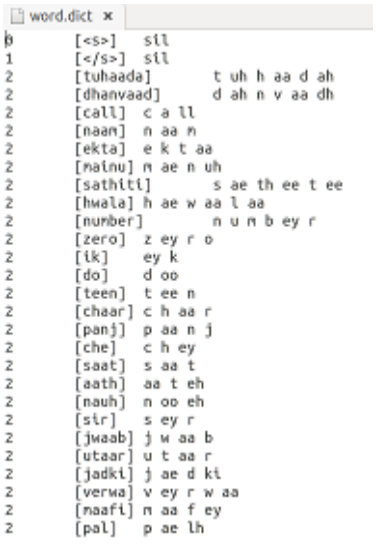


Figure 4.10: Sample word.dict file

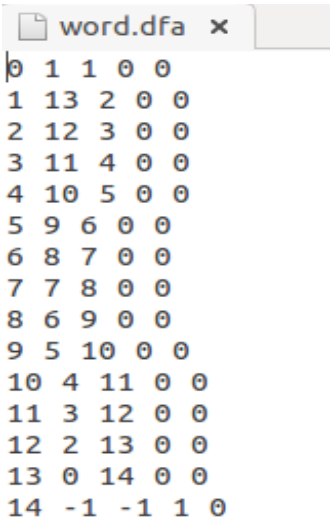


Figure 4.11: word.dfa file

After successful execution of this command 3 different files are generated namely: word.dict, word.dfa and word.term. Out of these three files first one defines the dictionary information and the latter two files define the information about finite automata of the system.

Figure 4.10 represents the dictionary file generated after executing mkdict.jl. This file tells about the phone sequence for each word.

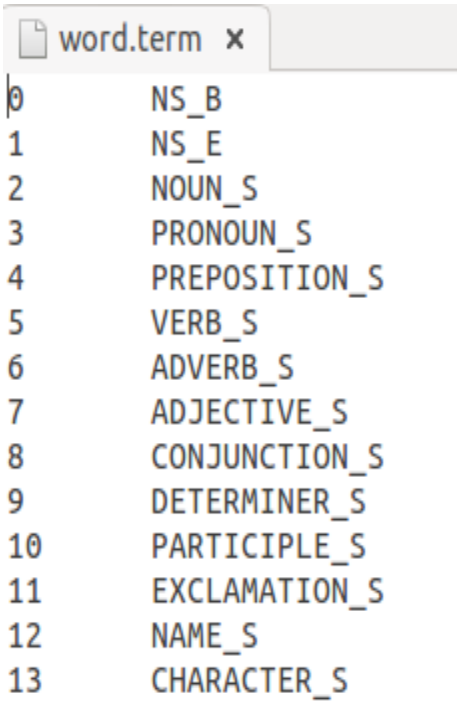
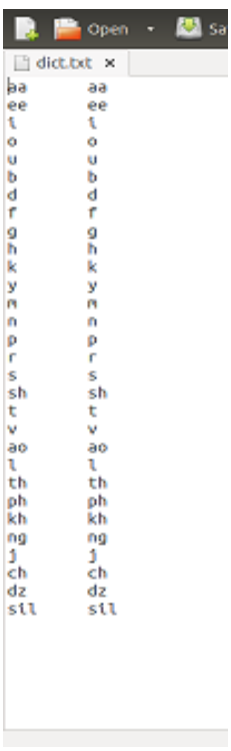


Figure 4.12: word.term file



(a)

(b)

Figure 4.13: Pronunciation Dictionary for (a) 29 phones (b) 33 phones

Figure 4.11 represents the *dfa* of the system generated after executing the script and figure 4.12 represents the term file generated after completion of execution of script. Both these files gives information about the finite automata of the system.

16. Pronunciation Dictionary

Pronunciation dictionary specifies the words pronunciations as the linear sequence of phonemes. Since, we are working at phoneme level, a file consisting of all phonemes with their corresponding pronunciation is prepared. If we work on word level then this file will consist of word and its corresponding pronunciation. It can be built easily from the sentences collected as samples, which are further prompted in the training data set. For example, if we talk about at word level, it will look like this:-

BAT b a t

BALM b a m

and so on. The pronunciation is not case sensitive. If multiple pronunciations exist for a single word then there will be repeated entry for the word. For example:-

vakh v aa kh

vakh w aa kh

It should be noted that no blank line should be left after any line in the dictionary. At phoneme level, pronunciation dictionary looks like as shown in figure 4.13 (a) and figure 4.13 (b).

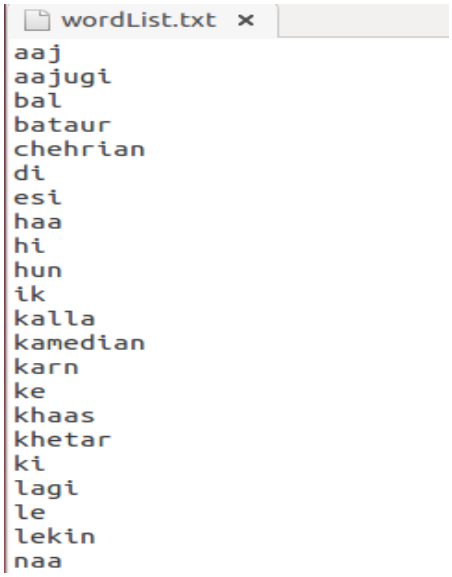


Figure 4.14: Sample wordlist.txt file

Figure 4.15: Sample pronunciation dictionary of word level

First column shows the phoneme (and at word level it will be a word) and second column shows the pronunciation of corresponding phoneme. This file will be different for 29 and 33 phone category. Figure 4.13 (a) represents the Pronunciation Dictionary for 29 phones whereas Figure 4.13 (b) represents the Pronunciation Dictionary for 33 phones.

If we talk about word level, first of all the transcription file is converted into list of words file say wordlist.txt. Further this file is used for creating pronunciation dictionary for word level. Figure 4.14 shows the wordlist file created for word level data.

Significance of using pronunciation dictionary is that when a speaker speaks out a word that need to be recognized, recognizer will listen the distinct sounds and looks for matching HMMs of each sound and then, determine the sequence of phones that make up a particular word based on training given to it and then these sequence of phonemes, *i.e.,* the pronunciation, are checked in dictionary, if entry exist then, the word mentioned against it, is picked up. The role of pronunciation dictionary is same at phoneme level. Figure 4.15represents the pronunciation dictionary for word level data using triphone models.

CHAPTER 5

Data Training and Testing of ASR for Punjabi Language

5.1 Data Training for phone level

In second step of system development, prototype definition is written for each HMM, so that the required topology for each HMM can be defined. The main objective of defining a prototype is to enumerate the overall characteristics and topology of HMM**.** Figure 5.1 depicts the overall training process of the system, *i.e.* the files needed for the training and the file came out as output after training, all can be depicted through the figure.

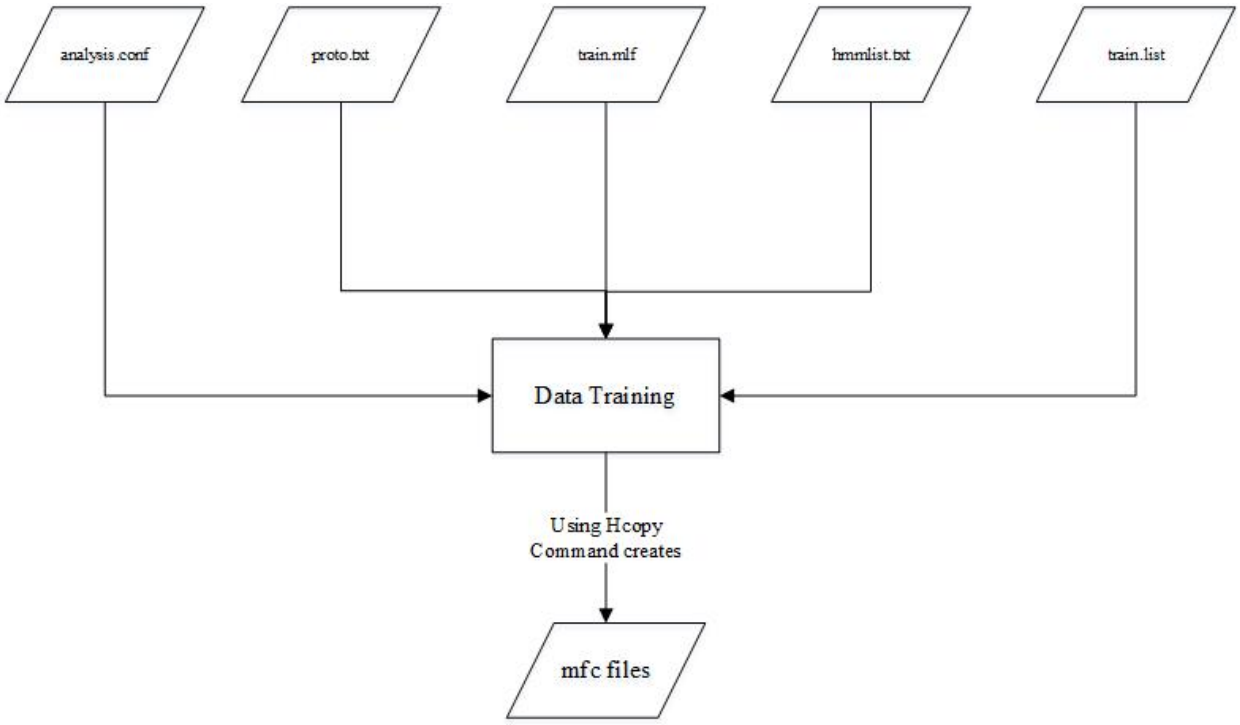


Figure 5.1: Data Training Phase

The actual training process takes place in stages:

1. Components required for phone level training:-
   i. Training file (say, train.mlf), figure 5.2 represents the train.mlf file needed for training.
   ii. Figure 5.4 represents a file consisting of path of MFCC features of all wav files used for training (say, train.list).
   iii. List of Models (say, hmmlist.txt).

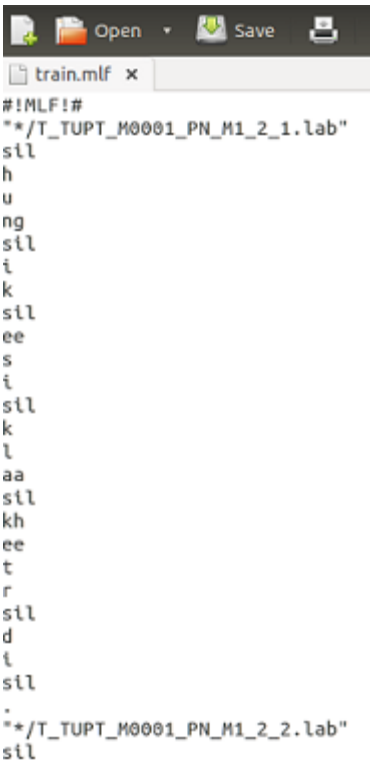


Figure 5.2: Sample train.mlf file for training data

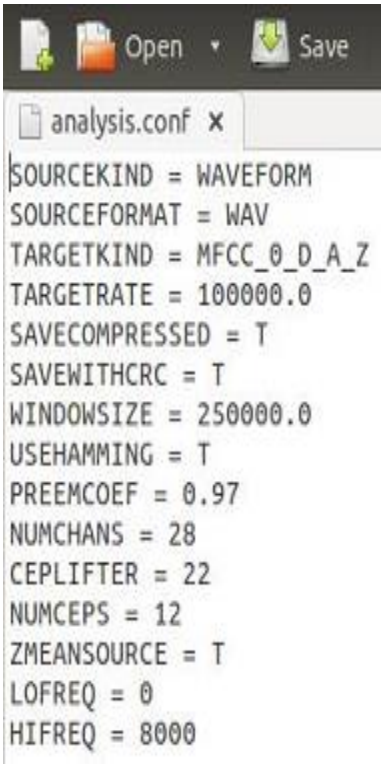


Figure 5.3: analysis.conf file

iv. Figure 5.3 depicts the configuration file (say, analysis.conf), needed for training process. This file will tell about various basic features like source kind, source format, target kind *etc.*

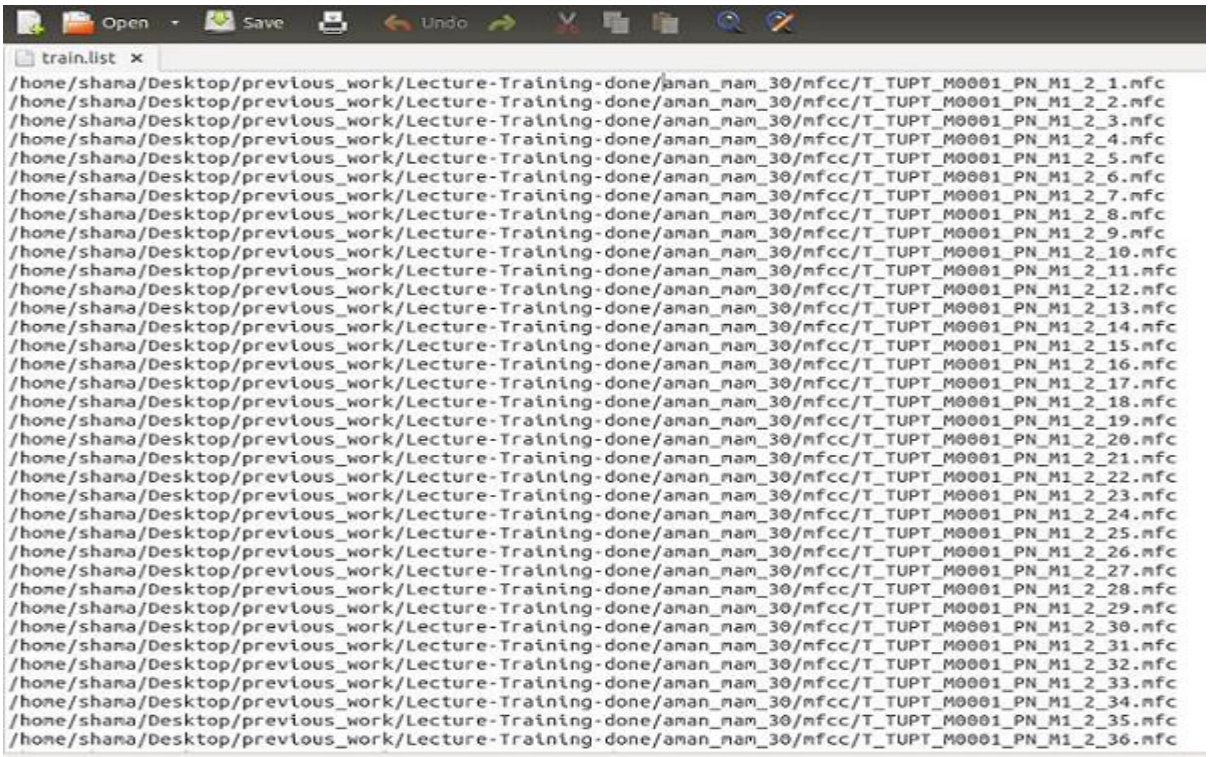


Figure 5.4: Sample train.list file

v. Figure 5.5, shows the Prototype Model (say, proto.txt) which indicated about the number of states and their means.

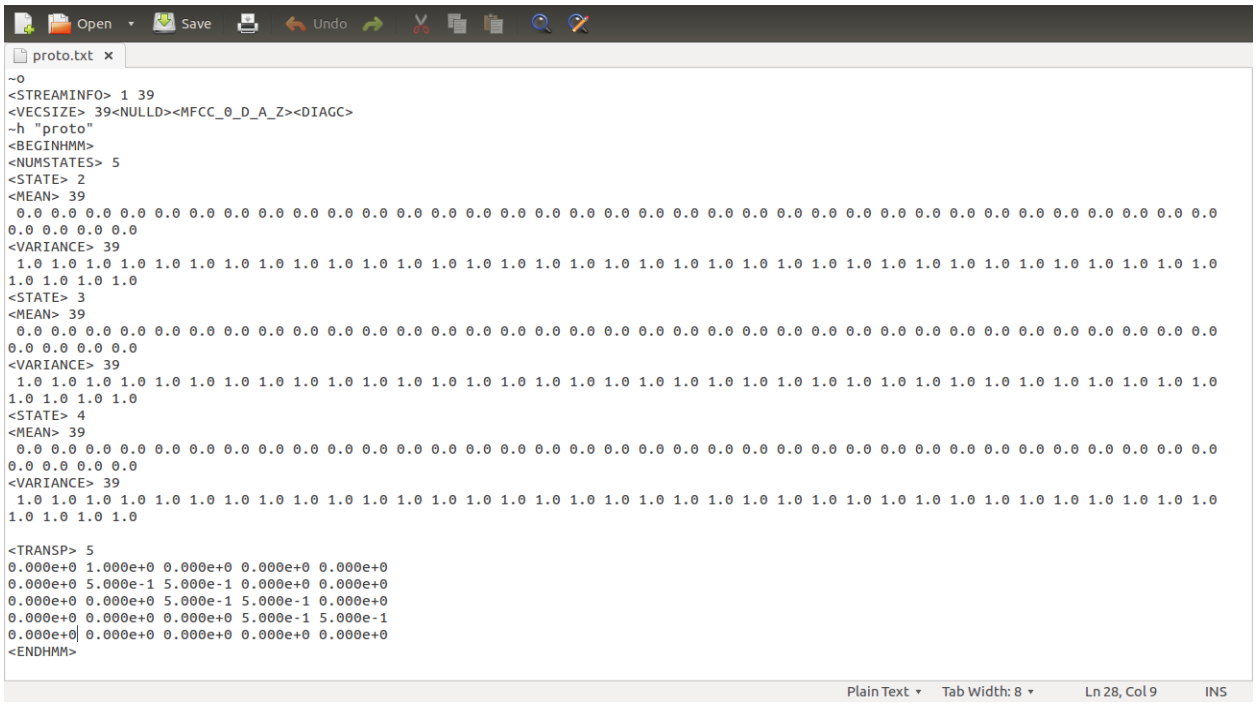


Figure 5.5: proto.txt file

vi. Mean and Variance Model (say, bashscript.sh)

2. Create .mfc files using target.list with the help of HCopy command on terminal as:-

HCopy -T 1 -C analysis.conf -S target.list

3. Now train.mlf and train.list will be given as input to htk and run bashscript.sh on terminal. This will train the system.
4. At the time of training the system, for each ASCII character 32 mixtures will generate from hmm0 to hmm32 for 30 phones (including silence) and 36 mixtures will generate from hmm0 to hmm36 for 34 phones (including silence).
5. These mixtures are generated using proto.txt.

After performing all these steps training for phone level data will be completed.

5.2 Data Training for word level

1. Components required for word level training:-
   i. A file consisting of path of MFCC features of all wav files used for training (say, train.list).
   ii. List of Models (say, hmmlist.txt).
   iii. Configuration file (say, analysis.conf).
   iv. Prototype Model (say, proto.txt).

v. Mean and Variance Model (say, hmmdefs and macros).

2. Create .mfc files using target.list with the help of HCopy command on terminal as:-

   HCopy -T 1 -C analysis.conf -S target.list

3. transcription_all.txt file is given to the transcription2mlf.jl script, to generate word.mlf file. word.mlf can be created by executing following command on terminal:

   julia transcription2mlf.jl transcription_all.txt word.mlf

word.mlf

```
#!MLF!#
"T_TUPT_M0001_PN_M1_2_1.lab"
hun
ik
esi
kalla
khetar
di
.
"T_TUPT_M0001_PN_M1_2_2.lab"
shaksiat
tuhade
rubru
karn
lagi
haa
tuhade
chehrian
.
"T_TUPT_M0001_PN_M1_2_3.lab"
te
ohda
```

Figure 5.6: Sample word.mlf file

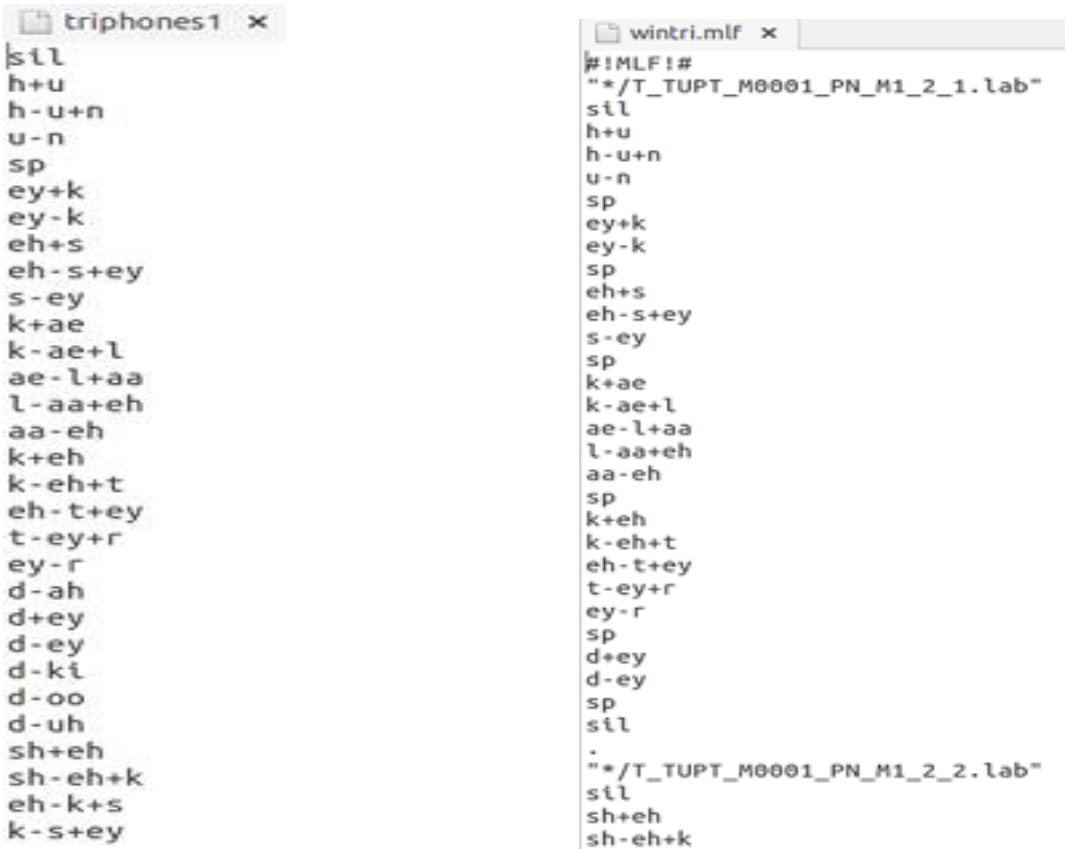
tiedlist

```
t+ae
b
d
g
h
t+au t+ae
j
k
l
m
n
p
r
s
t t+ae
u
t+ee t+ae
t+ey t+ae
t+ou t+ae
t+uh t+ae
ey-w+aa
j-w+aa
t-aa
t-ee
t-eh
t-ey
d-eh+t t-eh
sh-eh+k t-eh
t-uh
aa t-aa
```

Figure 5.7: Sample tiedlist file

This word.mlf file contains the transcription of each wave file but as oppose to transcription_all.txt one word per line. Figure 5.6 represents the sample for word.mlf file.

4. Now, this word.mlf is used to create train.mlf file.
5. At the time of training the system, for each ASCII character 10 mixtures will generate from hmm0 to hmm9. These mixtures are generated using proto.txt. and hmmlist
6. Now, train.mlf along with mktri.led (script of Julia language) is used to create triphone list (say, triphones1) and a mlf file containing these triphones instead of single phones. Figure 5.8 (a) shows the triphone list and figure 5.8 (b) represents the mlf file containing these triphones corresponding to each word.



(a) (b)

Figure 5.8: (a) Sample triphones list (b) Sample wintri.mlf file

7. For each ASCII character 6 new mixtures will generate from hmm10 to hmm15 using triphone and wintri,mlf. These mixtures are for training the system for word level recognition.
8. The triphone list and wintri.mlf are also used for generating dictionary for word level and tiedlidst (used further for testing). Figure 5.7 shows the tiedlist used in testing of word level data.

After performing all these steps training for word level data will be completed.

5.3 Data Testing for phone level

In this phase testing of the system is done. From this step a corresponding mlf file is generated which is then compared with the mlf file of training data. Figure 5.9, illustrates the overall testing procedure of the system.

1. Components required for phone level testing are:-
   i. List of Models (say, hmmlist.txt).
   ii. Pronunciation dictionary (say, dict.txt).
   iii. Configuration file (analysis_train.conf).

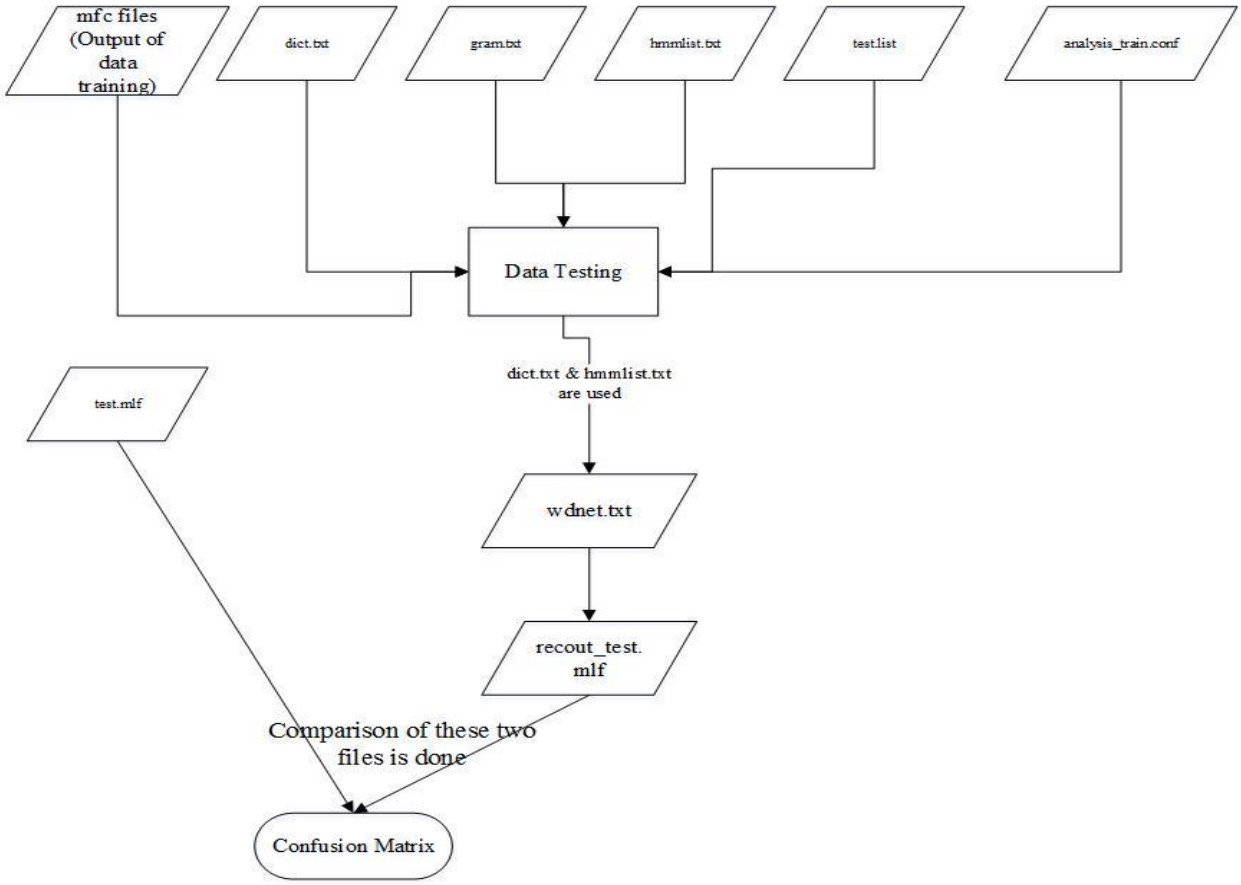


Figure 5.9: Data Testing Phase

iv. A file consisting of path of MFCC features of wav files that are to be recognized (say, test.list). Figure 5.10, represents the test.list file which contains the destination path for mfcc files in it.

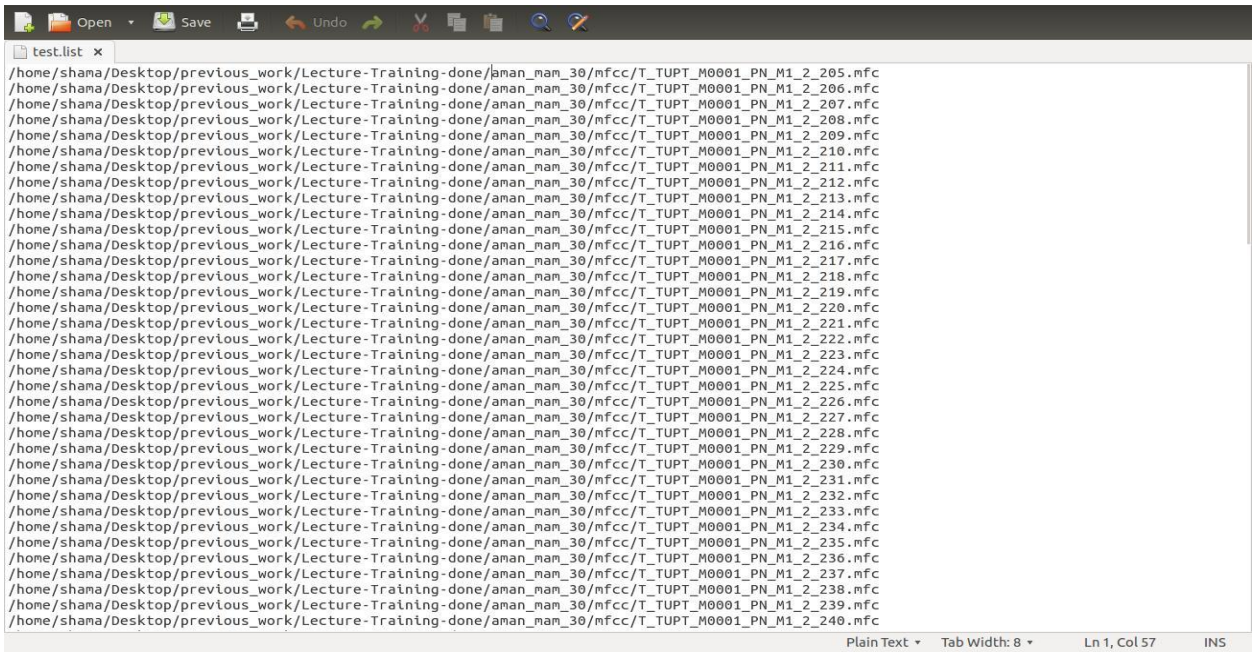


Figure 5.10: Sample test.list file

v. Grammar (say, gram.txt).
vi. Master Macro File in which 32 mixtures of each state of each HMM model are computed (say, hmmdef.txt).

2. Now run the testscript.sh on test.list using wdnet.txt (include all the grammar rules which is to be applied at the time of testing).
3. Wdnet.txt is created using dict.txt and hmmlist.txt.
4. At the time of running testscript.sh recout_test.mlf file will be created and this file will be compared with test.mlf and accordingly a confusion matix will be generated.

After all these steps testing for phone level data is complete.

5.4 Data Testing for word level

1. Components required for word level testing are:-
   i. List of Models (say, tiedlist).
   ii. Pronunciation dictionary (say, tri_dict.txt).
   iii. Configuration file (analysis_train.conf).
   iv. Testing file (say test.mlf)
   v. A file consisting of path of MFCC features of wav files that are to be recognized (say, test.list).
   vi. Grammar (say, gram.txt).
   vii. Master Macro File in which mixtures of each state of each HMM model are computed (say, hmmdef and macros).
2. Wdnet.txt is created using tri_dict.txt and tiedlist.
3. Now these files are used and with the help of HVite command recout_test.mlf is created.
4. At the time of testing, HResults command is used to compare recout_test.mlf (system generated) file with test.mlf (manual) and accordingly results are displayed on the terminal in the form of % accuracy and %correctness.

After all these steps testing for word level data is complete.

CHAPTER 6

Graphical User Interface of ASR

The system is trained with three different modes of speech. The whole work done in the implementation of the system that is, training and testing has been shown through the graphical user interface implemented for the system. In the GUI, starting from the audio recording, the speech has been recorded in the particular data set which is chosen. After that, mfc of that file is created, means the file has been trained. Now the path of the file has been set and recognized through the GUI itself. After that testing will be performed on the system. Once testing is completed, the result can be shown from the result button.

Figure 6.1: Home Page of the GUI

Figure 6.2: Selecting a particular mode

Figure 6.1 shows the home page generated for the system. Here, in the combo box list of all the data sets created for training and testing the system are shown. Among the list any of the data has can be chosen for storing and processing accordingly.

In figure 6.2, it is clearly mentioned that, out of all the modes, the read speech mode for male1 with 30 phones scheme is chosen to be processed further. After selecting the path, record button is pressed. As soon as, record button is pressed it will take us to the recording page.

Figure 6.3 shows that, after clicking the record button we reaches to recording screen. Here, by clicking on start button recording of the speech starts. And the recorded sound is in wave form, therefore try to record data for 4-5 seconds.

Figure 6.4, shows the image, which comes while recording is in process. Then the stop button enabled, so that the recording can be stopped whenever the person stops. Once the stop button is pressed, the

recording is stopped and we reached to the home page again but this time, with the path of the sound wave.

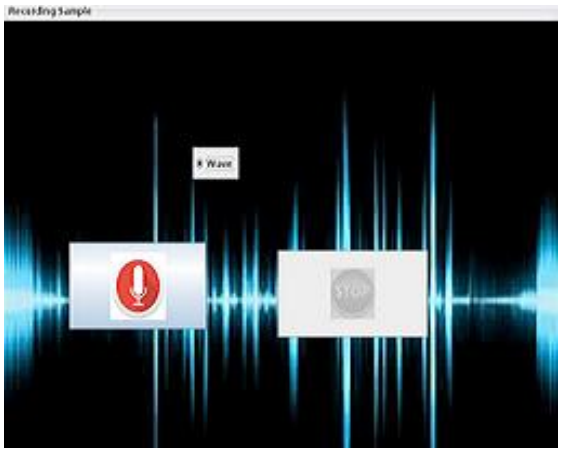


Figure 6.3: Start recording the audio

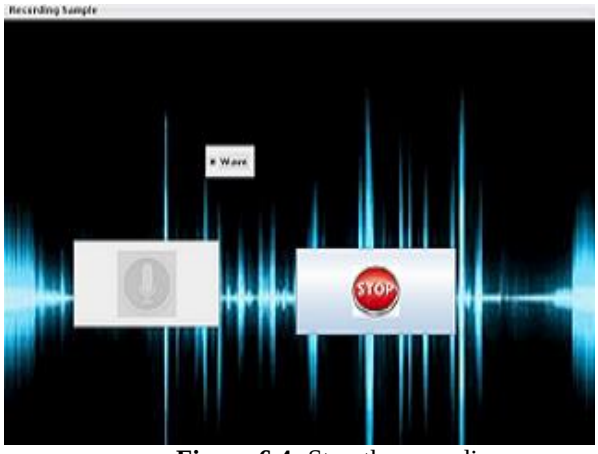


Figure 6.4: Stop the recording

Figure 6.5, shows that the process button is enabled which means recording is done and now we have to process the speech. Here, path of the recorded file has also been present from the recording window. As soon as, the process button is pressed, the audio file has been processed for training and testing and the window changes too.

In figure 6.6, we can clearly see that the path of audio file has been written in the text boxes above. This path has been came from previous work, not manually. One path is for wav file and another one is for mfcc file. First step is to set these paths. It means store two different files in these two different locations.

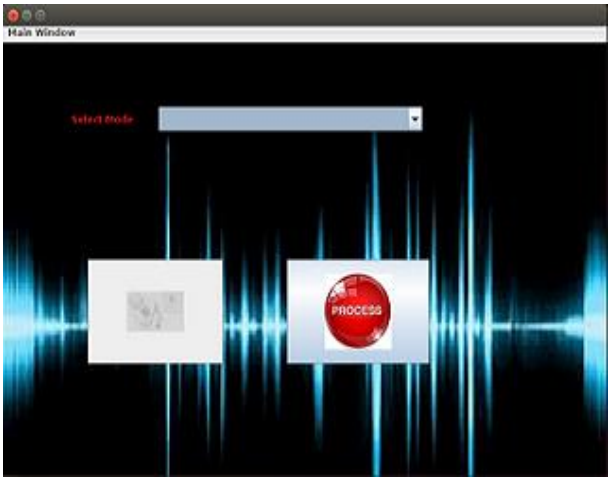


Figure 6.5: Processing the speech

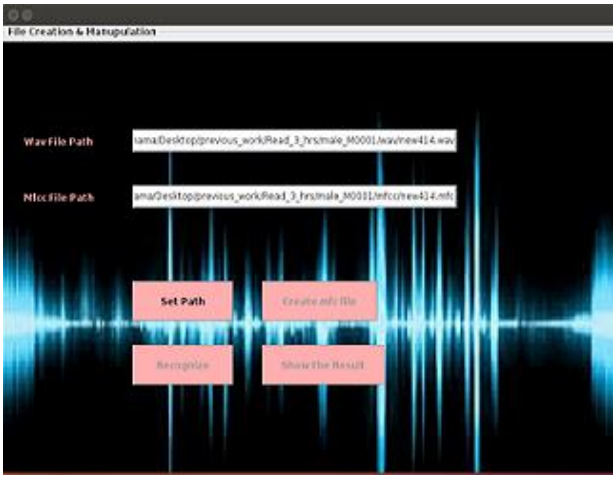


Figure 6.6: Path setting of recorded audio file

In figure 6.7, it is clearly mentioned that the corresponding mfc file of the audio file has been created in the location specified earlier. For creating mfc HCopy command is used.

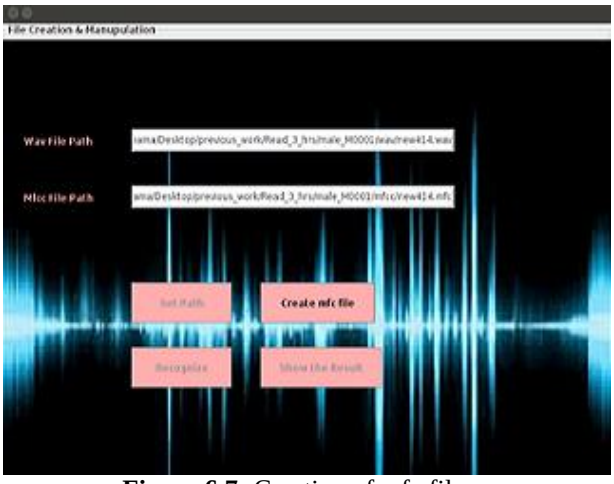


Figure 6.7: Creation of mfc files

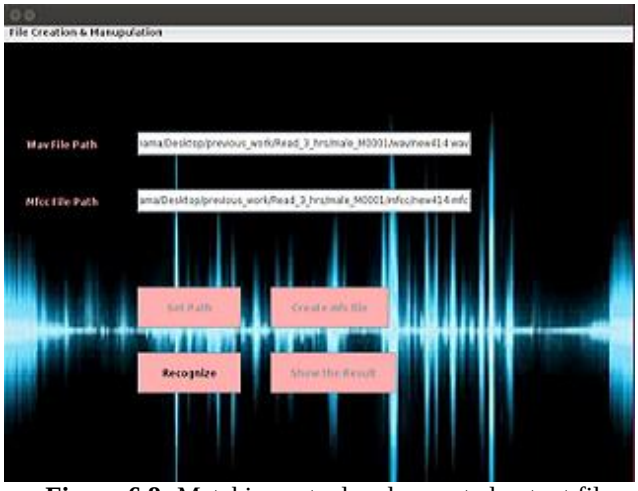


Figure 6.8: Matching actual and expected output file

In figure 6.8, the recognition of the system is done. It means the actual output file that has been created is compared with the expected output file and then a confusion is matrix is created after this comparison.

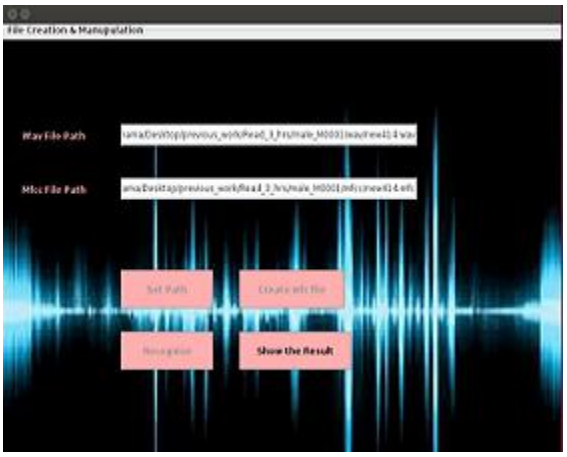


Figure 6.9: The result window

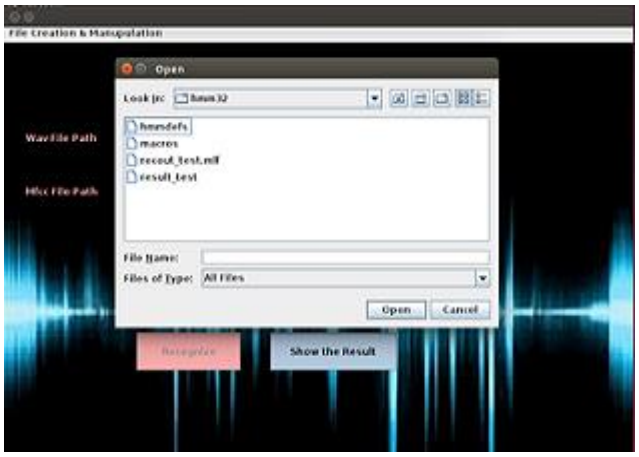


Figure 6.10: Choose the result file

In figure 6.9, it can be shown that the show the result button is enabled. Thus we can click on this button to get the results. It will take us to the window of the last hmm of the mode selected.

In figure 6.10, we have to choose the result_test file among all the other files. When we open this file, it will show us confusion matrix as a result.

CHAPTER 7

Results of Proposed ASR System

The proposed ASR system has been trained and tested for three different modes of speech:

- Read Speech Mode
- Lecture Speech Mode
- Conversational Speech Mode

Each mode is collected differently with different duration of data. The total duration of data considered for each and every mode is given in table 7.1.

Table 7.1: Total Duration of data for each Mode

Mode	Total Duration (in Minutes)
Read Speech Mode	226.33
Lecture Speech Mode	136.98
Conversational Speech Mode	35.45

For different modes of speech, different test cases have been generated which depends upon the nature of the speech data collected.

In all the modes, total speech set has been partitioned into two parts: speech data set for training and speech data set for testing. For training, 75% of the total data set has been reserved and the rest 25% of the total data set is used for testing. Once the recognition is process is over, as a result a confusion matrix is generated at the end of testing phase. This confusion matrix is a resultant of comparison of actual results of the transcription with the expected results of it. As each kind of speech has been divided into different categories, thus, for each and every case different confusion matrix has been generated separately. This confusion matrix tells about the overall correctness and accuracy of the system. Along with this, the confusion matrix also shows some related information too, like, total number of phones used, total number of phones substituted in testing phase, *etc.* In confusion matrix, the total number of rows shows the unique phonemes used for developing the system including silence and, and total number of columns shows the unique phonemes used excluding silence. Each row represents the instances in actual class and each class represents the instances in predicted class. All the correct matches of the phonemes are shown diagonally in the confusion matrix.

The percentage accuracy of the trained system can be recognized by equation (3):

$$\boldsymbol{N = H + D + S} \quad \textbf{(1)}$$

$$\boldsymbol{H = N - D - S} \quad \textbf{(2)}$$

$$\% \, \boldsymbol{Accuracy} = \frac{H-I}{N} \times 100 \qquad (3)$$

Where, N is total no. of phones, H is correctly detected phones, D is the number of deleted phones, I is the number of inserted phones *i.e.* external insertion of phones, and S is the substituted phones.

To compute the percentage of phones correctly recognized equation no. (4) is used:

$$\% \, \boldsymbol{Correctness} = \frac{H}{N} \times 100 \qquad (4)$$

In case of read speech mode, the system has been trained as well as tested for each gender and speaker. In lecture speech mode, system is trained and tested with different speakers and different transcribers. In conversational speech mode, system is trained and tested with conversational data.

7.1 Performance Evaluation for Read Speech Mode

The results for testing accuracy of the system in read speech mode are given in this section.

The results are represented in the form of confusion matrix. The confusion matrix generated for read speech mode with 29 phones is similar as shown in figure 7.1 for read speech mode with 33 phones. Here, in this case, *i.e.* read speech mode with 29 phones, correctness percentage is 68.19% and accuracy percentage is 61.84%. Here, total number of substituted phones (S) are 5019, total number of correctly recognized phones (H) are 10269, number of deleted phones (D) are 7536, number of inserted phones (I) are 809, and total number of the phones (N) are 22824.

Figure 7.1: Confusion matrix for Read Speech Mode total data with 33 phones

Figure 7.1 represents the confusion matrix for read speech mode having 33 phones. Here, each cell is representing the number of times an actual class being predicted as another class. For example, in row 1 label 'aa' is predicted as 'ee' fourteen times, as 'i' three times and so on. First line of the result shows the accuracy of sentence level which is depends on complete number of label files being indistinguishable to the transcription file, which is 0.00% in this case. The second line of the result shows the word accuracy of the system, which depends upon matches found between the transcription and label files. Here, in this case, *i.e.* read speech mode with 33 phones, correctness percentage is 75.26% and accuracy percentage is 69.95%. Total number of substituted phones (S) are 5395, total number of correctly recognized phones (H) are 15015, number of deleted phones (D) are 4372, number of inserted phones (I) are 1539, and total number of the phones (N) are 24782. '%c' shows the number of times a phone has been correctly labeled, *i.e.*, percentage correctness in the row (number of faultless labeled items divide by the total number of items in the row) and '%e' shows the counting of incorrectly labeled phones in a row as a percentage of total number of phones.

Data of Read Speech Mode has also be trained for both male as well as female voice. This process has been carried out for training and testing the system both for, 29 phones and 33 phones.

The accuracy of the system for total males of read speech mode with 29 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. Here, correctness percentage is 72.29% and accuracy percentage is 61.92%. Total number of substituted phones (S) are 4321, total number of correctly recognized phones (H) are 11945, number of deleted phones (D) are 2814, number of inserted phones (I) are 1981, and total number of the phones (N) are 19080.

The accuracy of the system for total males of read speech mode with 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. Here, correctness percentage is 79.45% and accuracy percentage is 68.80%. Total number of substituted phones (S) are 4352, total number of correctly recognized phones (H) are 12945, number of deleted phones (D) are 2854, number of inserted phones (I) are 1881, and total number of the phones (N) are 19090.

The accuracy of the system for total females of read speech mode with 29 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. Here, correctness percentage is 59.98% and accuracy percentage is 52.18%. Total number of substituted phones (S) are 1488, total number of correctly recognized phones (H) are 2494, number of deleted phones (D) are 1375, number of inserted phones (I) are 288, and total number of the phones (N) are 5155.

The accuracy of the system for total females of read speech mode with 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. Here, correctness percentage is 66.72% and accuracy percentage is 60.40%. Total number of substituted phones (S) are 1388, total number of correctly recognized phones (H) are 2484, number of deleted phones (D) are 1305, number of inserted phones (I) are 268, and total number of the phones (N) are 5177.

Table 7.2 displays the complete details of experiments that are carried out with the read speech data for 30 phones as well as 34 phones. The system has been trained as well as tested with total data and for both genders, male and female.

Table 7.2 Testing accuracy of system for read speech mode with 29 and 33 phones

Sr. No.	Gender (Male /Female)	No. of Speakers	No. of phones	Training Data (in Minutes)	Testing Data (in Minutes)	Accuracy (in %)	Correctness (in %)
1	Both	4	29	138.20	48.13	61.84	68.19
2	Both	4	33	169.20	68.13	69.95	75.26
3	Male	2	29	122.41	38.65	61.92	72.29
4	Male	2	33	137.05	47.01	68.80	79.45
5	Female	2	29	18.48	6.35	52.18	59.98
6	Female	2	33	30.50	11.33	60.40	66.72

The Comparative analysis of the experiments performed on read speech data with 29 phones as well as 33 phones are depicted through the following figures. The analysis for each gender is also depicted.

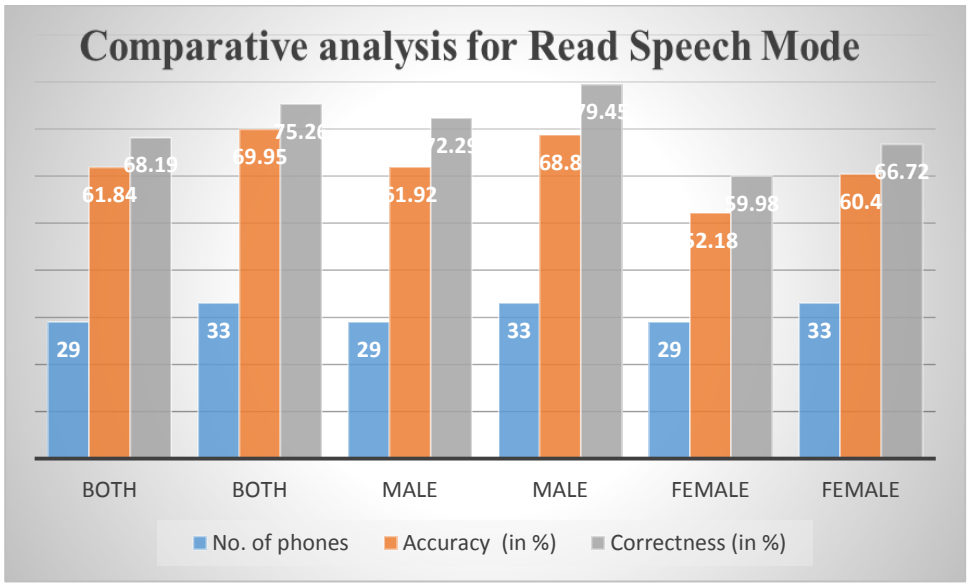


Figure 7.2: Comparative Analysis of system for Read Speech Mode with 29 and 33 phones

Figure 7.2 clearly displays the comparative analysis for the read speech mode with 29 phones and 33 phones corresponding to all the genders.

The system has also been trained for each individual person of read speech data. This process has also been carried out with 29 as well as 33 phones. In case of female speakers, first of all total duration of female data is divided into each female speaker, then 75% data of each individual's data is used for training the system and rest 25% is used for testing the system.

The accuracy of the system for total female_01 of read speech mode with 29 phones and 33 phones is represented individually in the form of confusion matrix in the same way as shown in figure 7.1. In case of 29 phones, correctness percentage is 55.70% and accuracy percentage is 48.78% and 61.24% correctness and 57.30 % accuracy with 33 phones.

The accuracy of the system for total female_02 of read speech mode with 29 and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. In case of 29 phones, correctness percentage is 64.72% and accuracy percentage is 56.77% and 70.84% correctness and 63.80% accuracy with 33 phones.

Table 7.3 shows the accuracy and correctness of the system for each individual female speaker of read mode with 30 and 34 phones.

Table 7.3: Testing accuracy of system for each female of read speech mode with 29 and 33 phones

Sr. No.	Speaker	No. of phones	Training Data (in Minutes)	Testing Data (in Minutes)	Accuracy (in %)	Correctness (in %)
1	Female_01	29	10.35	3.45	48.78	55.70
2	Female_01	33	20.47	7.5	57.30	61.24
3	Female_02	29	7.47	2.4	56.77	64.72
4	Female_02	33	9.26	3.83	63.80	70.84

Figure 7.3 clearly depicts the comparative analysis for the data set for each female speaker of read speech mode with 29 phones and 33 phones.

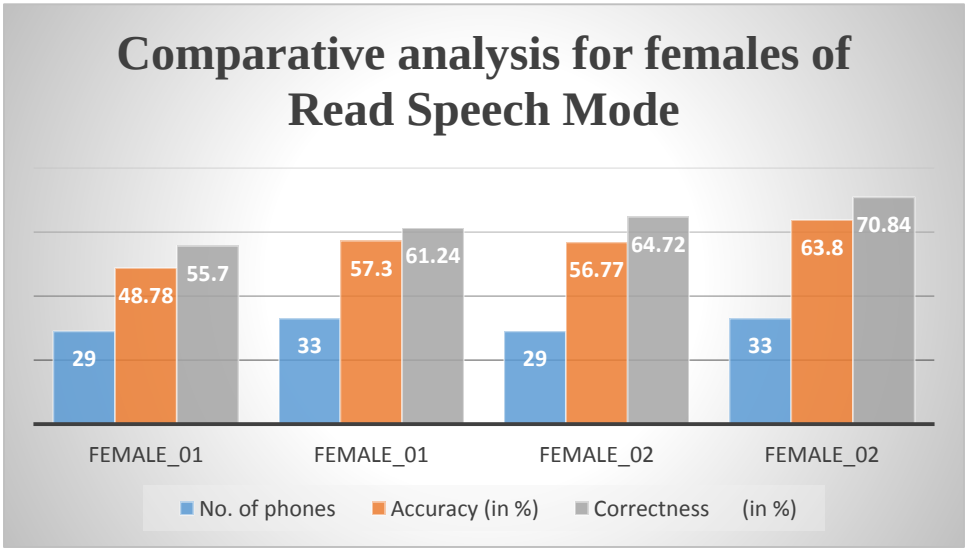


Figure 7.3: Comparative Analysis of system for each female of Read Speech Mode with 29 and 33 phones

Similar to each female speaker, the system has also been trained and tested for each individual male speaker both with 29 phones and 33 phones data set. Same as that of individual female data sets, the total male data is divided into each individual male speakers and then testing and training on their respective data sets is done.

The accuracy of the system for total male_01 of read speech mode with 29 phone as well as 33 phone is represented in the form of confusion matrix in the same way as shown in figure 7.1. Here with 29 phones, correctness percentage is 72.53 % and accuracy percentage is 60.93% and 80.24% correctness and 72.98% accuracy in case of 33 phones.

The accuracy of the system for total male_02 of read speech mode with 29 phones and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. Here with 29 phones, correctness percentage is 66.67% and accuracy percentage is 57.64% and 73.92% correctness and 62.40% accuracy in case of 33 phones.

Table 7.4 clearly shows the division of times in training and testing data sets, accuracy and correctness of the system for each individual male speaker of read mode with 30 phones and 34 phones.

Table 7.4: Testing accuracy of system for each male of read speech mode with 29 and 33 phones

Sr. No.	Speaker	No. of phones	Training Data (in Minutes)	Testing Data (in Minutes)	Accuracy (in %)	Correctness (in %)
1	Male_01	29	92.33	29.25	60.93	72.53
2	Male_01	33	103.51	34.53	72.98	80.24
3	Male_02	29	29.5	9.98	57.64	66.67
4	Male_02	33	34.51	11.50	62.40	73.92

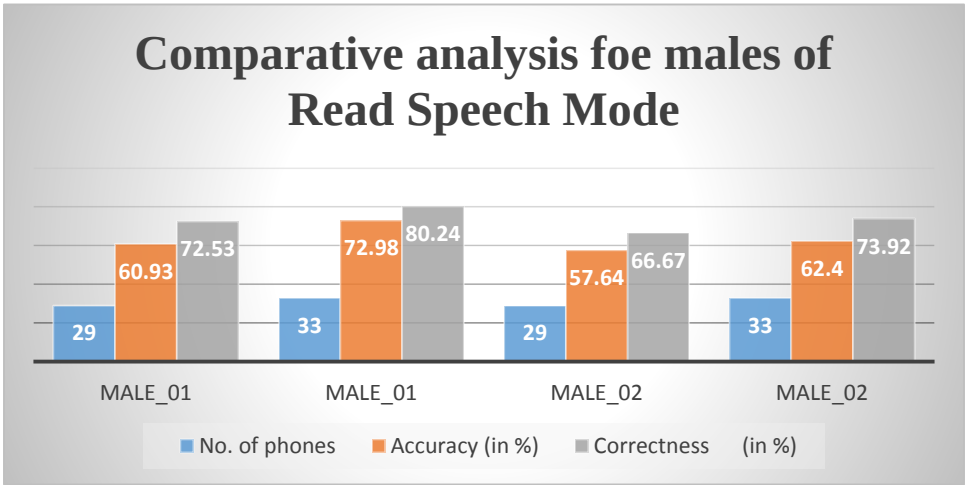


Figure 7.4: Comparative Analysis of system for each male of Read Speech Mode with 29 and 33 phones

Figure 7.4 clearly depicts the comparative analysis for the data set for each male speaker of read speech mode with 29 phones and 33 phones.

7.2 Performance Evaluation for Lecture Speech Mode

The system has also been trained for Lecture Speech Mode with both the categories of data set, *i.e.* 29 phones and 33 phones. For lecture mode, the system has been trained as well as tested speaker wise and transcriber wise too.

The accuracy and correctness of the system trained and tested for total data of lecture speech mode using 29 phones and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. In case of 29 phones, the system attains 44.99% correctness and 41.45% accuracy in this mode. Total number of substituted phones (S) are 5019, total number of correctly recognized phones (H) are 10269, number of deleted phones (D) are 7536, number of inserted phones (I) are 809, and total number of the phones (N) are 22824. Whereas in case of 33 phones, the system attains 64.49% correctness and 50.76% accuracy in this mode. Total number of substituted phones (S) are 5324, total number of correctly recognized phones (H) are 10203, number of deleted phones (D) are 7408, number of inserted phones (I) are 854, and total number of the phones (N) are 22935.

Table 7.5 clearly specifies the testing accuracy of system trained and tested for total data of lecture speech mode with 29 phones as well as 33 phonemes, corresponding to their training and testing data.

Table 7.5: Testing Accuracy of system for total data of lecture speech mode with 29 and 33 phones

Lecture Speech Data	No. of Phones	Training Data (in Minutes)	Testing Data (in Minutes)	Accuracy (in %)	Correctness (in %)
Total_data	29	93.31	31.10	41.45	44.99
Total_data	33	98.78	38.20	50.76	64.49

Figure 7.5 clearly illustrates the comparative analysis total data of lecture speech mode with 29 phones and 33 phones.

The system has been trained for various transcribers of lecture speech mode too and the experiment of calculating accuracy and correctness has been done for both 29 phonemes and 33 phonemes.

The accuracy and correctness of the system trained and tested for transcriber_01 of lecture speech mode using 29 phones as well as 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. In case of 29 phones, the confusion matrix shows that the system attains 50.31% correctness and 45.59% accuracy and in case of 33 phones, the confusion matrix shows that the system attains 66.56 % correctness and 53.32% accuracy.

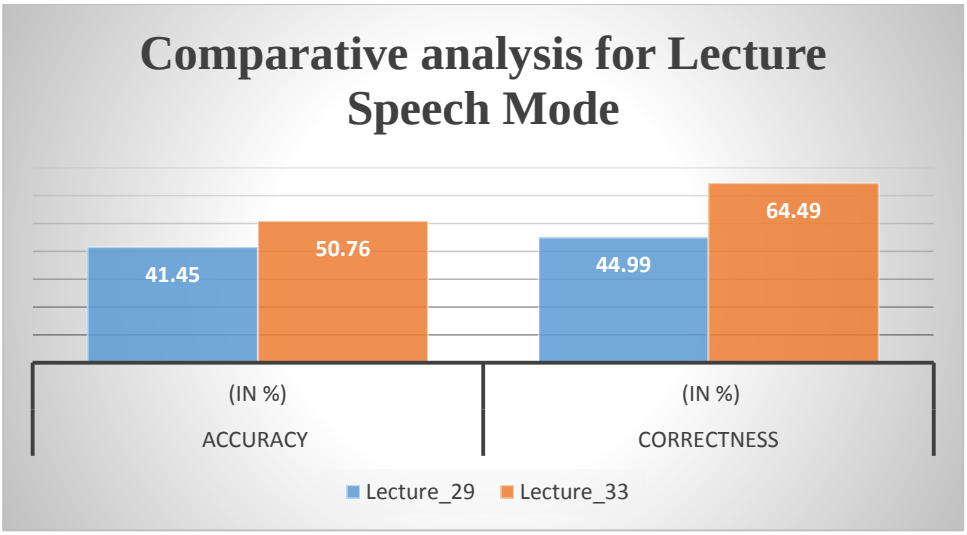


Figure 7.5: Comparative analysis of system for Lecture Speech Mode with 29 and 33 phones

The accuracy and correctness of the system trained and tested for transcriber_02 of lecture speech mode using 29 phones and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. In case of 29 phones, the confusion matrix shows that the system attains 51.73% correctness and 43.76% accuracy in this mode and 65.61 % correctness and 56.59% accuracy with 33 phones.

The accuracy and correctness of the system trained and tested for transcriber_03 of lecture speech mode using 29 and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. With 29 phones, the confusion matrix shows that the system attains 49.84% correctness and 45.51% accuracy and 70.93 % correctness and 57.39% accuracy with 33 phones.

The accuracy and correctness of the system trained and tested for transcriber_04 of lecture speech mode using 29 phones as well as 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. The confusion matrix shows that the system attains 55.99% correctness and 41.20% accuracy with 29 phones and 78.82 % correctness and 60.92% accuracy with 33 phones.

For comparative analysis, the accuracy and correctness of each transcriber corresponding to the data transcribed by them for 29 phones as well as for 33 phones are enlisted in table 7.6.

Table 7.6: Testing Accuracy of system for lecture speech transcribers with 29 and 33 phonemes

Sr. No.	Transcriber's ID	No. of phones	Training Data (in minutes)	Testing data (in minutes)	Accuracy (in %)	Correctness (in %)
1	01	29	35.25	11.76	45.59	50.31
2	01	33	36.27	12.80	53.32	66.56
3	02	29	32.45	10.81	43.76	51.73

4	02	33	33.59	11.90	56.59	65.61
5	03	29	15.36	4.12	45.51	49.84
6	03	33	16.45	5.15	57.39	70.93
7	04	29	15.34	4.13	41.20	55.99
8	04	33	16.70	6.20	60.92	78.82

The Comparative analysis of the experiments performed on each transcriber of lecture speech data transcribed by them for 29 phones as well as 33 phones is depicted through the figure 7.6.

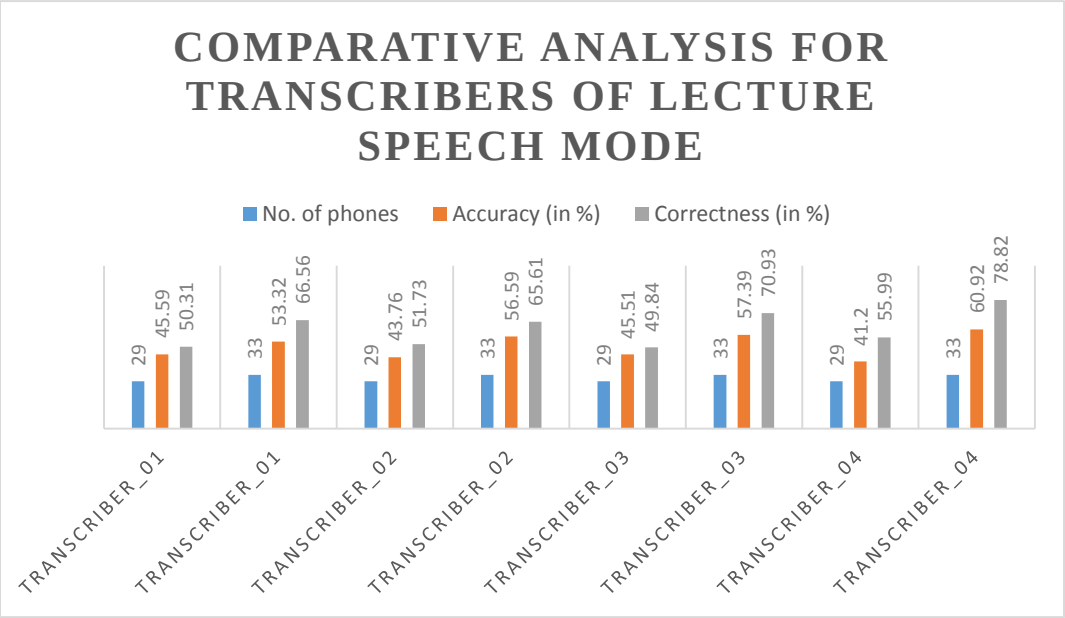


Figure 7.6: Comparative analysis of system for transcribers of Lecture Speech Mode with 29 and 33 phones

As discussed earlier the system has been trained for various speakers of lecture speech mode too and the experiment has been done for both the schemes *i.e.* 29 phonemes and 33 phonemes.

The accuracy and correctness of the system trained and tested for speaker_01 of lecture speech mode having 29 phones and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. The confusion matrix shows that the system attains 50.98 % correctness and 45.68 % accuracy with 29 phones and 65.72 % correctness and 54.88 % accuracy with 33 phones.

The accuracy and correctness of the system trained and tested for speaker_02 of lecture speech mode having 29 and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. In case of 29 phones, the confusion matrix shows that the system attains 57.88 % correctness and 48.43 % accuracy and with 33 phones, the confusion matrix shows that the system attains 66.56 % correctness and 50.21 % accuracy.

The accuracy and correctness of the system trained and tested for speaker_03 of lecture speech mode having 29 phones and 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. The confusion matrix shows that the system attains 41.65 % correctness and 36.75 accuracy with 29 phones and 59.85 % correctness and 48.96 % accuracy with 33 phones.

For comparative analysis, the accuracy and correctness of each speaker corresponding to the data recorded by them for 29 and 33 phones are enlisted in table 7.7.

Table 7.7: Testing Accuracy of system for lecture speech speakers with 29 phonemes and 33 phones

Sr. No.	Speaker's ID	Phones	Training Data (in minutes)	Testing data (in minutes)	Correctness (in %)	Accuracy (in %)
1	01	29	36.28	12.10	50.7	46.85
2	01	33	37.30	15.60	65.27	54.88
3	02	29	25.23	9.03	47.88	43.43
4	02	33	26.45	11.09	66.56	50.21
5	03	29	31.40	9.57	41.65	36.75
6	03	33	33.50	11.77	59.85	48.96

The Comparative analysis of the experiments performed on different speakers of lecture speech data with 29 phones as well as 33 phones is depicted through the figure 7.7.

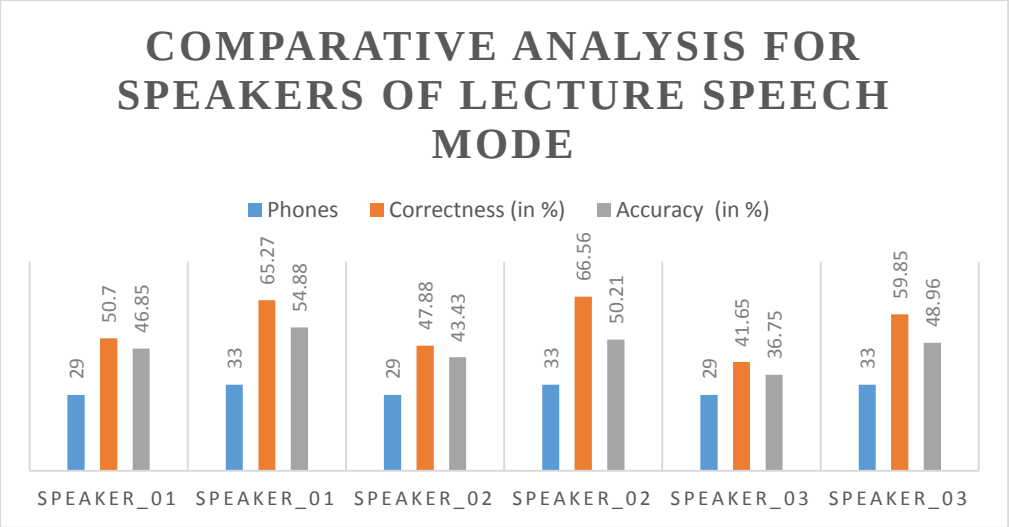


Figure 7.7: Comparative Analysis of system for Speakers of Lecture Speech Mode with 29 and 33 phones

7.3 Performance Evaluation for Conversational Speech Mode

In conversational mode of speech, two or more speakers are entitled to talk on a topic in their natural voice in the open environment. In this mode, the system has been trained and tested with 29 phones as well as 33 phones data set. In this case, approximately 35 minutes of data is processed.

The accuracy and correctness of the system trained and tested for conversational speech mode with 29 phones as well as 33 phones is represented in the form of confusion matrix in the same way as shown in figure 7.1. With 29 phones, the confusion matrix shows that the system attains 23.57 % correctness and 21.00 % accuracy in this mode. Total number of substituted phones (S) are 484, total number of correctly recognized phones (H) are 413, number of deleted phones (D) are 855, number of inserted phones (I) are 45, and total number of the phones (N) are 1752. Whereas with 33 phones, the confusion matrix shows that the system attains 31.45 % correctness and 25.34 % accuracy in this mode. Total number of substituted phones (S) are 680, total number of correctly recognized phones (H) are 530, number of deleted phones (D) are 475, number of inserted phones (I) are 189, and total number of the phones (N) are 1685.

For comparative analysis, table 7.8 clearly specifies the testing accuracy of system trained and tested for conversational speech mode with 29 phones as well as 33 phonemes, corresponding to their training and testing data.

Table 7.8: Testing accuracy of system for Conversational Speech Mode with 29 and 33 phones

Conversational Speech Data	No. of phones	Training Data (in Minutes)	Testing Data (in Minutes)	Correctness (in %)	Accuracy (in %)
Convsp_30	29	15.08	5.04	23.57	21.00
Convpsp_34	33	25.15	10.30	31.45	25.34

Figure 7.8 clearly renders the comparative analysis of conversational speech mode with 29 phones and 33 phones.

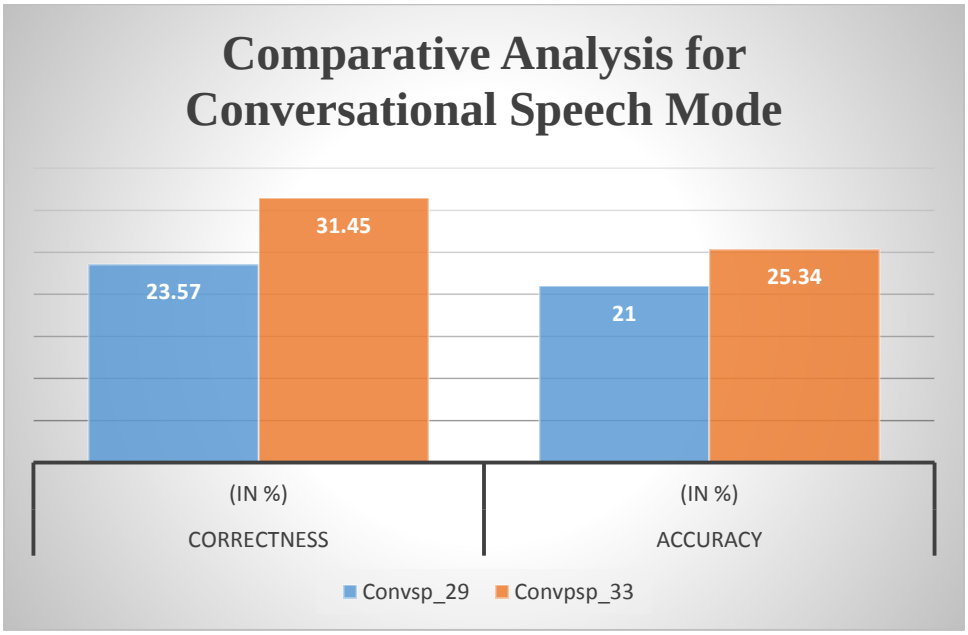


Figure 7.8: Comparative Analysis of system for Conversational Speech Mode with 29 and 33 phones

7.4 Performance Evaluation for Word Level Speech Recognition

In word level speech recognition, a comparison of manual and actually generated word level mlf files is done. After comparing the result is shown in the form of accuracy % and correctness % on the terminal. Figure 7.9 shows the result of word level recognition system.

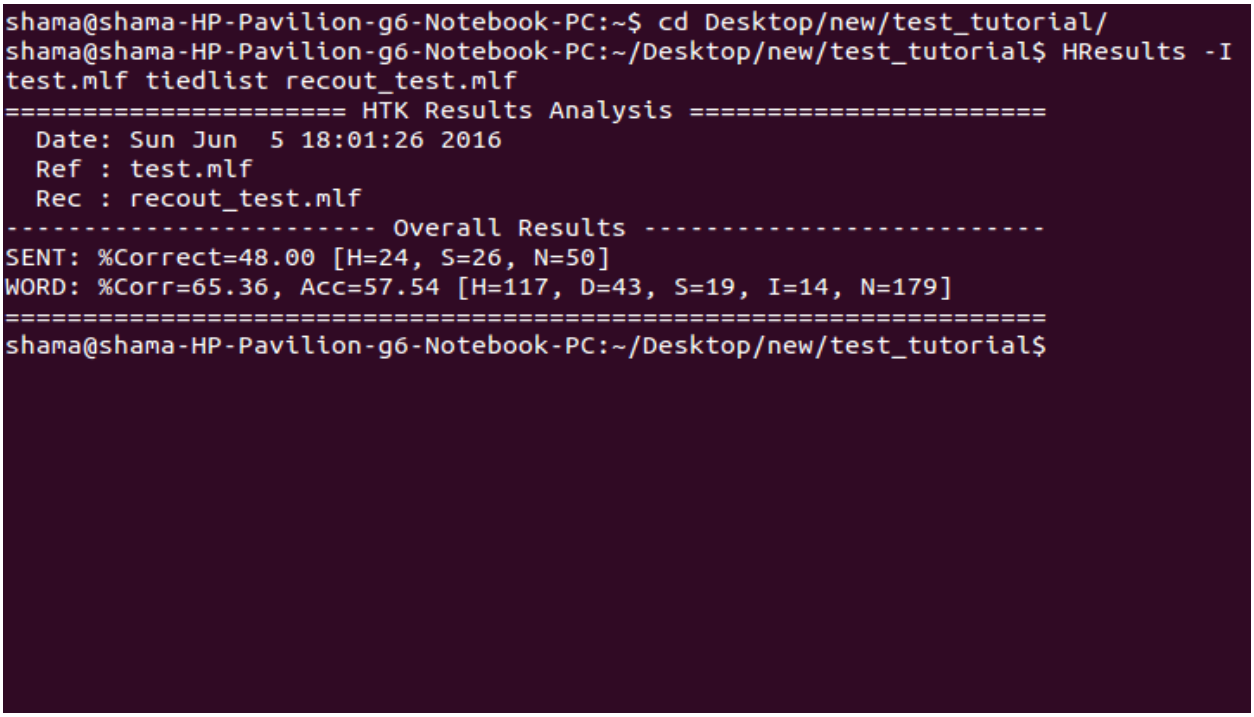


Figure 7.9: Result of word level speech recognition

CHAPTER 8

Conclusion and Future Scope

8.1 Conclusion

In this work, the total data for a duration of about 398.76 minutes is collected. This data has been collected in three different modes of speech, *i.e.* read speech mode, lecture speech mode and conversational speech mode. Here the system developed is a speaker independent system. So, this system is trained with two different schemes: one with 29 phones (excluding silence) and another with 33 phones (excluding silence) for building up the MLF file. This file is compulsory to train the system accordingly.

In this work, out of the total duration of data, about 226.33 minutes of data is collected for read speech mode, 136.98 minutes of data is collected for lecture speech mode and approximately 35.45 minutes of data is for conversational mode. This data is used both for 29 phones (excluding silence) scheme as well as for 33 phones (excluding silence) scheme.

Here, approximately 186.33 minutes of data out of total data for read speech mode is used for read speech mode with 29 phones (excluding silence) and 226.33 minutes of data is used for read speech mode with 33 phones (excluding silence). In lecture speech, 124.41 minutes of data out of total data for lecture speech is used for lecture speech mode with 29 phones (excluding silence) and 136.98 minutes of data is used for lecture speech mode with 33 phones (excluding silence). In case of conversational speech, about 20.12 minutes of total data for conversational speech is used for conversational speech with 29 phones (excluding silence) and about 35.45 minutes of data is used for conversational speech with 33 phones (excluding silence).

After collecting the whole data, it is transcribed using IPA chart, to represent all the basic units of sound that are present in the spoken utterances, in the symbolic form.

HTK toolkit has been used for training and testing of the system with a platform Ubuntu-14.04 of 64-bit. HTK toolkit is used because, it is a statistical tool used for constituting HMMs. For training the system, a set of 30 unique phones (including silence) and an uninterrupted concretion HMMs have been used. For second case, 34 unique phones (including silence) have been used in place of 30 phones. The system has been trained for all the three different modes of speech, *i.e.*, read, lecture and conversational speech mode.

In read speech mode, the system is trained for each gender collectively and for individual speaker too. In lecture speech mode, the system is trained for each Punjabi speaker and for each transcriber, who transcribed the Punjabi spoken utterances. In conversational speech mode, the whole conversation is trained. In each of the three case, 75% of the total data is used for training the system and 25% of the data is used for evaluating the performance of the system.

In word level speech recognition system, Julian scripts of Julius toolkit are used for training as well as testing of the system. The system is trained and tested for word recognition using the triphone models. Thus, after creating the triphone models the actual (system generated) and manually created mlf files are compared and the result is displayed on the terminal itself. The comparison gives a result of 57.54% accuracy and 65.36% correctness of the system used for word recognition.

8.2 Future Scope

The correctness as well as the accuracy of the system can be increased by increasing the number of recognized phonemes as well as training the data accordingly. The system can be developed with more Gaussian Mixtures and large number of MFCC features for the training purpose.

This same work can also be extended to syllable level recognition. Syllables are those sound units that focus mainly on the presence vowel surrounded by the consonants. Therefore, it will give an enchanting accuracy.

By using the same procedure, the system can be refined for other Indian languages like Hindi, Marathi, Kannad *etc*.

The correctness and accuracy of the system at word level can be increased by increasing the time duration of the data *i.e.* considering more data.

Here, in the proposed system, the output is taken with the help of HTK commands using pre-recorded data. So the output can also be achieved by speaking the testing file on the time of testing with the help of Audacity. For this, in place of HTK Julius toolkit is used.

References

1. HTK “Hidden Markov Model Toolkit”, available at “http://htk.eng.cam.ac.uk”, 2012.
2. L.A. Ogunsola, “Information and Communication Technologies and Effects of Globalization,” http://southernlibrarianship.icaap.org/content/v06n01/ogunsola_l01.htm, 2005.
3. Michael Collins. Language Modelling. Amsterdam, Columbia University Department of Computer Science, 2012.
4. “List of languages by number of native speakers,” http://www.ethnologue.com/statistics/size, 2010.
5. Stephen Tu. Derivation of Baum-Welch Algorithm for Hidden markov Models, 2006.
6. Steve Young, Gunnar Evermann, Mark Gales. HTK Book (for HTK version 3.4). England, Cambridge University of Engineering Department, 2006.
7. Richard Guitierrez. Refinements for HMM. Texas A&M University.
8. Er. Sheilly Padda, Rupinderdeep Kaur, Er. Nidhi,”Punjabi Phonetic: Punjabi Text to IPA Conversion”, International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 10, 2012,pp.475-478.
9. Parminder Singh and Gurpreet Singh Lehal, “Text-To-Speech Synthesis System for Punjabi Language”, January 2011.
10. M. H. Mateus, E. d'Andrade, “The Phonology of Portuguese: The phonology of the world's languages”, Oxford University Press Inc. New York, 2000.
11. Anthony Atkielski, “Using Phonetic Transcription in Class” ,2005
12. Prof. Ian Maddieson,”Applied Phonetics: Portuguese Text-to-Speech”, University of California, Berkeley Linguistics 110: May 16, 2003.
13. “Introduction of Julius,” http://julius.osdn.jp/en_index.php.
14. L.R. Rabiner , “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, Proc. of the IEEE Vol. 77, Issue 2,pp. 257–286,1989.
15. Lee, K. F., Hon, H. W., Hwang, M. Y., and Mahajan, S. (1989), "The SPHINX speech recognition system", *Proceedings of the IEEE International Conference in Acoustics, Speech and Signal* Processing.
16. Lee, K. F., and Hon, H. W. (1989). "Speaker-independent phone recognition using hidden Markov models", IEEE Transactions on Acoustics, Speech and Signal Processing, 37(11), 1641-1648.

17. Lamel, L. F., and Gauvain, J. L. (1992). "Experiments on speaker-independent phone recognition using BREF", *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, 1, 557-560.
18. Ratnayake, N., Savic, M., and Sorensen, J. (1992). "Use of semi-Markov models for speaker-independent phoneme recognition", *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, 1, 565-568.
19. Woodland, P. C., Leggetter, C. J., Odell, J. J., Valtchev, V., and Young, S. J. (1995). "The HTK large vocabulary speech recognition system", IEEE *International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, 1, pp.73-76.
20. Leggetter, C. J., and Woodland, P. C. (1995). "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models", *Computer Speech and Language*, *9*(2), 171-185.
21. Mari, J. F., Fohr, D., and Junqua, J. C. (1996). "A second-order HMM for high performance word and phoneme-based continuous speech recognition", *Proceedings of* the *IEEE International Conference on Acoustics, Speech, and Signal Processing,. ICASSP*, 1, pp.435-438.
22. Ming, J., and Smith, F. J. (1998). "Improved phone recognition using Bayesian tri phone models", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1, 409-412.
23. Zheng, F., Song, Z., Xu, M., Wu, J., Huang, Y., Wu, W., and Bi, C. (1999). "Easytalk: a large-vocabulary speaker-independent Chinese dictation machine", *EuroSpeech*.
24. Young, S. (1999). "Acoustic modeling for large vocabulary continuous speech recognition", Springer Berlin Heidelberg, Computational Models of Speech Pattern Processing, 18-39.
25. Pruthi, T., Saksena, S., and Das, P. K. (2000). "Swaranjali: Isolated word recognition for Hindi language using VQ and HMM", International Conference on Multimedia Processing and Systems, ICMPS, IIT Madras.
26. Woodland, P. C., and Povey, D. (2002). "Large scale discriminative training of hidden Markov models for speech recognition", *Computer Speech and Language*, *16*(1), 25-47.
27. Nwe, T. L., Foo, S. W., and De Silva, L. C. (2003). "Speech emotion recognition using hidden Markov models", *Speech communication*, *41*(4), pp.603-623.

28. Hasan, M. R., Jamil, M., Rabbani, M. G., and Rahman, M. S. (2004). "Speaker Identification Using Mel Frequency Cepstral Coefficients", *International Conference on Electrical and Computer Engineering, ICECE,* 565-568.
29. Satori, H., Harti, M., and Chenfour, N. (2007). "Introduction to Arabic speech recognition using CMUSphinx system", *Proceeding of the Information and Communication Technologies International Symposium, ICTIS.*
30. Bhuriyakorn, P., Punyabukkana, P., and Suchato, A. (2008). "A genetic algorithm-aided Hidden Markov Model topology estimation for phoneme recognition of thai continuous speech", *IEEE Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD,* 475-480.
31. Elshafei, M., Al-Muhtaseb, H., and Al-Ghamdi, M. (2008). "Speaker-independent natural Arabic speech recognition system", *International Conference on Intelligent Systems*.
32. Alotaibi, Y. A. (2008). "Comparative study of ANN and HMM to Arabic digits recognition systems", *Engineering Sciences, 19*(1), 43-60.
33. Azmi, M., Tolba, H., Mahdy, S., and Fashal, M. (2008). "Syllable-based automatic Arabic speech recognition", Proceedings of the 7th WSEAS International Conference on Signal Processing, Robotics and Automation, World Scientific and Engineering Academy and Society, WSEAS, pp.246-250.
34. Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., and Woodland, P. (2009). *The HTK Book.* Cambridge University.
35. R. Kumar "Comparison of HMM and DTW for Isolated Word Recognition of Punjabi Language" In Proceedings of Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, Sao Paulo, Brazil. Vol. 6419 of Lecture Notes in Computer Science (LNCS), pp. 244–252, Springer Verlag, November 8-11, 2010.
36. B. A. Q. Al-Qatab and R. N. Ainon, "Arabic Speech Recognition Using Hidden Markov Model Toolkit (HTK)", Paper presented at International Symposium in Information Technology (ITSim). Kuala Lumpur, June 15-17, 2010.
37. Mandal, S., Das, D., Mitra, P.: SHRUTI-II: A Verncular Speech Recognition System in Bengali and an Application for Visually Impaired Community, pp. 229-233, 2010.
38. R. Kumar and M. Singh, "Spoken isolated Word Recognition of Punjabi Language Using dynamic time Warping Technique" Demo in Proceedings of Information System for Indian Languages,

Punjabi University, Patiala, India, March 9 - 11, 2011. Vol. 139 of Communication in Computer and Information Science (CCIS), Page 301, Springer Verlag.

39. Kumar, K., and Aggarwal, R. K. (2011). "Hindi speech recognition system using HTK", *International Journal of Computing and Business Research*, *2*(2), pp.2230-6166.
40. K. Kumar, R. K. Aggarwal, and A. Jain "A Hindi speech recognition system for connected words using HTK" International Journal Computational Systems Engineering, Vol. 1, No. 1, 2012
41. Choudhary, A., Chauhan, M. R., and Gupta, M. G. (2013). "Automatic Speech Recognition System for Isolated and Connected Words of Hindi Language by Using Hidden Markov Model Toolkit (HTK)".
42. Saini, P., Kaur, P., and Dua, M. (2013). "Hindi Automatic Speech Recognition Using HTK", *International Journal Of Engineering Trends And Technology*, *4*.
43. Sarma, B. D., Sarma, M., Sarma, M., and Prasanna, S. R. M. (2013). "Development of Assamese Phonetic Engine: Some Issues", *Annual IEEE India Conference, INDICON*.
44. Mankala, S. R., Bojja, S. R., Ramaiah, V. S., and Rao, R. R. (2014). "Automatic Speech Processing Using HTK for Telugu Language", *International Journal of Advances in Engineering and Technology*, *6*(6), 2572-2578.

List of Publications

1. Shama Mittal and Rupinderdeep Kaur “Implementation of Phonetic level Speech to Text Recognition System for Punjabi Language”, in International Conference on Data Science and Engineering (IEEE) (ICDSE-2016) [Accepted]
2. Shama Mittal and Rupinderdeep Kaur “Implementation of Word level Speech to Text Recognition System for Punjabi Language”, in International Journal of Computer Applications (IJCA-2016) [Communicated]

Video Link

https://youtu.be/qpmNU2aoIbM

Thesis

ORIGINALITY REPORT

9%	6%	6%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.thapar.edu:8080 Internet Source	1%
2	Bassam A. Q. Al-Qatab. "Arabic speech recognition using Hidden Markov Model Toolkit(HTK)", 2010 International Symposium on Information Technology, 06/2010 Publication	1%
3	spsc.inw.tugraz.at Internet Source	1%
4	research.ijcaonline.org Internet Source	<1%
5	Dua, Mohit; Aggarwal, R. K.; Kadyan, Virender and Dua, Shelza. "Punjabi Automatic Speech Recognition Using HTK", International Journal of Computer Science Issues (IJCSI), 2012. Publication	<1%
6	gdeepak.com Internet Source	<1%
7	www.inderscience.com	