Segmentation of Punjabi Speech Signals into Phonemes Using Hidden Markov Models

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

Master of Technology
in
Computer Science and Applications

Submitted By
Divya Bansal
(Roll No. 601003007)

Under the supervision of:
Khushneet Jindal
System Analyst, SMCA

SCHOOL OF MATHEMATICS AND COMPUTER APPLICATIONS
THAPAR UNIVERSITY
PATIALA – 147004

June 2012

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Segmentation of Punjabi Speech Signals into Phonemes Using Hidden Markov Models*", in partial fulfillment of the requirements for the award of degree of Master of Technology in *Computer Science and Applications* submitted in School of Mathematics and Computer Applications, Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Khushneet Jindal* 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.

(Divya Bansal)

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

(Khushneet Jindal)
System Analyst
SMCA

Countersigned by

(Dr. S.S. Bhatia)
Head
School of Mathematics and Computer Applications
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

ACKNOWLEDGEMENT

First of all, I would like to express my gratitude to Mr. Khushneet Jindal**,** System Analyst**,** School of Mathematics and Computer Applications, Thapar University, Patiala for his patient guidance and support throughout this report. I am truly very fortunate to have the opportunity to work with him.

I am also thankful to our Head of the Department, Dr. S.S. Bhatia as well as PG Coordinator, Mr. Singara Singh, Assistant Professor**,** School of Mathematics and Computer Applications, entire faculty and staff of School of Mathematics and Computer Applications and my friends who devoted their valuable time and helped me in all possible ways towards successful completion of this work. I thank all those who have contributed directly or indirectly to this work.

Lastly, I would also like to thank my parents for their years of unyielding love and encouragement. They have always wanted the best for me and I admire their determination and sacrifice.

Divya Bansal
(601003007)

ABSTRACT

Recently, the use of computers in speech synthesis has become an important area of research among speech and computer scientists and linguists. Speech synthesis refers to the artificial production of human speech. For this purpose, speech synthesis systems often called text-to-speech (TTS) systems are developed that "read" text from a document, Web page etc. and generate speech in the form of audio wave or mp3 files. These systems (TTS) are very useful majorly for visually impaired people especially those having poor vision or visual dyslexia, for illiterate people who can understand spoken native language, for educational and research purposes. All TTS systems are developed with the aim to produce high quality synthesized speech which is both natural, intelligible, can be correctly understood and interpreted by the user.

This thesis attempts to implement speech synthesis support for Punjabi language in mobile device. It is achieved by segmenting a speech database into smaller units using HMM Toolkit (HTK) based on hidden markov models (HTS approach) that are further concatenated to generate speech signals. The proposed system converts English text in the form of the caller's name stored in contact list into Punjabi speech in mobile phones.

The input text data is initially processed in pre-processing stage for titles like Mr. Tapas, numbers like in Bharat1234, initials like K.K. Sharma and thereafter, the processed data is used in training and testing phase of HTK. With the help of HTK, various HMM acoustic models are firstly trained using spectral features (Mel-Cepstral Coefficients) extracted from the recorded Punjabi speech corpus and various context-independent monophones and context-dependent triphones models are generated. For example for word "bharat" generated monophones are a, bh, t etc. & triphones are bh-a+r. Later in the testing phase, correct phoneme sequence from a network of all possible combinations is generated corresponding to the test sample word using HMM models and feature vectors like for the word "Tapas" the output phoneme sequence is ਤ, ਪ, ਸ instead of phoneme sequence ਟ, ਪ, ਸ. These phoneme sequences are given as input to the application to generate speech signals by concatenating the phonemes.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER -1

INTRODUCTION

1.1 GENERAL INTRODUCTION

Speech is one of the most important ways for humans to communicate, so there have been a great number of efforts to incorporate speech into human-computer or man-machine communication environments so that humans can interact with machines through speech signals. As computers become more interactive and useful in daily life, demands for technologies in speech processing areas, such as speech recognition, speech understanding, natural language processing, and speech synthesis are increasing to establish high-quality human-computer communication.

In the present era of human computer interaction, the educationally under privileged and the rural communities of India are being deprived of technologies that pervade the growing interconnected web of computers and communications. One good solution for this problem would be computers talking to the common man in the language in which he/she is comfortable to communicate in [22].

Text-to-speech synthesis (TTS), one of the important technologies in speech processing area, is a technique for creating speech signal from given text or representations of text in phonetic transcriptions in order to transmit information from a machine to a person in the maximum possible natural and intelligible form. To fully transmit information contained in the speech signals, text-to-speech synthesis systems are required to have an ability to generate natural sounding speech with speaker's voice characteristics, various speaking styles and emotions that the listener can understand correctly. Based on the application, different implementations of a speech synthesizer may be used and there are various criteria for evaluating the resulting speech or the system as a whole, and various approaches can be used to meet the required specifications [21].

1.2 TTS APPLICATIONS

There are numerous applications of Text-To-Speech Synthesis Systems:-

1.2.1 AID TO VISUALLY IMPAIRED AND ILLITERATE PEOPLE

The human computer interaction greatly depends on written text and images that makes the use of computers difficult or rather impossible for visually and physically impaired and illiterate masses. Automatic speech generation from various natural language sentences can overcome these obstacles. Man-Machine communication helps illiterate and visually challenged people to interact with computers and other devices efficiently. Before synthesized speech was used, various audio books were there where the content of the book was stored and read into audio tape. It is obvious that making such spoken copy of various large books will take several months and is a very expensive approach.

These days, the synthesizers are mostly software based using various approaches and software tools like HTK (HMM Tool Kit), SPTK (Speech Signal Processing Toolkit) etc. As such with scanner and OCR (Optical Character Recognition) system, it is easy to construct a reading machine that convert written text to speech form for any computer environment with sufficient use of resources. Speech synthesis is currently used to read browser pages or other forms of media with normal personal computer. TTS System assembled with the Optical Character Recognition (OCR) systems, are also useful for blind and illiterate people.

1.2.2 EDUCATIONAL APPLICATIONS

Synthesized speech can also be used in many educational applications.

- A computer with speech synthesizer can teach 24 hours a day and 365 days a year i.e. 24*7 round the clock especially in remote areas.
- It can be used by the language teachers for teaching their tutorial in the language in which student is comfortable and can easily understand. Especially, with people who are visually impaired, synthesis may be very helpful because some children may feel themselves very embarrassing when they have to be helped by a teacher [13].
- It can be programmed for special phonetics tasks like spelling, pronunciation and different accent teaching for different languages. With proper computer TTS software, unsupervised training for such problems is easy and inexpensive to arrange.

- A speech synthesizer associated with word processor and browser is also an aid to proof reading. It also becomes easier to detect grammatical and stylistic problems when listening than reading text. Misspellings are also easier to detect.
- TTS synthesis coupled with a Computer Aided Learning system (Smart Classes) can provide a helpful tool for learning new languages. It is also very efficient to learn write and read with spoken help.

1.2.3 ROLE IN FUNDAMENTALS AND APPLIED RESEARCH

TTS synthesizers have a special feature, making them a helpful tool for linguists. As a result, the efficiency of intonative, rhythmic and phonetic models can be detected and analyzed. A lot of research has been done and going on in the field of speech synthesis, recognition etc. to develop browsers, TTS systems that convert text in desired languages or from one script to another like beta version of Sangam software that converts Gurumukhi script of Punjabi language into Shahmukhi script and other efforts are being carried out for developing browsers, mobile applications for other native languages and include different speaking styles to increase naturalness.

1.2.4 APPLICATIONS FOR TELECOMMUNICATIONS AND MULTIMEDIA

- Texts might range from simple messages to very large documents, which can easily be read and stored as digitized speech.
- Synthetic speech is very important in voice mail systems. Electronic mails have become very popular and important means of communication in last few years. But, it is difficult to read the mails without proper connectivity. Such situation may exist at various places, e.g., a person traveling in the plane. With speech synthesized from TTS, e-mail messages may be listened to via normal telephone lines.
- Synthesized speech may also be used to speak out short text messages (sms) and caller's identity in mobile phones.Synthesized speech is used in all types of telephone enquiry systems, but the quality is far from good for common customers. Today, with various types of TTS systems the quality has reached the level that normal customers are adopting it for everyday use.

1.2.5 TALKING BOOKS AND TOYS

With the help of TTS systems, the market of talking toys and books is achieving a great success. There is a huge development in the toys market through the TTS system which is very interactive.

There are talking toy laptops that produce either pre-recorded voices in different languages and accents or produce voices according to text input into it, mobile applications like Talking Tom that also uses speech processing technology to produce same voices that are given input to it and other applications like MPR (Multimedia Print Reader) which is a talking pen and uses listening skills along with traditional reading

Talking books like Aadarsh Talking Books also play a great role in efficient education in various languages and pronunciations. These can be used as an important tool of entertainment (in case of story books) and education (in case of text books) for children.

1.2.6 ROLE IN INTERACTIVE BROWSERS

TTS systems are being presently incorporated with the interactive voice browsers, which helps people, who are illiterate and blind but can understand speech sounds. They help to read the desired text in any language as required by the user for providing a wide access to most prevalent and highly useful internet to every person in any language. This makes internet more interactive and accessible even by those who are less educated and physically challenged. Various developments have been made in this area for various languages like Hindi, Tamil, and Portuguese etc.

1.2.7 VOCAL MONITORING

In most cases, oral information is more efficient than manually clicking some buttons etc. for input. Hence the idea of incorporating speech synthesizers in measurement or control systems as in cockpits etc. is very useful. This helps a lot in monitoring and operating various machines and software like washing machines, air-conditioners, refrigerators that can work through vocal commands also that make it more user-friendly.

1.3 TTS ARCHITECTURE

TTS software can "read" text and generate synthesized speech through a computer's speakers.

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read any text aloud, whether it was directly introduced in the computer by an operator or scanned and submitted to an Optical Character Recognition (OCR) system [6].

In TTS, the process of text to speech conversion allows the transformation of a text into string of phonetic and prosodic symbols and then into a synthetic speech signal. The quality of the resultant speech thus produced by a TTS synthesizer depends on the quality of the string given as input, as well as of the quality of the generation process.

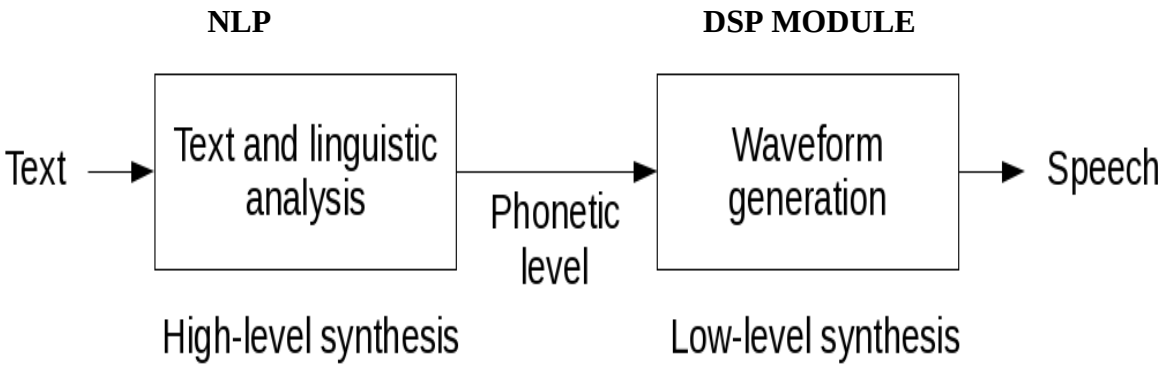


Fig. 1.1: TTS ARCHITECTURE

A text-to-speech system (or "engine") is composed of two parts:

1.3.1 FRONT END (ANALYZER)

Front end analyzes the text given as input to the system through various steps and processes it so that it can be used to produce speech signal corresponding to the given text in next phase i.e. DSP (Digital Signal Processing).

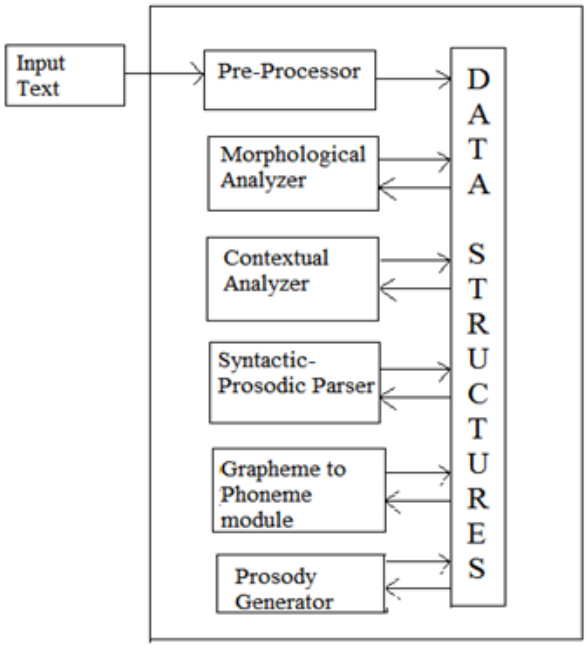


Fig. 1.2: NLP Module (Text Analyzer)

The front-end has following major tasks.

- **Pre-Processing** – This module converts raw text containing symbols like numbers, abbreviations and acronyms given as input to the system into equivalent written-out words and tries to remove punctuation ambiguities according to regular grammars. This process is often called text normalization, pre-processing, or tokenization.

  E.g. - 1776: one thousand seven hundred and seventy six

  Mrs. – Misses

  St. Joseph St. - Saint Joseph Street

- **Morphological Analysis** – In this phase all parts of speech i.e. POS categories e.g. noun, verb etc. are assigned to all the words in the pre-processed text according to their spellings and in the case of compound words, they are divided into basic units with help of regular grammar.

- **Contextual Analysis** – This phase focuses on the context of words in the text i.e. the neighboring words are also considered. Analysis of contexts of the words helps in reducing its part of speech (POS) categories according to the

possible POS category of neighboring words following certain techniques like n-gram etc.

- **Syntactic-Prosodic parser** – This parser lastly searches the left-over space and finds the text structure like clauses, phrases etc. of the words that relates it to its prosodic realizations.

- **Grapheme-to-Phoneme module** -- This phase (also called Letter-to-Sound module) then assigns phonetic transcriptions to each word either with the help of pronunciation dictionary (Dictionary based Phonetization) used in MITTALK system or with the help of certain rules (Rule based Phonetization)

  E.g. - /ow/: down

  /ae/: man

- **Prosody Generation** – Prosodic features like pitch, intensity, loudness etc. plays an important function in speech communication. These features segment the speech chain into groups of syllables; phrases etc. and present the relationship between them. Control over these prosodic features, gender, age, emotions etc. in speech can be used to model very natural sounding speech leading to the development of high quality speech synthesizer system.

1.3.2 BACK END (DSP COMPONENT)

The back-end (often referred to as the synthesizer) converts the symbolic linguistic representation into sound i.e. the transcriptions from NLP component are given as input and synthesized speech signal is generated as an output of this component. In certain systems, this part includes the computation of the target prosody (pitch contour, phoneme durations), which is then imposed on the output speech. As an output, at this phase, speech waveform is generated either with the help of various spectral and excitation parameters as in HMM based systems or by modeling the human speech articulators directly as in Articulatory synthesis or by using stored speech samples as in Concatenative or by generating speech frequencies using source-filter theory as in Formant approach of synthesis based on the choice of approach or other constraints like naturalness, intelligibility, memory etc. . This is called low-level synthesis.

1.4 QUALITY OF A TTS SYNTHESIZER

Usually, the quality of TTS is decided by two quality factors stated below:

1.4.1 INTELLIGIBILITY

It refers to how easily the output speech can be understood i.e. the accuracy with which the listener can understand the spoken word. It can be analyzed by considering several kinds of speech units (phonemes, syllables, words, phrases, etc.) used to synthesize speech. Every TTS system is aimed to generate speech that do not contain ambiguities and produce correct voice signal according to the text so that listener can understand and interpret the meaning appropriately. E.g. whether the character string like S. corresponds to Sardar in S. Badal or initials like Sandeep in a person's name as in S.K Sinha in the text, this should be resolved by the system and correct speech must be produced corresponding to it.

1.4.2 NATURALNESS

It refers to the measure of closeness of output synthesized speech signals and the human speech sounds i.e. how much it sounds like the natural speech. Listening to a synthetic voice must allow the listener to attribute this voice to some pseudo-speaker and to understand some kind of expressivities and emotions as well as some features characterizing the speaking style of speaker. For this purpose, the corresponding extra-linguistic information must be supplied to the synthesizer [9]. In order to generate natural voice various factors like prosody generation, different speaking styles, nasalized voice, pitch differences for male and female voice are required to be considered.

It is the most important factor in the development of speech synthesizers that it becomes hard for the listeners to distinguish between the voice generated by machine or produced by human being and research is being performed in this field to incorporate this feature for better quality of TTS systems.

Most of the existing TTS synthesizers produce an acceptable level of intelligibility, but the naturalness dimension, the ability to control expressivities, speech style and pseudo-speaker identity are still now the areas of concern and need improvements [9].

However, demands of users vary according to the field of application in which synthesized voice is to be used e.g. in general applications such as in telephonic information retrieval more natural voice is required, while in some applications where professionals (process or vehicle control) or physically challenged people are involved more intelligible speech rather than natural speech is required so that they can understand it correctly even if it sounds like a machine generated speech signal.

1.5 SPEECH SYNTHESIS TECHNIQUES

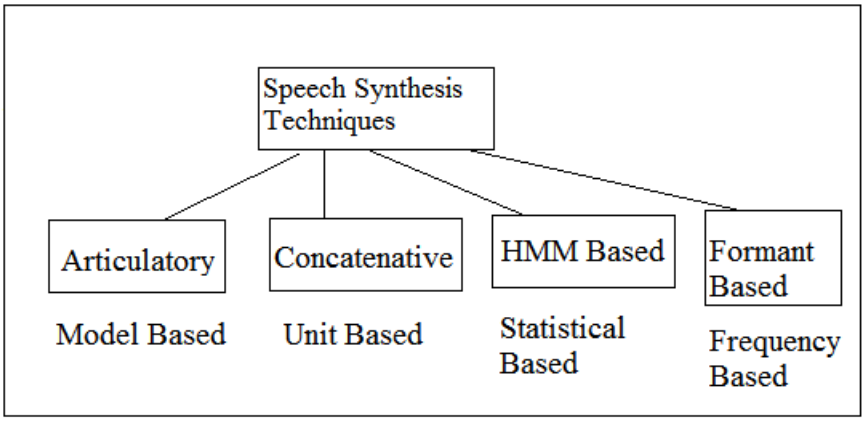


Fig. 1.3: Speech Synthesis Techniques

Synthesized speech can be produced by several different methods. All of these have certain benefits and deficiencies. The methods are classified into three groups:

1.5.1 ARTICULATORY SYNTHESIS

This technique attempts to model the human speech production system (especially vocal tract system, various articulators' viz. lip, tongue, jaw etc.) and articulatory processes directly. It tries to model the human vocal organs as accurately as possible, so it is most effective method to produce high-quality synthetic speech that can be used in various applications as compared to other approaches as it aims to articulate natural synthesis system of body.

But the disadvantage is that it is the most difficult methods to implement because it requires analyzing and modeling human vocal organs and articulators which are difficult to realize practically and the computational load of the approach is also higher than other speech synthesis methods. Thus, this technique is not developed much as compared to its other counterparts. In order to implement this method certain

sets of area functions, formant frequency etc. of different articulators and parts of speech production system in body like vocal tract, glottis are modeled. The first articulatory system was based on table of vocal tract area functions from larynx to lips for each phonetic segment [23]. Another problem with this approach is the absence of sufficient amount of data and information about the motion, degree of freedom, and masses of articulators because X-ray analysis of speech provides 2-D data while real vocal tract is 3-D which then becomes difficult to realize.

Advantages of articulatory synthesis are that the vocal tract models allow accurate modeling of transients due to abrupt area changes, whereas formant synthesis models only spectral behavior [23]. The articulatory synthesis is rarely used today to synthesize speech signals, but as the various analysis methods are developing and the computational resources are increasing, it might be a promising and useful synthesis method in the future [15].

1.5.2 CONCATENATIVE SYNTHESIS

Concatenative text-to-speech systems can, in theory, produce very naturally sounding synthetic speech, since they directly join pre-recorded segments, waveforms or units to form any sentence as required in the input text. In practice, several factors in this approach leads to less perfect speech output quality. For instance, unit selection i.e. the choice of the best set of pre-recorded speech units e.g. words, syllables, phonemes etc. that can be used as building blocks or unit of concatenation to generate speech signal is a difficult task. Moreover, the concatenation of units recorded using different intonation; phonetic contexts, emotions, speaking styles, pitch, loudness etc. may produce suboptimal results even if the recorded unit set is reasonably complete. Other problems like time domain discontinuities and spectral mismatch may also come forward and need to be dealt with in the concatenation process.

This technique uses prerecorded samples or units of different length derived from natural speech. The most important aspects in concatenative synthesis are to find the correct unit length known as unit selection and this factor majorly decides the quality of speech signal produced. The selection is usually a trade-off between longer and shorter units that affects and includes various other factors like naturalness, smoothness at concatenation points, database memory required etc.

With longer units like words, high naturalness in generated voice, less concatenation points i.e. more smoothness and good control of co articulation are achieved, but the amount of required units and memory is increased as they can't be generalized for more phrases. With shorter units like syllables, phonemes etc., less memory is needed due to achieved generalization by concatenating them, but the procedure of collecting, labeling and segmenting the speech sample becomes more difficult and complex.

In present systems units used are usually words, syllables, demisyllables, phonemes, diaphones, and triphones.

There are several problems in concatenative synthesis:-

- Distortion from discontinuities at concatenation points, which can be reduced using diphones or smoothing signal [23].
- Memory requirements are very high due to pre-recorded database, especially when long concatenation units are used, such as syllables or words. These requirements further increase when prosodic features like pitch, loudness, different speaking styles etc. are also considered.
- Data collecting, labeling and segmentation of speech sample either manually or automatically are usually time-consuming procedures.

Concatenative methods are the most commonly used in present synthesis systems.

Various concatenative units used in this approach according to application are:-

- **Words**: - These concatenative units like in written text or messaging systems with limited vocabulary, are the most obvious choice for generating sentences by concatenating different words or for using them individually in case of caller's name (like ਕਰਨ) in mobile phones. Concatenation of words is

  easy and co-articulation effect which is there in case of concatenation of smaller units like syllables is low.

  But usage of words cannot generalize the database for more phrases and sentences. Moreover, speech signals of phrases generated by only concatenating words that were recorded in isolation have low quality and low naturalness.

- **Syllables: -** These are smaller units as compared to words and consist of combination of consonants and vowels e.g. ਰਅ (CV), ਰਅਮ (CVC), ਆ (V). They are made up of two or more phonemes viz. ਰ (C), ਅ (V), ਮ (C) etc. and further combine to form words. Unlike with words, the co-articulation effect is not included in stored units, so using syllables as a basic unit is not very reasonable [15]. Besides co-articulation problem, their large number e.g. 10,000 syllables in English, 59300 and more syllables in Punjabi [23] also presents recording and storage problems and thus TTS systems aim to cover maximum possible number of phonemes and not all of them. There are no syllable based concatenative TTS systems at this time [1].

- **Phonemes: -** Phonemes are the most commonly used units in speech synthesis because they represent speech linguistics e.g. ਅ \a\, ਕ \k\ etc. The TTS systems based on concatenation of phonemes need smaller databases consisting of 40 to 50 units [1]. Thus they provide more generalization over large number of phrases and sentences with smaller corpus. Also rule-based systems become more flexible by using phonemes as concatenation units. Phonemes are sometimes used as an input for speech synthesizer to drive for example diphone based synthesizer [15].

  Words like ਕਰਮ can be represented by phonemes as ਕ ਅ ਰ ਮ in Punjabi and word “cunning” has phoneme representation as ”\k\, \a\, \n\, \i\, \ng\” in English.

- **Diphones: -** A diphone is approximately the last half of one phone followed by first half of the next phone. E.g. for the word “Tapas” the diphone sequence is “t+a (ਤ+ਅ), a+p (ਅ+ਪ), p+a (ਪ+ਅ), a+s (ਅ+ਸ)” and phonemes are “\t\ (ਤ), \a\ (ਅ), \p\ (ਪ), \a\ (ਅ), \s\ (ਸ)”. Thus diphones present the transition from one phone to other and the boundaries of a diphone occur in the middle of phones. This means that the concatenation point will be in the most steady state region of the signal, which reduces the distortion from concatenation points. Another

advantage with diphones is that the co-articulation effect needs no more to be formulated as rules [15].

However, with diphones memory requirements increases and this makes data collection difficult. But still it is most suitable concatenation unit for TTS systems due to its advantages.

- **Triphones: -** These are longer concatenation units as compared to diphones and contains one phoneme between steady state point i.e. half phoneme-phoneme-half phoneme. Its structure is like L-X+R where L and R are left and right contexts (half phonemes) respectively, of phoneme X e.g. for word Kamya triphones generated are "k-aa+m (ਕ-ਅ+ਮ), aa-m+y (ਅ-ਮ+ਯ), m-y+aa (ਮ-ਯ+ਅ)". Thus triphones also consider left and right contexts of a phoneme thus preserving co-articulation effect of complete phone along with its starting and ending contexts.

1.5.3 HMM-BASED SYNTHESIS

This technique is used to synthesize speech with the help of HMM (Hidden Markov Model). It is the statistical model that includes markov models which can be used for modelling the speech parameters like fundamental frequency F0, Mel-cepstral coefficients etc. extracted from a speech database in the form of different phone acoustic models, and then regenerating the parameters according to text input for creating the speech waveform. HMM-based speech synthesis systems are able to produce speech in different speaking styles with different emotions.

If speech is synthesized from HMMs directly, it is feasible to synthesize speech with various voice characteristics by applying various speaker adaptation techniques developed in HMM-based speech recognition area. An HMM-based TTS system (HTS) is developed in which spectral parameter sequences are generated from HMMs directly based on maximum likelihood criterion using the re-estimated HMM phone acoustic models trained from feature parameters of speech database.

These systems have better adaptability, flexibility and smaller memory requirement. However, the HMM-based TTS systems have the drawback of less naturalness in the quality of speech generated as compared to concatenative based speech synthesizers

because speech waveforms are not directly concatenated to synthesize voice signal. In this spectral frequency and fundamental frequency are modelled and used to generate the speech waveform [18].

1.5.4 FORMANT SYNTHESIS

This approach of speech synthesis is based on the rules which describe the resonant frequencies of the vocal tract. The formant method uses the source-filter model of speech production, where speech is modeled by parameters of the filter model [17]. It models only the sound source and formant frequencies, not any physical characteristics of vocal tract like in articulatory approach.

According to the source-filter theory of speech production, speech signal can be represented in terms of source and filter characteristics [8]. In human speech production, the primary sound source is the excitation of the vibrating vocal folds that generates a rich harmonic spectrum, whose energy declines with increasing frequency. The vocal tract modifies the excitation spectrum through a transfer function with formants or anti-formants. Finally the sound radiates to the surrounding air at lips and nostrils. This causes a frequency dependent effect called lip radiation, which acts as a high-pass filter [21].

Rule-based formant synthesis can produce quality speech which sounds unnatural, since it is difficult to estimate the vocal tract model and source parameters [13].

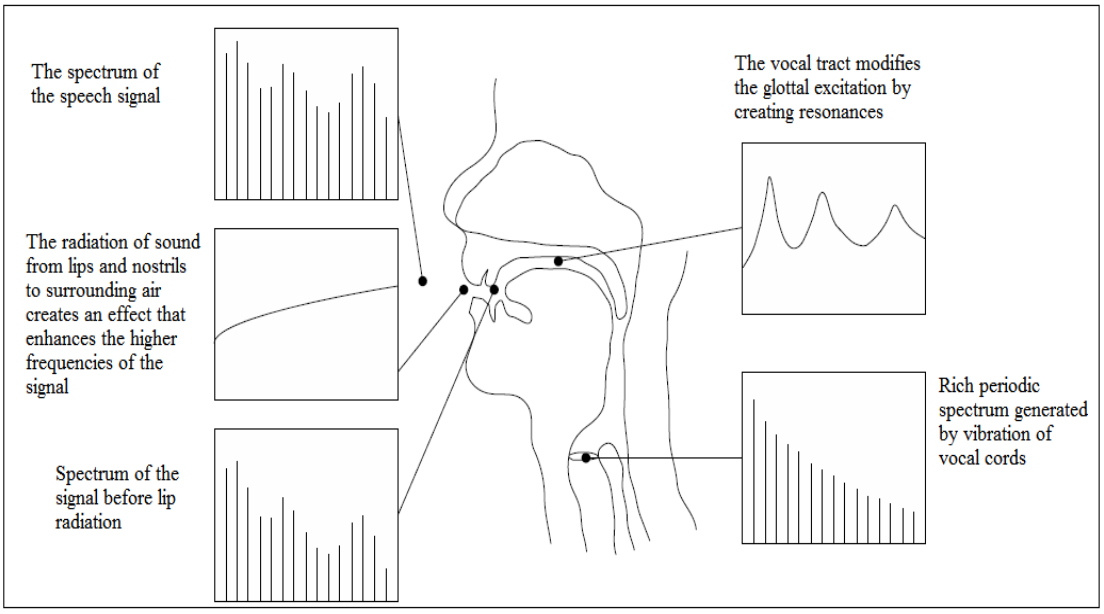


Fig. 1.4: Source Filter Theory

1.6 SIGNAL ANALYSIS

1.6.1 CEPSTRAL ANALYSIS

This analysis technique is very useful as it provides methodology for separating the excitation from the vocal tract shape. It is used for pitch (f0) detection [25].

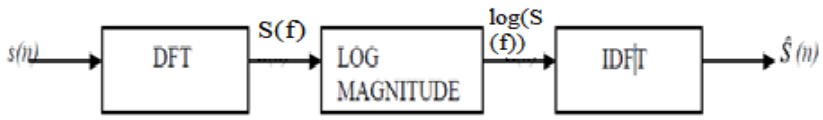


Fig. 1.5: CEPSTRAL ANALYSIS [11]

Following steps are followed in cepstral analysis:

- DFT (Discrete Fourier Transform) and Log Magnitude - Speech signal s(n) received for analysis after spectral shaping is given as input for cepstral analysis where it is transformed from time domain signal to frequency domain spectrum (i.e. S(f) is obtained) by using FFT (Fast Fourier Transform) Algorithm and log magnitude of amplitude is taken that transforms S(f) into log(S(f)).

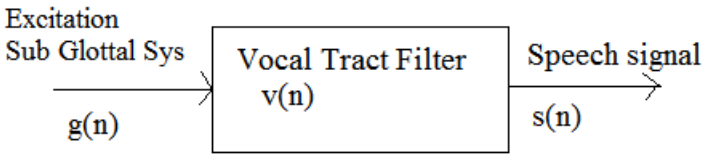


Fig. 1.6: Linear Acoustic model of Speech Production [11]

$$s(n)=g(n)*v(n)$$

Where *v(n):* vocal tract impulse response

g(n): excitation signal

Frequency Domain Representation after FFT

$$S(f) = G(f) \cdot V(f)$$

After taking log of all terms

$$\log(S(f)) = \log(G(f)) + \log(V(f))$$

- IDFT (Inverse DFT) - In order to obtain cepstrum from spectrum in frequency domain, IFFT (Inverse Fast Fourier Transform) is applied on the spectrum and cepstrum of signal ($\hat{S}(n)$) is obtained in pseudo-frequency region.

Spectrum in frequency domain consists of – spectral envelope and spectral details.

Spectral envelope is obtained by joining the peaks or formants (dominant frequency regions) of spectrum. In order to obtain this spectral envelope, the lower frequency region of cepstrum is extracted. E.g. in the following fig. spectral envelope of spectrum log X[k] is given by log H[K] and spectral details by log E[k] which is then converted to cepstrum x[k] by taking IFFT (Inverse FFT) of signal.

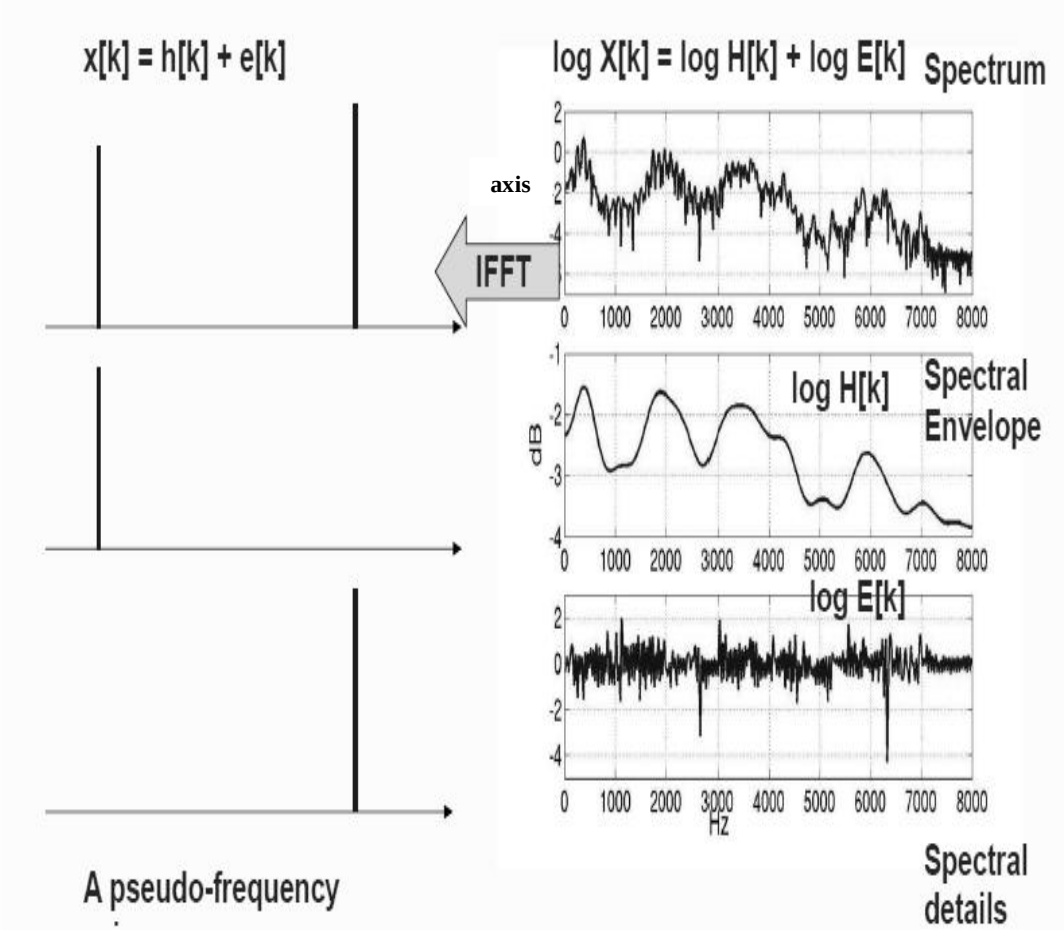


Fig. 1.7: Inverse Fast Fourier Transform

1.6.2 MEL-CEPSTRAL ANALYSIS

This technique is based on human perception experiments. Human ear concentrates only on certain frequency regions. It does not perceive all the frequencies present in spectral envelope; therefore, instead of performing cepstral analysis on spectrum, mel spectrum is analyzed [16].

Mel-spectrum is the human ear simulation part of spectrum. It is further analyzed in cepstral analysis to obtain mel-cepstrum. For obtaining Mel cepstrum:-

- The speech waveform *s(n)* is first windowed with analysis window *w(n)*
- Then its DFT *S (f)* is computed.
- The magnitude of *S (f)* is then weighted by a series of *Mel* filter frequency responses whose center frequencies and bandwidth roughly match those of auditory critical band filters.

$$\log X[k] = \log (\text{Mel-Spectrum}) \quad (4)$$

$$\log X[k] = \log H[k] + \log E[k] \quad (5)$$

$$x[k] = h[k] + e[k] \text{ (Taking IFFT)} \quad (6)$$

Cepstral coefficients h[k] == Mel-Frequency Cepstral Coeff. (MFCC)

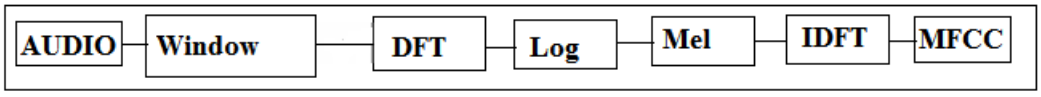


Fig. 1.8: Mel Cepstral Analysis [11]

1.7 HMM BASED SPEECH SYNTHESIS SYSTEM (HTS)

1.7.1 HMM (HIDDEN MARKOV MODEL)

It consists of finite set of states each of which is associated with the probability distribution. Only the outcome is visible to an external observer and therefore the states are 'hidden' to outsiders. HMM have three model parameters (A, B, π) that is there are finite number, say N, of states in HMM. At each time t, a new state is entered based on the transition probability distribution (A) which depends on previous

state. After each transition, an observation output symbol depends on the current state based on output probability distribution (B). (π) denotes initial state probability.

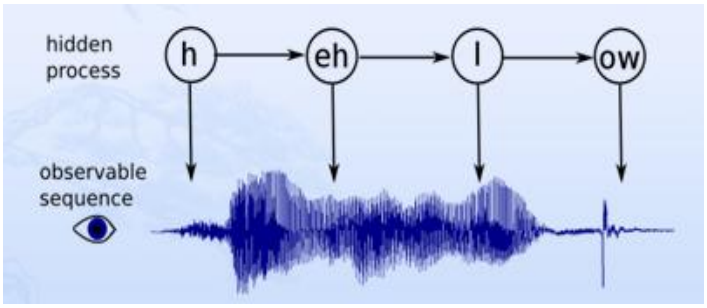


Fig. 1.9: HMM model with hidden states and visible output

Only the output of HMM in the form of sequence of feature parameters or speech signal is visible while the states containing the phone transcriptions are hidden from the observer.

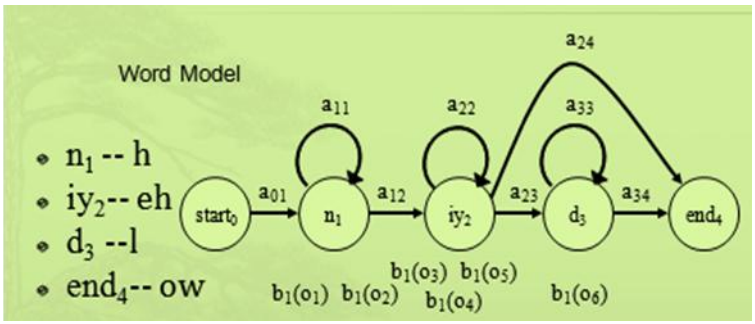


Fig. 1.10: HMM model for word "Hello"

In the above figure, various states of markov model are represented by phonemes viz. h, eh, l, ow of word Hello; a_{01}, a_{11} etc. represents the transition probability from states 0 to 1 and states 1 to 1 respectively and $b_1(o_1)$, $b_2(o_2)$ etc. represents output probabilities of o_1, o_2 respectively.

In order to put HMM-based TTS to work it is necessary to record a speech database which is eventually segmented and annotated with contextual information about syllable, word, phrase, and utterance that could possibly have influence in the proposed TTS [18].

The HMM-based speech synthesis technique comprises of two phases that are followed to generate speech signals as an output for the text input to the system as shown in the figure i.e.

- **Training part**

  It involves training of HMM models using a speech database and feature vectors extracted from it.

- **Synthesis part**

  It involves analysis of input text and synthesis of speech signals by regenerating the feature parameters using the trained acoustic context-dependent phone HMM models.

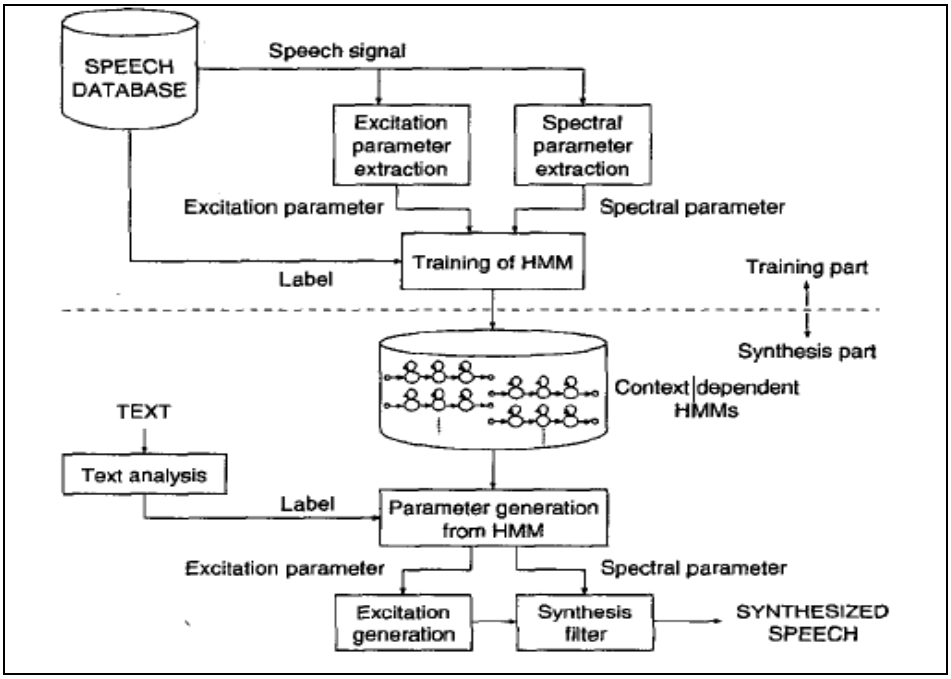


Fig. 1.11: Hmm Based TTS [25]

1.7.2 TRAINING PART OF HTS (HMM BASED TTS)

In HTS, firstly the speech signals from speech database are modelled to extract features parameters and then these features are used to train the HMM phone models to regenerate the feature vectors and synthesize the speech signal in the synthesis phase of the system. It consists of two phases:-

- Signal Modelling
- HMM Training (SETTING RULES FOR HMM)

1.7.2.1 SIGNAL MODELLING

Signal modeling represents process of converting speech signal into a set of feature parameters i.e. output vector that involve Mel-cepstral coefficients and fundamental frequency F0. It includes following tasks:-

- **Spectral Shaping**

  Spectral shaping is the process of converting the speech signal from analog signal to a digital signal through highest possible sampling rate; and emphasizing important frequency components in the signal in order to improve SNR(Signal-to-Noise Ratio) [11].

Fig. 1.12: Spectral Shaping

Spectral shaping involves two basic operations i.e. digitization which involves the conversion of analog speech signal from sound pressure wave to a digital signal which will further be used to extract features; and digital filtering in which important frequency components in the signal are being emphasized. The main aim of digitization is to represent the speech data signal into sampled data with high signal-to-noise ratio (SNR). After emphasizing frequency components in the signal a process called pre-emphasis is done mostly using a finite impulse response (FIR) filter. Pre-emphasis is mainly done to improve overall signal-to-noise ratio of the signal by increasing the magnitude of certain frequencies w.r.t. the magnitude of other frequencies in the signal.

Thus pre-emphasis is used to improve signal quality at the output of data transmission because while transmitting signals at high data rates, distortions may be introduced by the transmitting medium, so pre-emphasis is used to correct the distortion of transmitted signal. After the implementation of this

phase the signal thus obtained closely resembles the original or desired signal, allowing the use of higher frequencies or producing fewer bit errors.

- **Feature Extraction**

  Feature extraction involves analysis of speech signal obtained after spectral shaping i.e. after digitizing and pre-emphasizing it. It is the process of deriving a "compact and useful" representation from speech signal i.e. a process of obtaining different features. E.g. power, pitch, and vocal tract configuration from the speech signal [11].

  In feature extraction phase signal can be analyzed to extract features either through Temporal analysis that involves analysis of speech waveform itself or Spectral analysis that involves analysis of spectral representation of speech signal.

  In HTS spectral analysis of speech signal present in speech database is performed for extraction of spectral and excitation parameters.

  The spectrum part consists of mel-cepstral coefficient (MFCC) vector including the zeroth coefficients, their delta and delta-delta coefficients. (Mel-Cepstral Analysis) [18].

  The excitation part consists of log fundamental frequency (log F0), its delta and delta-delta coefficients [16].

1.7.2.2 SECOND PHASE OF HMM TRAINING

In this phase, the feature parameters of speech signal obtained in previous stages are used to train the system. The feature vectors viz. Mel-cepstral coefficients (MFCC) and fundamental frequency (F0) initialize the HMM acoustic models of context-independent monophones and context-dependent triphones and re-estimate them to train these phone models so that they can regenerate the spectral and excitation features in synthesis phase to synthesize the speech signal.

1.7.3 SYNTHESIS PART OF HTS

In the synthesis part of HTS, firstly the input text to be mapped into speech is converted to a context-based label sequence. Secondly, a sentence HMM is

constructed by concatenating context dependent HMMs obtained in training phase (by modeling acoustic phone models) according to the label sequence of the input text. State durations of the sentence HMM are determined so as to maximize the output probability of state durations, and then a sequence of mel-cepstral coefficients and log F0 values including voiced / unvoiced decisions is determined in such a way that its output probability for the HMM is maximized using the speech parameter generation algorithm [16]. Thus in the end with the help of these regenerated MFCC and F0 feature vectors speech is synthesized by the system.

1.8 PHONEME RECOGNITION

After mel-cepstral analysis of speech signal from speech database, mel-cepstral coefficients are obtained. These MFCC coefficients are then used in phoneme recognition phase to train HMM for particular state sequence q^.

In this phase, decoding problem of HMM is analyzed and solved using Viterbi Algorithm.

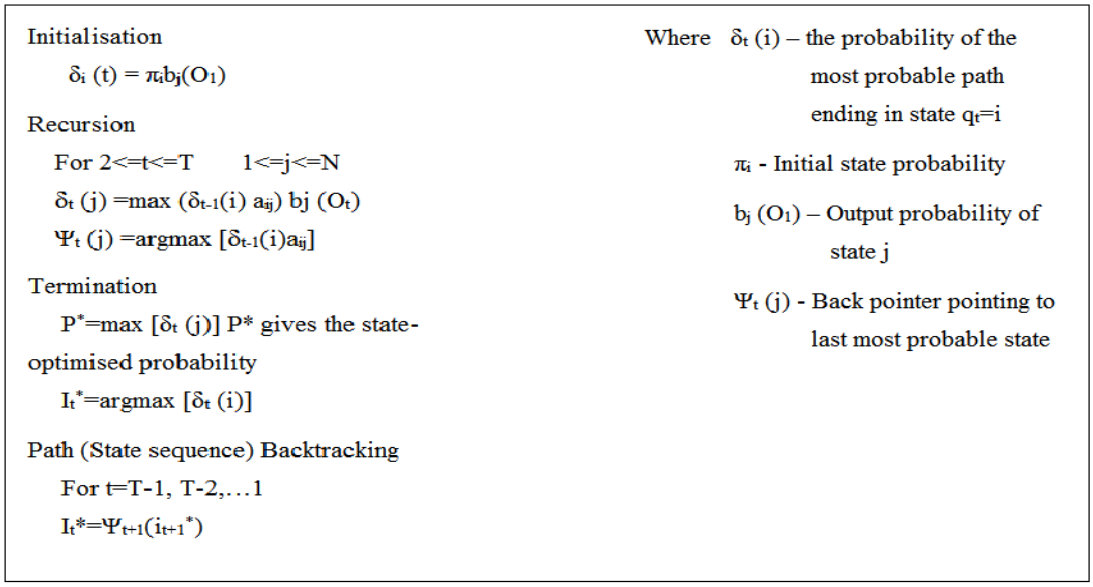


Fig. 1.13: Viterbi Algorithm

Decoding Problem-- Given the observation sequence $O = O_1, O_2, \ldots\ldots O_t$, how we choose a state sequence $i = i_1, i_2, \ldots i_T$ which is optimal in some meaningful sense [20].

Viterbi Algorithm: - The Viterbi algorithm provides a computationally efficient way of analyzing observations of HMMs to recapture the most likely underlying state sequence of words. The algorithm works by finding the most probable sequence of

phonemes (states) that could generate the output feature parameters of speech signal, according to the probabilities of the model.

Through this algorithm, the most probable state sequence of a particular phoneme is found that will produce given output sequence that was derived from cepstral and mel-cepstral analysis of speech signals. Here, partial probability of reaching a particular state is calculated by considering the partial probability of last state, transition probability and output probability. Thus, state sequence of maximum probability is found recursively. In this way the most probable state sequence q^ is obtained that generate the given output feature parameter vector O_i. Consequently, HMM phone models are trained for given output feature vectors using Viterbi Decoding.

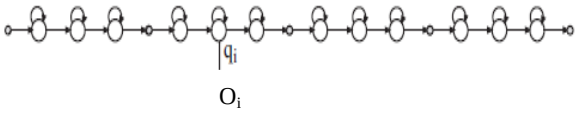


Fig. 1.14: Concatenated HMM Chain

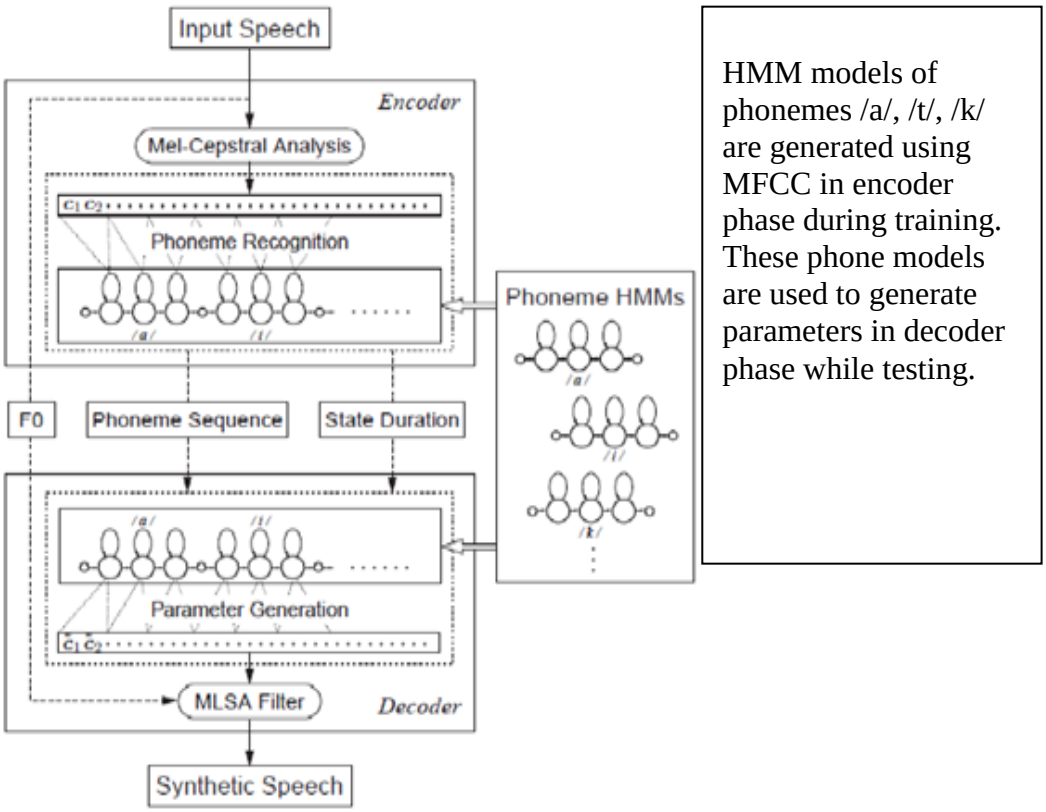


Fig. 1.15: Speech Synthesis System Based On HMM [18]

1.9 PARAMETER GENERATION

In this phase, parameters or features are regenerated in order to synthesize speech. After pre-processing the input text, graphemes or text syllables are processed to form phonemes. According to the input phonemes, corresponding phoneme HMM is selected and output parameters are generated according to the training of HMM.

In this phase, evaluation problem of HMM is analysed and solved using Forward Algorithm.

Evaluation Problem: - Given the observation sequence O = O_1, O_2... O_T, and model λ = (A, B, π), how we compute Pr (O| λ), the probability of the observation sequence [11].

Forward Algorithm: - This algorithm provides a way to efficiently find the probability of the observation sequence. It is used to recognize each word model based upon the given test observation sequence. After analyzing the phonemes and selecting the corresponding phoneme HMM, using Forward algorithm, the most probable output or observation sequence is selected.

Initialise:

$$\alpha_1(i) = \pi_i b_i(o_1)$$

Calculate:

$$\alpha_{i+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1})$$

Obtain:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Where π_i - Initial state probability

$b_j(O_1)$ – Output probability of state j

$\alpha_t(i)$ - probability of observing a partial sequence of observables $o_1,\ldots o_t$ such that at time t, state q_t=i

Fig.1.16: Forward Algorithm

Through this algorithm, given the particular phoneme HMMs the most probable output is generated i.e. the signal parameters which are further used for synthesizing speech in synthesis phase.

CHAPTER – 2

LITERATURE SURVEY

In order to develop a text-to-speech (TTS) system that produces a natural and intelligible, in short, high quality speech signals following literature was surveyed to obtain an overall idea of implementation of various approaches, different phases and their development for various languages by various researchers in the field of speech processing.

Rabiner and Juang [1986] presented the theory of markov models and markov chains and also illustrated their application for various speech recognition problems. They had given a detailed overview of hidden markov model and their problems of determining the observation sequence probability, state sequence and model parameters. Solutions to these problems were also being discussed with the help of the Forward-backward procedure, Viterbi algorithm and Baum-Welch re-estimation formulae. Further various issues regarding the types of HMMs, issues in implementation and extension and adaptation of basic HMM model to more advanced form had been presented. Finally the use of HMM in isolated word recognition was discussed.

Carlson [1993] discussed various approaches for generating the synthetic speech and presented some of the basic motivations for preference of one method over other synthesis method used. Initially, synthesis field was being analyzed according to flexibility and technical criteria and three main synthetic techniques were discussed mainly waveform coding, analysis-synthesis, synthesis by rule and accordingly an overview of speech synthesis models viz. Formant synthesis, Concatenative model and articulatory method was presented. Lastly various issues regarding various speaking characteristics and styles, accents, multilingual synthesis and speech quality were discussed.

Dutoit [1996] overviewed the various applications and uses of Text-To-Speech (TTS) synthesis system and explained the different components of synthesis system i.e. Natural Language Processing (NLP) component and Digital Signal Processing (DSP)

component. He highlighted the importance and implementation of various phases involved in NLP component viz. pre-processing, morphological analysis, prosody generation etc.. Further he presented different approaches of DSP component to synthesize speech signal on the basis of rules or concatenation of different speech units and factors related to unit selection. Finally he analyzed various existing TTS systems and specified various computational and economical constraints to be considered for designing good quality TTS systems.

Yoshimura *et al.* [1998] proposed a new approach to state duration modelling for HMM- based speech synthesis. They presented the modelling of a set of state durations of each phoneme HMM by a multi-dimensional Gaussian distribution and clustering of duration models using a decision tree based context clustering technique. For the development of HMM- based speech synthesis system in the training part, mel-cepstral coefficients were obtained and used to train context-dependent HMMs which were further clustered and re-estimated using embedded training. In the synthesis stage, state durations were determined by using the state duration models. Further, in the paper various contextual factors such as stress related factors and locational factors in addition to phone identity factors were being discussed. Lastly results of the experiments performed using ATR Japanese speech database were analyzed showing that good quality speech with natural timing could be synthesized and the speaking rate could be varied easily.

Lemmetty [1999] studied audiovisual speech synthesis project which was carried out at Helsinki University of Technology during 1998-2000. In the thesis he analyzed the present scenario of speech synthesis technology and discussed the potential methods for the project. Various techniques, methods, applications and products in the speech synthesis research area were presented instead of a single method, technology or technique which was usually presented in various articles, papers in speech synthesis area. The project focussed on developing an audiovisual speech synthesis of high quality mainly in Finnish. Various other issues like that of naturalness, platform independence and quality assessment were also analyzed along with different speech synthesizers, human speech production system and algorithms. He discussed the achievement of a synthesis system through adaptation and modularity characteristics instead of standalone synthesizers that do not share common parts and work on

independent platforms. Lastly, various evaluation methods and tests were studied and different testing levels and test methods were discussed.

Yoshimura *et al.* [1999] presented the modelling of spectrum, pitch and state duration models in a unified framework of HMM. They described that in order to model pitch and state duration models multispace probability distribution HMMs and multidimensional Gaussian distributions were required. In the training phase, they used and described in detail decision tree based context clustering technique for clustering the spectral parameters, pitch parameters and state duration distributions. In the synthesis phase, HMM based speech parameter generation algorithm and mel-cepstrum based vocoding techniques were discussed and analyzed for generation of synthetic speech signals. Lastly they performed various listening tests and experiments to verify that the speech signals produced by proposed speech synthesis system were more natural sounding and resemble the speech of speaker in ATR Japanese speech database used in training phase to train the system.

Donovan and Woodland [1999] discussed a new approach for speech synthesis that was based on concatenation of sub-phone units that were obtained by clustering context-dependent hidden Markov models using cross-word decision tree method and segmented by statistical methods. They used speech database in training for generating clustered states that represent various models trees etc. and TD-PSOLA waveform concatenation synthesizer to synthesize speech as clustered state sequence. Using this system, they generated natural and intelligible sound and performed Modified Rhyme Test on it to measure intelligibility as a result of which an error rate of 50% was obtained. This paper also presents that speech synthesized by the system was identical to that recorded in the training database and that system could be adapted for new voice in 48 hours.

Prudon and d'Alessandro [2001] discussed a text-to-speech synthesis system in French based on concatenation of speech segments from annotated speech databases. Further they presented and analyzed four selection criteria to find the optimal chain of phonemes. A comparative evaluation test was performed on proposed system proving it more efficient and natural sounding than the concatenative system that used diphones as concatenation unit and synthesized speech by various rules of prosody. Instead of realizing according to phonetic rules they realized prosodic facts as in

actual signal by describing speech at phone level using phonetic pruning. At last, they concluded that developed system produced pleasant voice and could be adapted for new voices and new speaking styles based on speech database but certain inconsistencies and ambiguities regarding prosodic units were found in the speech signal generated.

Tokuda *et al.* [2002] proposed a new speech synthesis system in English language. They developed a text-to-speech synthesis system that used HMM for synthesizing speech waveforms in English language using the architecture of Festival. This HMM based speech synthesis technique i.e. HTS was first implemented for Japanese language and they adapted it for English language in two phases. Firstly in training phase they extracted spectral and excitation parameters from speech database and decision tree based clustering technique was used for combining contextual factors and later in synthesis phase speech signal was synthesized from generated mel-cepstral coeff. And F0 values using MLSA filter. HTS was implemented on Festival architecture and modified version of HTK (HMM Toolkit) along with SPTK was used. Further, they compared unit-selection concatenative based and HTS approach of speech synthesis and concluded that though unit-selection method produced more natural sounding and high quality speech signal because speech waveforms were concatenated directly, HTS approach also produced good results. Moreover, HTS system was small in size with run time engine less than 1M bytes and could be implemented for different speech characteristics by various speaker adaptation techniques used in speech recognition. They also proposed that speech quality can be improved by mixed model based on MELP speech coder and post filtering.

Masuko [2002] discussed an approach to synthesize speech from text with the help of hidden markov model using maximum likelihood criterion instead of unit waveform concatenation techniques. They presented that natural sounding speech could be synthesized through HMMs by modeling static and dynamic parameters that, in turn, generated smooth spectral sequences to synthesize speech signals. They performed subjective experiments and investigations to show the importance of dynamic features and relation between model complexity and quality of synthesized speech. Instead of using discrete or continuous HMMs they used multi-space probability distribution HMMs (MSDHMM) to model observation sequence of F0 patterns so that one-dimensional continuous values and zero- dimensional discrete symbols can be

included along with spectral parameter sequence. In this thesis, they applied the MAP-VFS algorithm i.e. combination of maximum a posteriori (MAP) estimation and a vector field smoothing (VFS) technique, on the HMM-based TTS system and performed ABX listening tests on four target speakers (two males and two females) that verified that speech samples synthesized from adapted models were 88% closer to target speakers' models using only one adaptation sentences from each target speaker. It was also presented that HMM based speech synthesis system can be used in speaker verification systems and proved that false acceptance rates for synthetic speech were 63% by training the system using only one training sentence for each customer of the speaker verification system. Lastly they also analyzed a HMM- based vocoder and presented that its performance at 340 bit/s was comparable to a multi-stage VQ based vocoder at about 2200 bit/s without F0 and gain quantization for both coders.

Basu *et al*. [2003] discussed the importance of speech synthesis systems for visually impaired community in India and presented the development, technology and various features of an Audio Qwerty Editor, KGP-Talk i.e. a talking web browser and a text to speech synthesis system for Indian Languages. For the development of TTS system they used concatenative approach that concatenated signal units or specific waveforms according to certain rules and unit of concatenation selected was partnemes. Partnemes were the waveforms which were transitory from a consonant to a vowel (CV), from a vowel to a consonant (VC) or only consonant or only vowel. The Audio Qwerty Editor developed works like text editor to create, edit and open various documents along with an audio interface useful for visually handicapped people who can easily and efficiently perform editing functions with audio assistance. Lastly KGP-Talk was discussed that includes features like a talking Chat Messenger, a talking mail client, talking net sender, talking printing system etc.. It helps sightless people to browse internet by reading out texts, hyperlinks, links etc. TTS systems discussed in this paper were implemented in Hindi and Bengali language but can be extended for other Indian languages.

Kesarkar [2003] analyzed signal modeling approach for speech recognition. For signal modeling i.e. to convert speech signal into set of parameters basic operations viz. spectral shaping and feature extraction were being studied in detail in the report. For feature extraction both temporal analysis and spectral analysis techniques were studied. He presented Critical Band Filter Bank analysis for transduction in human

auditory system, Cepstral analysis for separating the excitation from the vocal tract shape, Mel- Cepstral analysis for extracting Mel-Cepstral coefficients, Linear Predictive Coding analysis for approximating speech as linear combination of past speech samples and Perceptually Based Linear Predictive Analysis (PLP). Further he discussed the basis and usage of Power Estimation, Fundamental Frequency Estimation, Gold and Rabiner algorithm and Cepstrum based pitch determination. Lastly, all these feature extraction techniques were being discussed and their various advantages, usage and drawbacks were being concluded by the author.

Maia *et al.* [2003] developed a text-to-speech synthesis system from Hidden Markov models for Brazilian Portuguese and presented its description. For HMM based speech synthesis they recorded speech database and segmented it phonetically using HMM Tool Kit (HTK). Further database was used to extract spectral and excitation parameters in training phase which were regenerated in synthesis phase to synthesize the speech signals. Contextual information of phoneme, syllable, word, phrase and utterance was determined and decision tree based clustering was used to cluster phonemes based on certain user-defined questions. They trained the system for 80 utterances and evaluated F0 pattern comparisons between natural uttered sentence and synthesized sentence which shows close resemblance in them and also concluded that phoneme contexts were important for spectrum syllable, word and phrase information mainly affect *F*0 and state durations.

Blunsom [2004] summarized the basics of markov processes and Hidden Markov Models (HMM) and applications of HMM in various fields like signal processing, speech processing, various NLP tasks like that of Part-of-speech tagging during morphological analysis, noun-phrase chunking etc.. He analyzed the problems of Evaluation wherein the probability of the observation sequence given a HMM models needed to be computed, Decoding wherein the most probable hidden state sequence given a observation sequence was to be discovered and Learning wherein the model parameters that best described the process were to be estimated and discussed their solutions by implementing the Forward algorithm, Viterbi algorithm and Baum-Welch re-estimation algorithm, respectively. He also presented the problems related with the use of these algorithms i.e. Viterbi underflow and Forward algorithm underflow and suggested their solutions.

Zen and Toda [2005] introduced a speech synthesis system based on Hidden Markov Model (HMM) developed in Nagoya Institute of Technology (Nitech-HTS) for a competition of text-to-speech synthesis systems using the same speech databases, named Blizzard Challenge 2005. They gave an overview of training and synthesis phases of basic HMM-based speech synthesis system and presented the developments in recent speech synthesis approaches such as STRAIGHT-based vocoding and hidden semi-Markov model (HSMM) based acoustic modeling. They also illustrated the generation of parameters for speech synthesis using global variance and verified that constructed voices could synthesize speech around 0.3 xRT (real time ratio) and their footprints were less than 2 MB. Finally they performed listening test on the proposed system and concluded that performance of system was much better.

Borkar [2006] presented the development of a system that synthesize speech signals for Konkani language using concatenative approach on VB programming language platform with the help of database containing more than thousand words that were used as concatenative unit to synthesize speech. He presented the analysis of Konkani language script. For the development of TTS, he performed noise (unwanted signal) removal from the recorded speech signals and stored the sound files using their ASCII values which helped to recognize the file in database during synthesis. In case of complex words (called Jod- Akshar) he used database Concatenation technique.

Martincic-Ipsic and Ipsic [2006] presented the development of HMM based speech synthesis system for Croatian language. They discussed various approaches of speech synthesis viz. articulatory synthesis, Formant model, concatenative approach and another method that includes generalization of concatenative method and dynamic selection of speech units from corpus called corpus based synthesis. They presented the source filter theory for speech signal generation and basis of hidden markov model along with viterbi approximation for feature generation. In training phase they used Croatian speech database containing 1111 speech utterances of speaker of national radio and phonetically segmented it using monophones speech recognizer that provided correctness of 78.62%. 41 monophones models were trained and converted to context-dependent triphones models based on 216 Croatian phonetic rules using decision tree based clustering to synthesize speech. System was tested on 41 Croatian sentences and pitch and spectrogram of signals were presented. Lastly they concluded that system produced vocoded buzzy speech but could be understood

and in order to improve phone models more speech material needed to be recorded and annotated.

Singh and Lehal [2006] presented the development of TTS system based on concatenative approach for Punjabi language written in Gurumukhi script. Syllables were used as basic concatenative unit and the input text after pre-processing was syllabified using an automatic syllabification algorithm based on grammatical rules of Punjabi language. Punjabi speech database used contained thirty three hundred syllables out of about ninety three hundred valid Punjabi syllables and labeled wave files of syllable sound positions using Sonic Foundry Sound Forge 5.0b. Lastly it was concluded that Punjabi speech synthesized by proposed system was of good quality but needed improvement in the part where smoothness at concatenation point of syllables was involved. They also concluded that the quality of speech produced by concatenative TTS majorly was dependent on precision in labeling the recorded sound files.

Shirbahadurkar and Bormane [2009] selected different units of concatenation viz. phonemes and words for generating speech signal. They analyzed and compared the speech signal synthesized by proposed system with the signal synthesized by conventional system using only phonemes as concatenation unit and found that quality of speech produced was better in terms of naturalness and intelligibility as compared to that produced by latter one. They used database containing words, syllables and phonemes for generating speech and Sound Forge software to segment words into syllables and phonemes automatically. They performed subjective performance evaluations and paired comparison test on the speech produced by system and verified that 81% of synthesized speech by proposed method was more natural than speech synthesized by conventional method and error rate of system was 8.22%. Lastly they concluded that word units were better than phoneme units for speech synthesizers for languages like Marathi.

Kaur and Singh [2010] proposed a new method to segment Punjabi speech signals into acoustic sub-units e.g. words, syllables, phonemes etc. which were further useful in speech recognition, speech synthesis and other fields of signal modeling. Further they presented that syllable was the best unit of segmentation as compared to words that couldnot be generalized and phoneme models that exhibit overgeneralization.

Thus in order to segment speech signals into syllables a novel approach was being discussed in which either the speech signal to be segmented was compared to a reference waveform already segmented or generated from acoustic units or segmented automatically using MATLAB as compared to the conventional method in which signals were segmented manually by analyzing their waveform of spectrogram, pitch etc. using graphical software displays that was prone to more errors and was tedious. They segmented signal into syllables by defining syllable boundaries in terms of certain threshold values and analyzing the signal for different threshold range. Lastly they made comparison studies between the proposed and traditional method and concluded that new method was less tedious and provide more accurate boundaries of syllables.

Balyan *et al*. [2011] in their paper "Development of Hindi Speech Synthesizer for Metro Rail Information System" discussed the development of a speech synthesizer that converts text to speech signals based on concatenative approach of speech synthesis. In the introduced TTS they used phoneme as basic unit of concatenation instead of syllable because of large domain synthesis, to synthesize speech signals for Hindi language that was used for Metro Rail Passenger Information System (MRPIS). Speech database of 292 sentences in Hindi was recorded and Hidden Markov Model based Technique was used to segment the speech sentences into phonemes by extracting mel-cepstral coefficients of speech signals and using them to train HMM acoustic models for each phoneme following Viterbi algorithm. Further they evaluated the performance of segmentation in terms of deviation and range of tolerance of error and verified that speech synthesized by automatically segmented monophones was more intelligible and natural than obtained by manually segmented labels. Lastly they concluded that larger number of Hindi sentences needed to be recorded and segmented to overcome the problem of some missing phonemes like uu, d, r etc. from synthesized speech.

Kumar and Aggarwal [2011] discussed the area of speech recognition i.e. to convert speech signals to text and presented the development of a speech recognizer based on Hidden Markov Model (HMM) for Hindi language. The system was implemented using HTK (HMM Tool Kit) on Linux platform. In the training phase a speech database of 30 Hindi words was recorded by eight speakers at room environment and HTK was used to extract the Mel frequency cepstral coefficients and train the HMM

acoustic word models by re-estimating them. Recognition was based on these re-estimated acoustic word models. Lastly the system performance was evaluated with help of five speakers uttering each word at least once and accuracy of 94.63% and word error rate of 5.37% was obtained.

King [2011] presented statistical parametric approach of speech synthesis by using Hidden Markov Models in a non-mathematical way. He summarized the speech synthesis approach from linguistic specification to waveform generation phase and also discussed other systems like exemplar based systems and model based systems. Further he answered various questions related to statistical parametric speech synthesis approach.

CHAPTER -3

PROBLEM STATEMENT AND PROPOSED SOLUTION

3.1 PROBLEM STATEMENT

Speech signals in local languages for uneducated people and voice in itself for sightless people makes their communication with machines i.e. either computers, mobiles or other gadgets very easy and efficient

For this, high quality TTS (Text-to-Speech) synthesis systems should be designed and implemented so that the speech signals generated by them are easily and correctly understood by people i.e. intelligible speech and also the signals should be natural sounding like the one produced by humans.

Various approaches are being followed to develop TTS systems in various languages like Hindi, English, Portuguese, Croatian, Finnish, and Punjabi etc. with their own drawbacks like in

Articulatory synthesis wherein human speech production system and articulatory organs are being tried to model directly but is very difficult due to high complexity and lack of 3-D information.

Formant synthesis which is a rule-based approach and tries to model the frequencies of vocal tract but unfortunately, produces unnatural sounds because of difficulty in estimation of the vocal tract model exactly.

Concatenative approach that synthesize speech by directly concatenating speech units like words, syllables, phonemes etc. is being extensively used for developing TTS systems for languages like English, Hindi, Finnish etc. but has disadvantage of high memory requirements and less flexibility for different speaking styles.

The most flexible and adaptive approach is HMM based speech synthesis systems (HTS) that use hidden markov models and speech spectral and excitation parameters to generate speech and adapt it using various adaptation techniques.

Till now this approach is not being used to synthesize speech signals in Punjabi language i.e. to map English text in Punjabi voice signals for either sightless or sighted people that understand Punjabi language instead of English due to which they are unable to interact with mobile phones, computers etc..

To develop TTS system for mobile phones in which caller's identity stored in English language will be uttered in Punjabi language in maximum possible intelligible and natural form, HMM based approach is used to segment speech database into smaller concatenation units viz. phonemes. Further, Concatenative approach is used to concatenate generated phonemes to form more natural speech sound in Punjabi language that can be easily understood and interpreted by illiterate and sightless people.

Thus the main objective of this thesis is to generate correct sequence of phonemes from all possible options using hidden markov models corresponding to the text to generate Punjabi speech. While mapping English text into Punjabi language phonemes, problems in pronunciation or phoneme sequence of certain words like Tony with phoneme sequence \ਤ\, \ੋ\, \ਨ\, \ਇ\ or \ਟ\, \ੋ\, \ਨ\, \ਇ\ also arise. Words having phonemes like

- ਤ or ਟ like in word "Tanya", for correct pronunciation \t\ represents \ਤ\ while in "Tony" \t\ represents \ਟ\
- ਅ or ਆ like in word "Jaya", Punjabi phoneme corresponding to letter 'a' in second position should be \ਅ\ while in "Maya", 'a' represents \ਆ\
- ਧ or ਢ e.g. in word "Dhani", phoneme \dh\ should be mapped into \ਧ\ while \dh\ in "Dholakiya" represents \ਢ\
- ਥ or ਠ like in word "Thakur", \th\ represent \ਠ\ while \th\ in "Martha" should be mapped into \ਥ\

Have ambiguous pronunciations in different languages.

3.2 PROPOSED SOLUTION

TTS (text to speech) synthesizer systems as discussed above constitute an important area of research. In order to develop a TTS system for Punjabi language and produce correct phoneme sequences of words with ambiguous pronunciations HTK (Hidden Markov Model Tool kit) is used. Segmentation of recorded speech database is done using the various phases (Data Preparation, Training and Testing) of this toolkit with the help of Hidden Markov Models.

While developing this TTS system, problems regarding pronunciation of words containing ਅ, ਆ and ਤ, ਟ is considered that whether in a particular word either ਤ or ਟ is uttered out of four problems stated earlier. E.g. in word ਟੋਨਿ the correct phoneme sequence among ਟ, ੋ, ਨ, ਇ and ਤ, ੋ, ਨ, ਇ is ਟ, ੋ, ਨ, ਇ and for word ਮਾਨਸ the correct phoneme sequence among ਮ, ਆ, ਨ, ਸ and ਮ, ਅ, ਨ, ਸ is ਮ, ਆ, ਨ, ਸ. These problems regarding the choice of correct phoneme sequence is being solved through this system to a greater extent with the help of HTK that uses modeling of acoustic models through HMM and spectral features like MFCC. In addition to this, rule based approach is used for producing the phoneme sequence of the words that are not correctly segmented by HTK.

3.3 INTRODUCTION TO HTK (HMM TOOL KIT)

The Hidden Markov Model Toolkit (HTK) is a portable toolkit for building and manipulating hidden Markov models on windows based systems. HTK is mainly used for speech recognition research although it has been used for numerous other applications including research into speech synthesis.

HTK consists of a set of library modules and tools like HVite, HHEd etc. available in C source form. These tools and modules provide functionality for speech analysis, HMM training, testing and results analysis, thus are helpful for speech processing whether it is speech recognition or synthesis. The software supports HMMs using both continuous density mixture Gaussians and discrete distributions and can be used to build complex HMM systems that further help to train acoustic models and use them for testing and decoding. The HTK release contains extensive documentation and examples e.g. HTK Book for HTK version 3.4 (by Young *et al*.) that helps users

to implement and use HTK easily and efficiently through a series of steps of training and testing in speech analysis research area.

Various Pre-requisites used for installing HTK and using HTK tools:-

- Operating System: Windows XP (32/64-bit)
- Environment Variable: System Path variable is set as “Install folder Physical Path\htk\bin.win32”.
- Perl Edition: ActivePerl 5.14.2 Build 1402(32/64-bit)

HTK tools are designed to run with a traditional command-line interface i.e. in the DOS mode i.e Command Prompt. Each tool in HTK is used for a specific purpose and has a number of required arguments along with some optional arguments. These optional arguments are always prefixed by a minus sign during the execution of the tool. E.g. Consider an HTK tool HInit used to initialize the prototype model in following command

```
HInit -T 1 -C Config.conf -n -S codetr.scp –m 4 hmm0 monophones0
```

This tool has two main arguments called hmm0 and monophones0 plus four optional arguments viz. T, C, S, m. Options are always introduced by a single letter option name preceded by a minus ‘-‘ sign and followed, where appropriate, by the option value. The option value is always separated from the option name by a space. Thus, the value of the –m (minimum segments) option is an integer number ‘4’, the value of the –T (Trace option) option is an integer number ‘1’ and the value of the –S option (Script file name) is a string ‘codetr.scp’. The -n option has no following value and it is used as a simple flag to enable or disable some feature of the tool. Options whose names are a capital letter have the same meaning across all tools viz. T, C, S. For example, the -T option is always used to control the trace output of a HTK tool and -S for script file name.

In addition to command line arguments, the operation of a tool can be controlled by parameters stored in a configuration file. For example, if the command

```
HInit -T 1 -C Config.conf -n -S codetr.scp –m 4 hmm0 monophones0
```

is executed, the tool HInit will load the parameters stored in the configuration file Config.conf during its initialization procedures. The main use of configuration files is to control the detailed behavior of the library modules on which all HTK tools depend.

The main advantage of command-line working as compared to modern graphical user interfaces is controlling the HTK tool execution through scripts and providing records and documentation of details of system construction or experimental procedure.

The summary of the command line and options for any HTK tool can be obtained simply by executing the tool with no arguments. E.g.

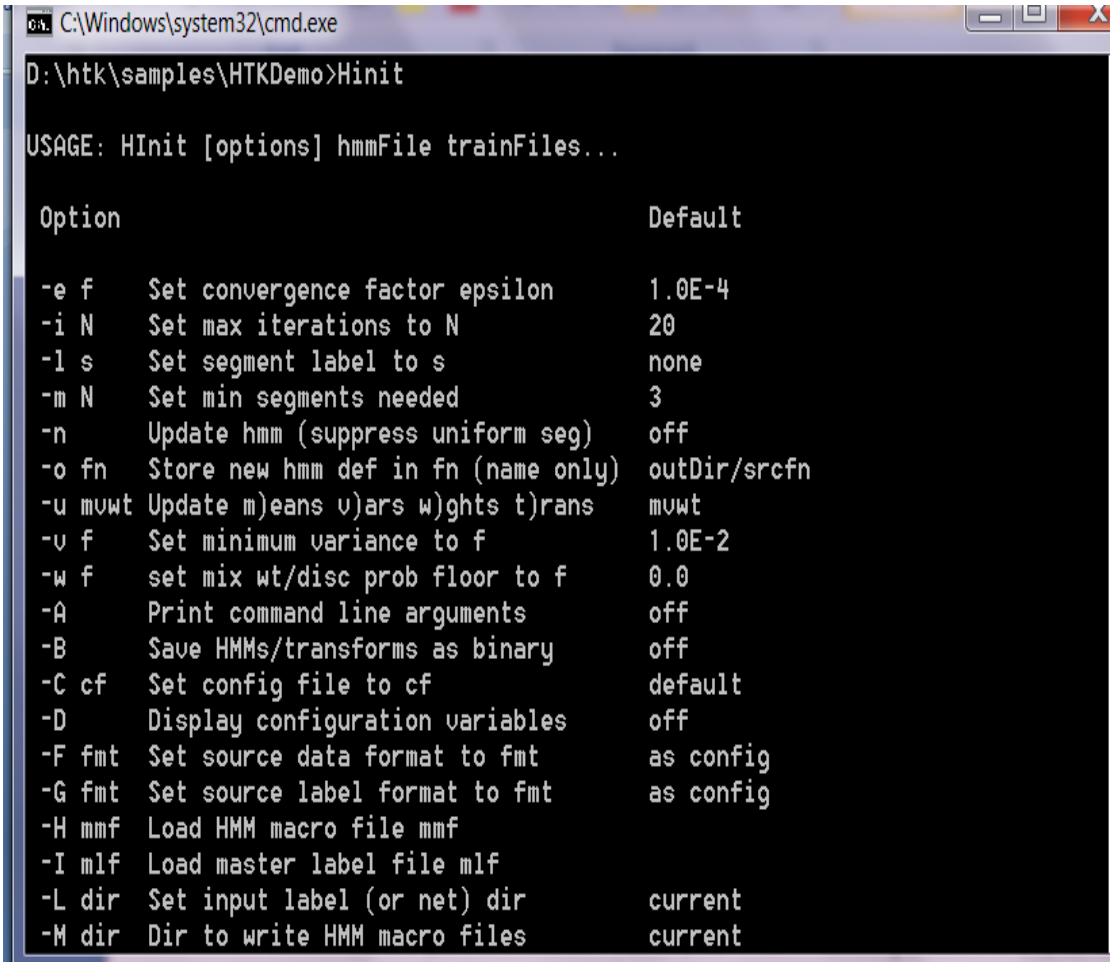


Fig. 3.1: Summary of Options of Tool HInit

3.4 DATA PREPARATION AND FEATURE EXTRACTION USING HTK (HMM TOOLKIT)

3.4.1 BASIC HTK TOOLS

In order to build a set of HMMs, in this phase, a set of speech data files and their associated transcriptions are required that are obtained by using either an already prepared speech database or by recording one using certain sound recording software viz. Audacity, Sound Forge etc. after considering particular settings.

Before this speech database can be used in training, it must be converted into the appropriate parametric form of feature vectors and any associated transcriptions must be modified so that they have the correct format and make use of the required phone or word labels. In HTK, if the speech needs to be recorded, then the tool **HSLab** can be used both to record the speech and to manually annotate it with any required transcriptions either at phone level or word level. By setting the appropriate configuration variables, all input files can be converted to parametric form as they are read-in.

The tool **HList** can be used to check the contents of any speech file. Transcriptions will also need processing as the original source transcriptions will not be in the form as required. The tool **HLEd** is a script-driven label editor which is designed to make the required transformations to label files. The HTK tool **HDMan** can be used to prepare a pronunciation dictionary from one or more sources. It reads in a list of editing commands from a script file and then outputs an edited and merged copy of one or more dictionaries.

Finally in data preparation, **HLStats** gather and display statistics on label files and where required.

3.4.2 IMPLEMENTATION

The training of HMM models and testing of speech synthesis system requires speech utterances and their phone level transcriptions. Data words used in both training and testing phases are initially pre-processed for various initials, titles, abbreviations, numbers etc. and then the refined data is passed as input to the training phase of HTK.

- **Developing The Pronunciation Dictionary**

HTK in order to compile the speech audio database and transcriptions into an Acoustic Model requires a phonetically balanced Pronunciation Dictionary. To develop a dictionary following steps were followed:-

Step -1: A list of words that are needed to be recorded in next step are listed in a file called **prompts.txt** file. Initially in the first phase we have considered recording data that consists of 61 words (i.e. words containing letter ਤ and ਟ) that are arranged in 17 samples included in the prompts.txt file and 81 words (i.e. words containing letter ਅ or ਆ) that are arranged in 23 samples included in the prompts1.txt file. The reason of considering said words speech database is to test run on a smaller size speech database which can further be extended to larger database. The first column of the prompts file contains the name of the audio file to be created, and the following columns contain the text transcriptions of what to be recorded in the audio file.

```
*/sample1 TONY TANVI TINKU TANYA
*/sample2 TEENA TEHAL TRISHA TRIPTI
*/sample3 TOMAR TISHA TULSI TIYA
*/sample4 TANU TANUJA TEJINDER TRISHNA
*/sample5 TRIPTI TAPESH TAPAN TARAN
*/sample6 TARA TILAK TUSHAR SAMRAT
*/sample7 TARAK TANUSH TANUJ TAKSHAK
*/sample8 TAKSHILA TAKSHEEL TILAK TANMAY
*/sample9 VATIKA TAPAS TANISH
*/sample10 TANUKA TATINI TITHI
*/sample11 TEJASVI TULSIDAS TAYAGRAJ TUKARAM
*/sample12 TAMANNA TAJ
*/sample13 TANISHA TAVISHI TARPANA
*/sample14 TRISHNA TAPASYA BHATNAGAR
*/sample15 TARANJOT TEJASWINI TEJAL BHATTACHARYA
*/sample16 TOYA TEJAS TUNGANATH
*/sample17 TARACHAND TANISHQ TRUSHA BHATT
```

Fig. 3.2: (prompts.txt) For Words Containing ਤ or ਟ

```
*/sample1 KARAN PRANAV CHIRAG TARUN
*/sample2 BHAWAN YAMINI MADHURI JASPAL
*/sample3 NARAYAN KUNAL KALYAN
*/sample4 RAKESH SADHNA
*/sample5 RUKSANA RUPALI JAWAHAR DEEPALI
*/sample6 RAMAN DHVANI MANI VANI
*/sample7 JANVI RATNAKAR MANAV MANSI
*/sample8 BHARAT MANYA VARUN
*/sample9 AMITABH MADHAVI KAMYA
*/sample10 ASHA VIKRANT MALINI KASANA
*/sample11 CHARU PRABHAKAR PARAS DHARIKA
*/sample12 AALAP BALAJI BALRAM
*/sample13 ANAND BALGOPAL SHYAM
*/sample14 SAHIL PRAKASH SADHIKA RAM OMKAR
*/sample15 KHAN RAVI RAJNI JAYA
*/sample16 RATNA AMAR MANAS NAGARJUN
*/sample17 MALA BHAKTI BHAVANI MALTI
*/sample18 PRABHAT GAYATRI GANDHI
*/sample19 FALGUNI BIPASHA BALAKRISHNA PARAG
*/sample20 AAKAR MAYUR GANDHARI RADHA
*/sample21 KALI DIPANKAR
*/sample22 DHARMANAND DIVAKAR BALAKUMAR
*/sample23 DEVANAND PANNALAL SAYA
```

Fig. 3.3: (prompts1.txt) File For Words Containing ઋ or ૠ

Step -2: The HTK Perl script **prompts2wlist** is used to convert prompts.txt file into word list file called **wlist** containing all the words in prompts.txt file in alphabetical order using the following command and later SENT-END and SENT-START are also added in the file. These are HTK internal entries required for creation of the Acoustic Model of "sil".

```
C:\htk> perl../HTK_scripts/prompts2wlist prompts wlist
```

```
AAKAR
AALAP
AMAR
AMITABH
ANAND
ASHA
BALAJI
BALAKRISHNA
BALAKUMAR
BALGOPAL
BALRAM
BHAKTI
BHARAT
BHAVANI
BHAWAN
BIPASHA
CHARU
CHIRAG
DEEPALI
DEVANAND
DHARIKA
DHARMANAND
DHVANI
DIPANKAR ------
```

Fig. 3.4: Word List (wlist1) For Words With ઋ or ૠ

```
BHATNAGAR
BHATT
BHATTACHARYA
SAMRAT
SENT-END
SENT-START
TAJ
TAKSHAK
TAKSHEEL
TAKSHILA
TAMANNA
TANISH
TANISHA
TANISHQ
TANMAY
TANU
TANUJ
TANUJA
TANUKA
TANUSH
TANVI
TANYA
TAPAN
TAPAS
TAPASYA
TAPESH
```

Fig. 3.5: Word List (wlist) For Words Containing ਤ or ਟ

Step -3: In this dictionary is created using HDMan command that go through the wlist file, and look up the pronunciation for each word in a separate lexicon file **list.txt**, and output the result in a **Pronunciation Dictionary (dict).** A script called global.ded is also used to convert all the words in dict file to uppercase.

```
HDMan -A -D -T 1 -m -w wlist -n monophones1 -i -l
dlog dict.../lexicon/list.txt
```

This creates dict file i.e. the pronunciation dictionary, monophones1 that contain list of phones used in dict and dlog which is a log file.

```
AAKAR        [AAKAR]        aa k aa r sp
AALAP        [AALAP]        aa l aa p sp
AMAR         [AMAR]         a m a r sp
AMITABH      [AMITABH]      a m i tt aa b sp
ANAND        [ANAND]        aa n a n d sp
ASHA         [ASHA]         aa sh aa sp
BALAJI       [BALAJI]       b aa l aa j i sp
BALAKRISHNA  [BALAKRISHNA]  b aa l a k r i sh n aa sp
BALAKUMAR    [BALAKUMAR]    b aa l a k u m aa r sp
BALGOPAL     [BALGOPAL]     b aa l a g o p aa l sp
BALRAM       [BALRAM]       b a l r aa m sp
BHAKTI       [BHAKTI]       bh a k tt i sp
BHARAT       [BHARAT]       bh a r a tt sp
BHAVANI      [BHAVANI]      bh a v aa n i sp
BHAWAN       [BHAWAN]       bh a v a n sp
BIPASHA      [BIPASHA]      b i p aa sh aa sp
CHARU        [CHARU]        ch aa r u sp
```

Fig. 3.6: Pronunciation Dictionary of Words Containing ਅ or ਆ

```
BHATNAGAR     [BHATNAGAR]     bh a t n aa g a r sp
BHATT         [BHATT]         bh a t sp
BHATTACHARYA  [BHATTACHARYA]  bh a t aa ch aa r y aa sp
SAMRAT        [SAMRAT]        s a m r aa t sp
SENT-END      []              sil
SENT-START    []              sil
TAJ           [TAJ]           tt aa j sp
TAKSHAK       [TAKSHAK]       tt a k sh a k sp
TAKSHEEL      [TAKSHEEL]      tt a k sh ee l sp
TAKSHILA      [TAKSHILA]      tt a k sh i l aa sp
TAMANNA       [TAMANNA]       tt a m a n aa sp
TANISH        [TANISH]        tt a n i sh sp
TANISHA       [TANISHA]       tt a n i sh aa sp
TANISHQ       [TANISHQ]       tt a n i sh k sp
TANMAY        [TANMAY]        tt a n m e sp
TANU          [TANU]          tt a n u sp
TANUJ         [TANUJ]         tt a n u j sp
TANUJA        [TANUJA]        tt a n u j aa sp
```

Fig. 3.7: Pronunciation Dictionary For Words Containing ञ or ट

```
bh
a
t
n
aa
g
r
sp
ch
y
s
m
sil
tt
j
k
sh
ee
l
i
e
u
v
```

Fig. 3.8: (monophones1) File Containing Phones

- **Recording The Data**

  Punjabi speech corpora used for training the system contains speech utterances of one female speaker. The data is recorded using microphones at room environment. Distance between speaker's mouth and microphone is approximately 5-7 cm. All the samples contained in prompts.txt file are recorded using Power Sound Editor at

  --- Sampling rate of 8000 Hz

  --- 16 bits bit depth and mono channel

  --- Recorded speech files are stored in .wav format.

  Files recorded are like **sample1.wav, sample2.wav** etc.

- **Creation of Transcription Files**

  For training HMMs each recorded sample need to have corresponding word and phone level transcriptions. This is done using following steps:-

  Step -1: HTK toolkit cannot process prompts file directly, it is required to be converted to a lab file or Master Label File (MLF). So a Master Label File (MLF) called **words.mlf** is created, which is a single file that contains a label entry for each line in the prompts file using the HTK script **prompts2mlf**.

```
C:\htk> perl../HTK_scripts/prompts2mlf words.mlf prompts
```

```
#!MLF!#
"*/sample1.lab"
TONY
TANVI
TINKU
TANYA
.
"*/sample2.lab"
TEENA
TEHAL
TRISHA
TRIPTI
.
"*/sample3.lab"
TOMAR
TISHA
TULSI
TIYA
.
```

Fig. 3.9: Word Transcription File (words.mlf)

Step -2: HLEd command is used to expand the Word Level Transcriptions i.e. words.mlf to Phone Level Transcriptions and each word is replaced with its phonemes, and the results are stored in a new Phone Level Master Label File called **phones0.mlf**. For this each word in the MLF file is reviewed, and the phones that make up that word in the dict file created earlier are looked up, and the result are outputted in a file called phones0.mlf using the edit script mkphones0.led.

```
C:\htk> HLEd -A -D -T 1 -l '*' -d dict -i phones0.mlf
mkphones0.led words.mlf
```

phones1.mlf file is also created in same way using mkphones1.led edit script, dictionary dict and words.mlf file which include short pauses ("sp") after each word phone group e.g. "sp" is included between phoneme sequence of words "Tony", "Tanvi", "Tinku" and "Tanya" ("'*'/sample1.lab" sil t o n i sp tt a n v i sp t i n k u sp tt aa n y aa sp sil .)

```
HLEd -A -D -T 1 -l '*' -d dict -i phones1.mlf mkphones1.led
words.mlf
```

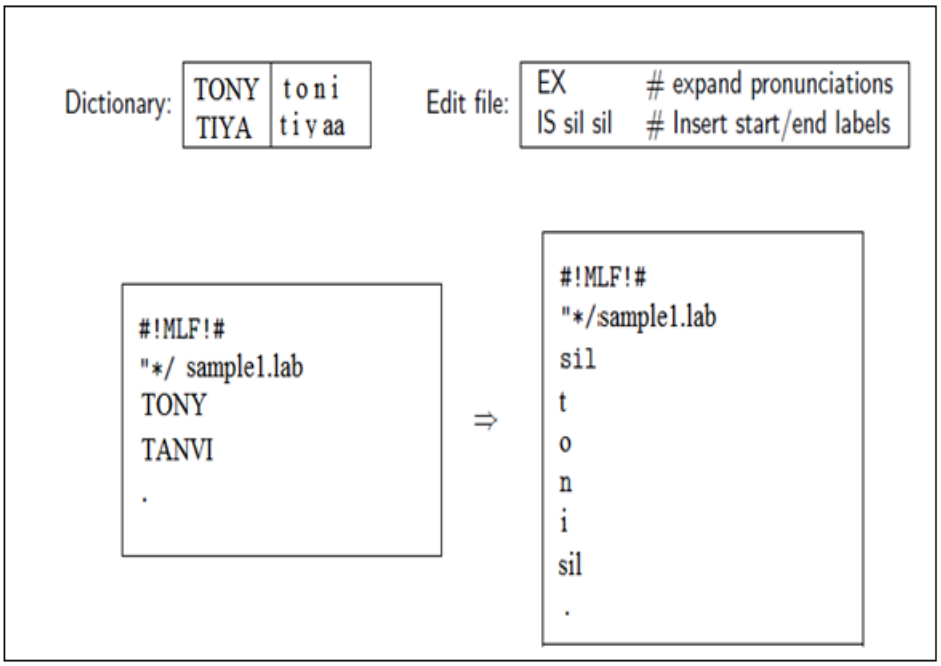


Fig. 3.10: Converting Word To Phone Transcriptions

```
#!MLF!#
"*/sample1.lab" sil t o n i tt a n v i t i n k u tt aa n y aa sil .
"*/sample2.lab" sil t ee n aa t ai h ai l tt r i sh aa tt r i p tt i sil .
"*/sample3.lab" sil tt o m a r tt i sh aa tt u l s i t i y aa sil .
"*/sample4.lab" sil tt a n u tt a n u j aa tt e j i n d a r tt r i sh n aa sil .
"*/sample5.lab" sil tt r i p tt i tt a p e sh tt a p a n tt aa r a n sil .
"*/sample6.lab" sil tt aa r aa tt i l a k tt u sh aa r s a m r aa t sil .
"*/sample7.lab" sil tt aa r a k tt a n u sh tt a n u j tt a k sh a k sil .
"*/sample8.lab" sil tt a k sh i l aa tt a k sh ee l tt i l a k tt a n m e sil .
"*/sample9.lab" sil v aa t i k aa tt a p a s tt a n i sh sil .
"*/sample10.lab" sil tt a n u k aa tt a t i n i tt i th i sil .
"*/sample11.lab" sil tt e j a s v i tt u l s i d aa s tt a y aa g r aa j.
"*/sample12.lab" sil tt a m a n aa tt aa j sil .
"*/sample13.lab" sil tt a n i sh aa tt a v i sh i tt a r p a n aa sil .
"*/sample14.lab" sil tt r i sh n aa tt a p a s y aa bh a t n aa g a r sil .
"*/sample15.lab" sil tt a r a n j o tt tt e j a s v i n i tt e j a l.
"*/sample16.lab" sil tt o y aa tt e j a s tt u n g n aa th sil .
"*/sample17.lab" sil tt aa r aa ch a n d tt a n i sh k tt r u sh aa bh a t sil.
```

Fig. 3.11: Transcription File (phones0.mlf) For Words Containing ਤ or ਟ

```
#!MLF!#
"*/sample1.lab"
sil
k
a
r
a
n
p
r
a
n
a
v
ch
i
r
aa
g
tt
a
r
```

Phoneme representation corresponding to words
Karan
Pranav
Chirag
Tarun

Fig. 3.12: Phone Transcription File For Words Containing ਅ or ਆ

- **Feature Extraction from Data**

  This step in data preparation phase is also known as the "parameterization of the raw speech waveforms into sequences of feature vectors". Because HTK is not as efficient in processing wav files as it is with its internal format therefore the recorded speech samples were parameterized into sequence of spectral features. And the audio wav files viz. **sample1.wav, sample2**.wav etc. were converted to another format called MFCC format (which refers to Mel Frequency Cepstral Coefficients; generally referred to as 'feature vectors' which are derived from FFT (Fast Fourier Transform) based log spectra) e.g. corresponding mfcc files were **sample1.mfc, sample2.mfc** etc.

  The HCopy tool is used to convert these wav files to MFCC format. For this, a script file **codetrain.scp** containing a list of each source audio file and the name of the MFCC file it will be converted to, is created and used as a parameter to the HCopy command.

  A configuration file (**config1.conf**) that specifies all the needed conversion parameters is also used. The speech signals were windowed using a 25 ms Blackman window and 10 ms frame period. The spectral feature vector consisted of 39 mel-cepstral coefficients including the zeroth coefficient (=13) and its delta coefficients (=13) and acceleration coefficients (=13). All these parameters were specified in config1.conf file.

```
SOURCEFORMAT= WAV
TARGETFORMAT = HTK
TARGETKIND = MFCC_0_D_A
TARGETRATE = 100000.0
SAVEWITHCRC = T
WINDOWSIZE = 250000.0
USEHAMMING = T
PREEMCOEF = 0.97
NUMCHANS = 26
CEPLIFTER = 22
NUMCEPS = 12
```

Fig. 3.13: Configuration File (config1.conf)

```
./data/train/wav/sample1.wav ./data/train/mfcc/sample1.mfc
./data/train/wav/sample2.wav ./data/train/mfcc/sample2.mfc
./data/train/wav/sample3.wav ./data/train/mfcc/sample3.mfc
./data/train/wav/sample4.wav ./data/train/mfcc/sample4.mfc
./data/train/wav/sample5.wav ./data/train/mfcc/sample5.mfc
./data/train/wav/sample6.wav ./data/train/mfcc/sample6.mfc
./data/train/wav/sample7.wav ./data/train/mfcc/sample7.mfc
./data/train/wav/sample8.wav ./data/train/mfcc/sample8.mfc
./data/train/wav/sample9.wav ./data/train/mfcc/sample9.mfc
./data/train/wav/sample10.wav ./data/train/mfcc/sample10.mfc
./data/train/wav/sample11.wav ./data/train/mfcc/sample11.mfc
./data/train/wav/sample12.wav ./data/train/mfcc/sample12.mfc
./data/train/wav/sample13.wav ./data/train/mfcc/sample13.mfc
./data/train/wav/sample14.wav ./data/train/mfcc/sample14.mfc
./data/train/wav/sample15.wav ./data/train/mfcc/sample15.mfc
./data/train/wav/sample16.wav ./data/train/mfcc/sample16.mfc
```

Fig. 3.14: (codetrain.scp) Script File

```
C:\htk> HCopy -A -D -T 1 -C config1.conf -S codetrain.scp
```

3.5 TRAINING OF HMM USING HTK (HMM TOOLKIT)

In the training phase of HTK **Viterbi algorithm** is implemented theoretically so that most probable sequence of state phonemes q^ is found from HMM λ (A i.e. transition probability distribution, B i.e. output probability distribution, π i.e. initial state probability), either in the form of concatenated context-dependent triphones like t-o+n or context-independent monophones like t, o, n (phone transcriptions) corresponding to the given output sequence O_i (containing Mel-cepstral coefficients) that is derived from cepstral and mel-cepstral analysis of speech signals.

Here, partial probability of reaching a particular state is calculated by considering the partial probability of last state (δ_t), transition probability (A) and output probability (O). Thus, state sequence of maximum probability is found recursively. In this way the most probable state sequence q^ is obtained that generated the given output feature parameter vector O_i.

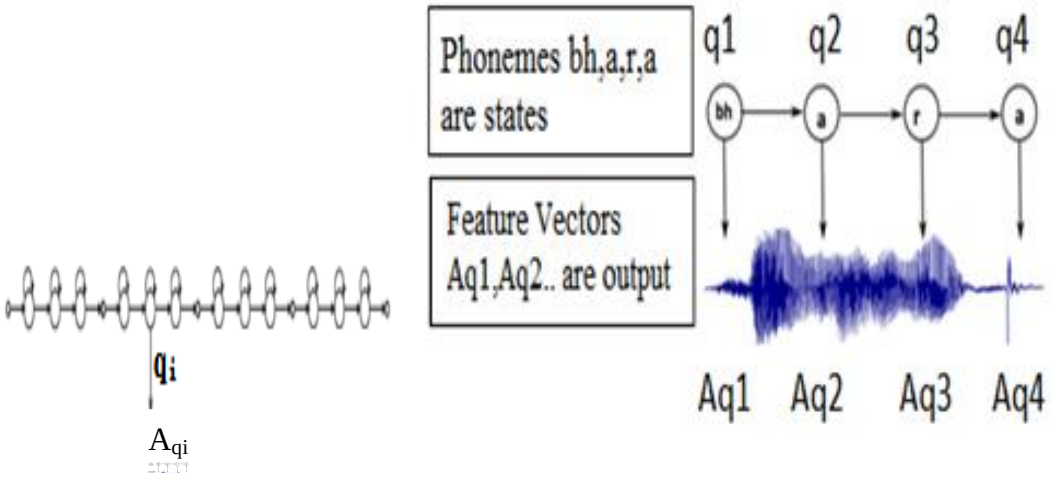
$$\delta_t(i) = \max_q P(q_1,q_2,\ldots, q_t = i, o_1, o_2, o_t \mid \lambda)$$

Where $\delta_t(i)$ – the probability of the most probable path ending in state q_t=i

$q_1, q_2 \ldots$- state sequence

$o_1 o_2 \ldots$- output sequence

Consequently, HMM phone models were trained for given output feature vectors using Viterbi Decoding. These acoustic phone models were thus obtained in training phase after modeling HMMs by various feature parameters obtained from stored speech corpora.



3.15(a) 3.15(b)

Fig. 3.15(a): Concatenated HMM Chain 3.15(b): HMM Chain For Word bharat

The state sequence q^ of the model λ could be maximized independently of Ô

$$q^{\wedge} = \arg\max \{P(q \mid \lambda, L)\}$$

Where $q^{\wedge}$ = q1, q2.., qL is the path through the states of the model λ

q_i is a state at time t_i

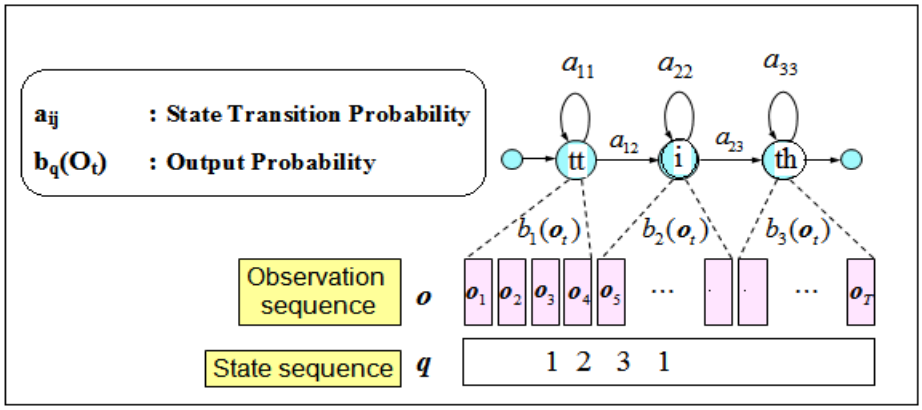


Fig. 3.16: State Sequence Generation For Word "Tithi"

HTK represents output distributions by Gaussian mixture densities. Thus output probability distribution of each state *qi* is represented by one Gaussian density function with a mean vector μ_i and covariance matrix $\sum_i$. The HMM model λ is a set of all means and covariance matrices for all N states:

$$\lambda = (\mu_1, \sum_1, \mu_2, \sum_2, \mu_3, \sum_3, \mu_N, \sum_N).$$

During HMM modeling of acoustic models, means vector μ_i and covariance matrix $\sum_i$ were calculated initially from features extracted from speech corpora and re-estimated for each state of all phone models i.e. firstly for all monophone models and later for all triphone models.

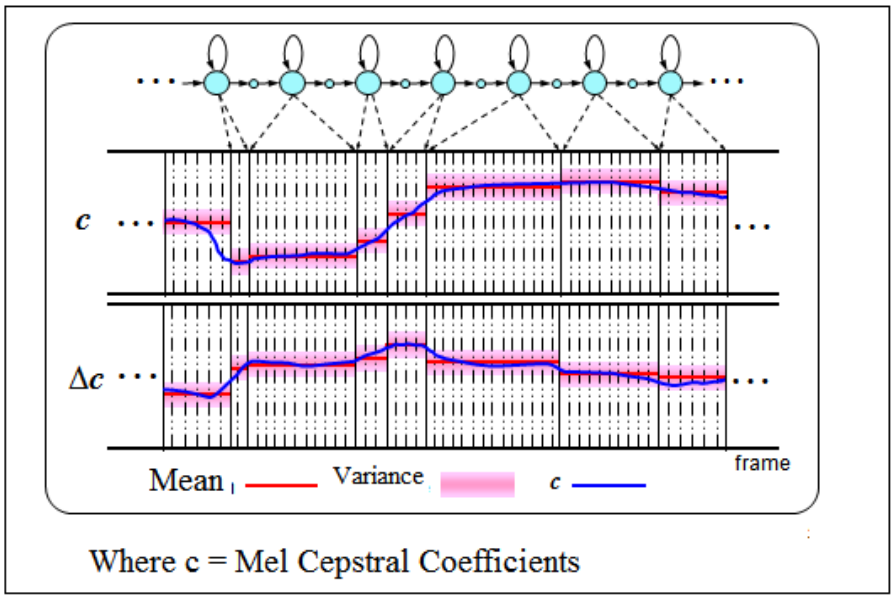


Fig. 3.17: Mean And Variance Representation of MFCC Feature Vector

- **Initialization of HMM (Hidden Markov Models)**

Step -1: Initially for training of HMM a prototype model, **proto**, is defined. It is initialized using HTK tool HCompV that computed the global mean and variance and set all the Gaussians in a given HMM to have the same mean and variance. 5-state left-to-right HMMs with no skips were used in which first and last states were non-emitting states. Remaining second, third and fourth states are emitting ones and their mean and variance vectors are calculated and re-estimated.

A script file train.scp is required to tell HTK where all feature vector files were located and a configuration file config.conf is also used.

```
HCompV -A -D -T 1 -C config.conf -f 0.01 -m -S train.scp -M
hmm0 proto
```

This created **proto** and **vfloors** file in HMM0 folder. vfloors file contained global variance vector multiplied by factor f = 0.01.

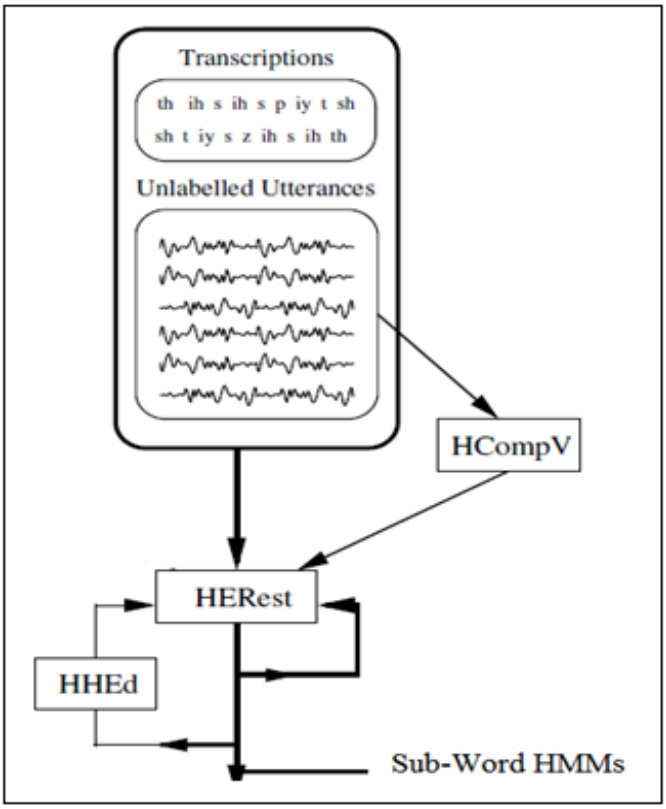


Fig. 3.18: Training Sub-Word HMMs

```
~o <VecSize> 39 <MFCC_0_D_A>
~h "proto"
<BeginHMM>
<NumStates> 5
<State> 2
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0…..
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0…..
<State> 3
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0…..
<Variance>39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0…..
<State> 4
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0…..
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0…..
<TransP> 5
0.0 1.0 0.0 0.0 0.0
0.0 0.6 0.4 0.0 0.0
0.0 0.0 0.6 0.4 0.0
0.0 0.0 0.0 0.7 0.3
0.0 0.0 0.0 0.0 0.0
<EndHMM>
```

Fig. 3.19: Initial Prototype Model (proto)

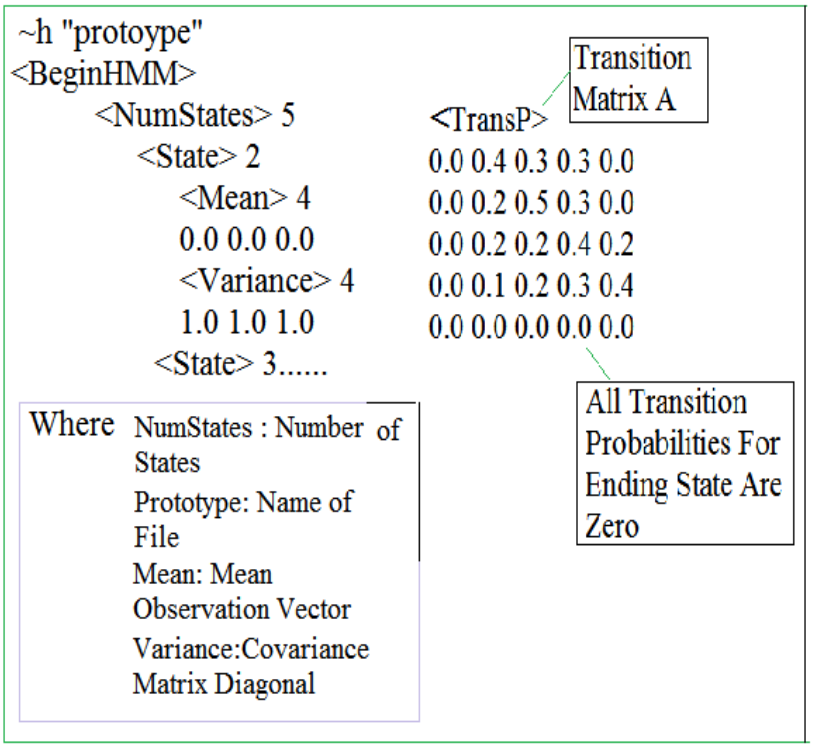


Fig. 3.20: HMM Model

```
~v varFloor1 <Variance> 39  2.777049e-001 7.315862e-001 4.440626e-
001 6.700849e-001 3.530515e-001 5.895303e-001 4.417108e-001
6.626121e-001 3.455022e-001 5.118520e-001 3.947881e-001
4.259625e-001 2.465594e+000 1.560931e-002 2.507905e-002
2.930563e-002 4.227093e-002 2.574000e-002 3.440582e-002
2.875688e-002 4.522292e-002 2.780856e-002 3.446863e-002
3.179696e-002 2.745556e-002 1.368608e-001 2.634353e-003
3.296673e-003 4.581431e-003 6.709039e-003 4.611471e-003
5.634170e-003 5.161444e-003 7.645173e-003 5.180779e-003
5.809075e-003 5.807454e-003 4.803537e-003 1.902239e-002
```

Fig. 3.21: Generated vfloors File

Step -2: In this step, **hmmdefs** file that contained flat start monophones and **macros** file is created manually. The global mean and variance calculated by HCompV is copied for each phone contained in monophones0 into hmmdefs file thus creating a HMM model with 3 emitting states for each phone. The macros file is created by copying vfloors file to it with certain modifications.

The HMM model developed for various monophones representing mean (~u) and variance (~v) vectors represented in this file were re-estimated in later steps.

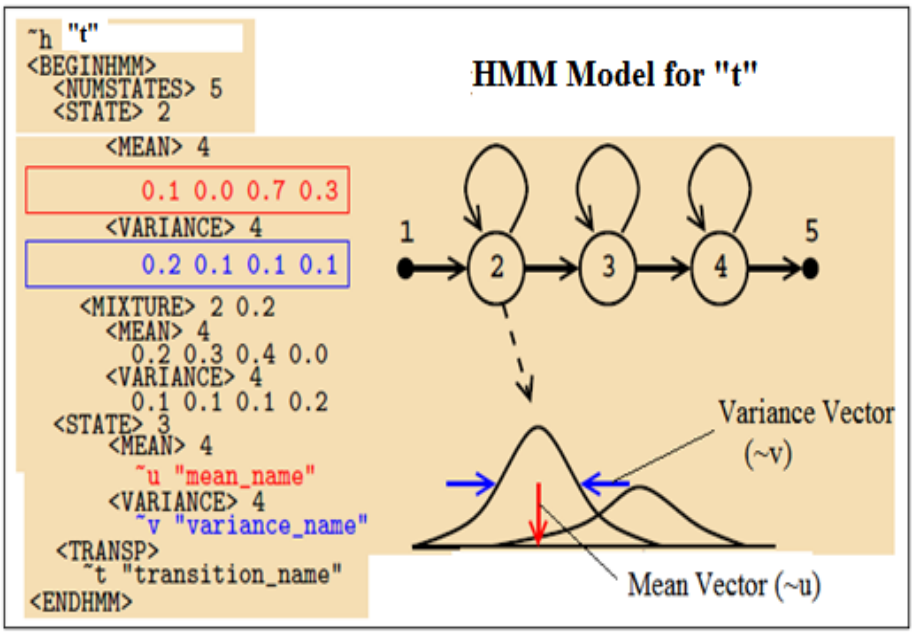


Fig. 3.22: HMM Model For Monophone “\t\” i.e. \ट\ of Word bhatt etc.

```
~h "a"
<BEGINHMM>
<NUMSTATES> 5
<STATE> 2
<MEAN> 39  -6.793139e+00 -2.883674e+000 -1.007505e+001
<VARIANCE> 39  2.777049e+001 7.315862e+001 4.440626e+001 …..
<GCONST> 1.312633e+002
<STATE> 3
<MEAN> 39  -6.793139e+000 -2.926306e+000  -1.007505e+001 -
4.524404e-001….
<VARIANCE> 39  2.777049e+001 7.315862e+001  4.440626e+001....
<GCONST> 1.312633e+002
<STATE> 4
<MEAN> 39  -6.793139e+000 --1.007505e+001 -4.524404e-001…..
<VARIANCE> 39  2.777049e+001  3.455022e+001 5.118520e+001…..
<GCONST> 1.312633e+002
<TRANSP> 5  0.000000e+000 1.000000e+000 0.000000e+000
0.000000e+000 3.000000e-001  0.000000e+000 0.000000e+000
<ENDHMM>
```

Fig. 3.23: Monophone hmmdefs File

- **Re-estimation (Training) of HMM models**

The system is trained for 27 monophones models. In this, various parameters of these Flat Start Monophones are re-estimated using the embedded re-estimation tool HERest following Baum-Welch Re-estimation theoretically by recalculating new mean and variance vectors for the phone HMM models. Through this various parameters (A, B, π) of all phones HMM model (λ) is re-estimated using following formulae.

$\hat{\pi} = \gamma_1(i)$ Re-estimated Initial Probability — the expected frequency of state i at time $t=1$

$\hat{a}_{ij} = \frac{\sum \xi_t(i,j)}{\sum \gamma_t(i)}$ • Re-estimated Transition Probability - ratio of expected no. of transitions from state i to j over expected no. of transitions from state i

$\hat{b}_j(k) = \frac{\sum_{t,o_t=k} \gamma_t(j)}{\sum \gamma_t(j)}$ • Re-estimated Output Probability - ratio of expected no. of times in state j observing symbol k over expected no. of times in state j

Re-estimated HMM model parameters from initial HMM model (λ) parameters (A, B, π)

Fig. 3.24: Baum-Welch Re-estimation Algorithm

The aim of this step is to load all the phone models in the hmm0 folder contained in the hmmdefs file, and re-estimate them using the MFCC files listed in the train.scp script, and create a new model set in hmmdefs file in hmm1 directory. For each state, of all the monophones, mean and variance vectors are estimated.

```
HERest -A -D -T 1 -C config -I phones0.mlf -t 250.0 150.0
1000.0 -S train.scp -H hmm0/macros -H hmm0/hmmdefs -M
hmm1 monophones0
```

Similarly, HMM phone models are re-estimated again creating new model sets in hmm2 and hmm3 using label (.lab) files like phones0.mlf, feature (.mfc) files, configuration file (config.conf), and hmmdefs and macros files i.e. HMM models of previous stage using HERest (HMM Re-estimation) tool of HTK that follows Baum-Welch Re-estimation algorithm.

-T option helps to trace the execution of the command. –A prints the command line arguments and –D option helps to display configuration variables that could be used for examination.

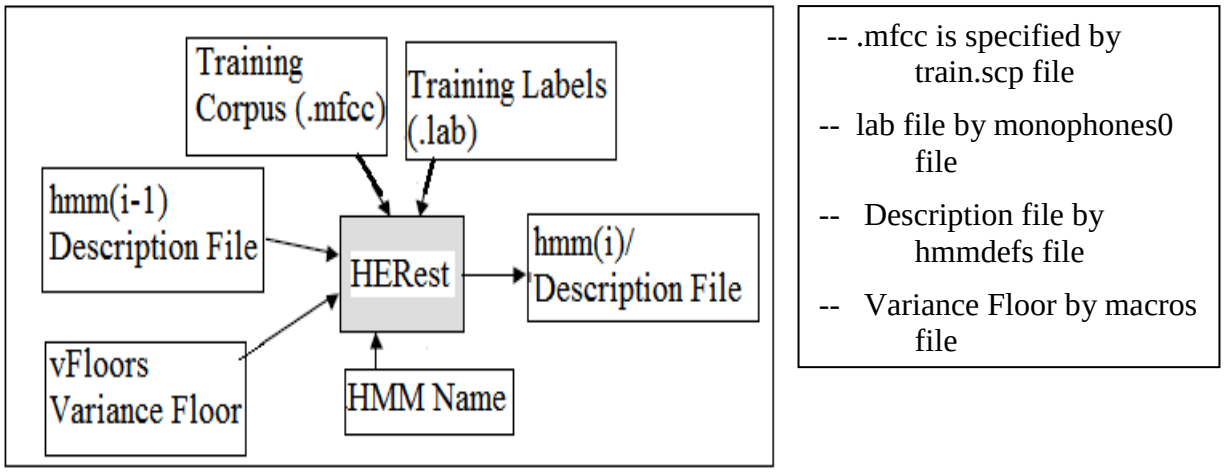


Fig. 3.25: Re-estimation By HERest Tool

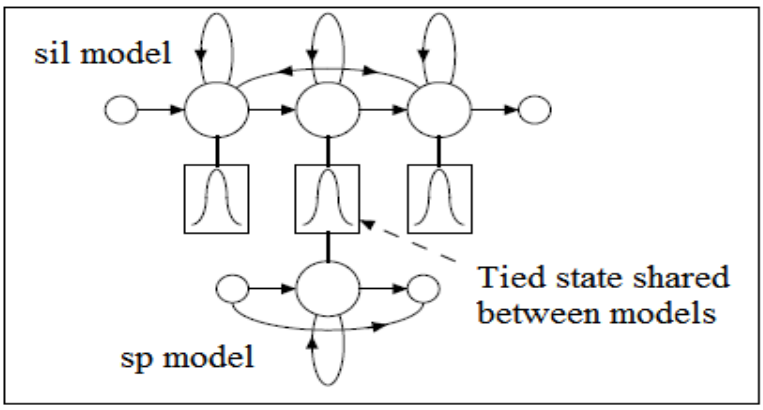
- **Fixing of The Silence Models**

HMM models created up to level HMM 3 did not include a "sp" (short pause) silence model i.e. the short pauses that occurred between words in normal

speech while “sil” the silence models were of longer duration, and referred to the pauses occur at the end of a sentence. The sp model needed to have its "emitting state tied to the centre state of the silence model". For this hmmdefs in HMM 3 directory is copied to HMM 4 directory and a new sp model in hmmdefs (in HMM 4 directory) is created and tied to the centre state of sil using the HMM editor called **HHEd** and command script, called **sil.hed.**

Tying helped one or more HMMs to share the same set of parameters.

```
HHEd -A -D -T 1 -H hmm4/macros -H hmm4/hmmdefs -M
hmm5 sil.hed monophones1
```



Sil model with 5 states having 3 non-emitting states
Sp model with 3 states having 1 non-emitting state

Fig. 3.26: Silence Models

```
~h "sp" <BEGINHMM> <NUMSTATES> 3 <STATE> 2
<MEAN> 39
-5.264097e+000 -5.562384e+000 8.856168e+000 ...
<VARIANCE> 39
2.485007e+001 8.017378e+001 4.419154e+001...
<GCONST> 1.244895e+002
<TRANSP> 3
0.0 1.0 0.0
0.0 0.9 0.1
0.0 0.0 0.0
<ENDHMM>
```

Fig. 3.27: sp HMM Model

After this, the HMM models were re-estimated two more times and hmmdefs and macros in directory HMM 7 were created.

- **Realignment of Training Data**

This step converted the input word level transcription **words.mlf** to a new phone level transcription file **aligned.mlf** using the pronunciations stored in the dictionary **dict** given as input. As the dictionary could contain multiple pronunciations for some words, particularly function words e.g. word "the" could have pronunciations corresponding to "t, h, a" or "t, h, i". Therefore, HVite tool is used to consider all pronunciations for each word and output the pronunciation that best matched the acoustic data. The phone models thus created were used to realign the training data and new transcriptions were created.

```
HVite -A -D -T 1 -l '*' -o SWT -b SENT-END -C config.conf -
H hmm7/macros -H hmm7/hmmdefs -i aligned.mlf -m -t 250.0
150.0 1000.0 -y lab -a -I words.mlf -S train.scp dict
monophones1> HVite_log
```

The -b option is used to insert a silence model at the start and end of each utterance. The -t option set pruning level of 250.0 and the -o option is used to suppress the printing of scores, word names and time boundaries in the output MLF. **HVite_log** is the log file to record the output command.

After this HMM model re-estimation is repeated two times to create **hmmdefs** and **macros** files of HMM 8 and HMM 9 directories.

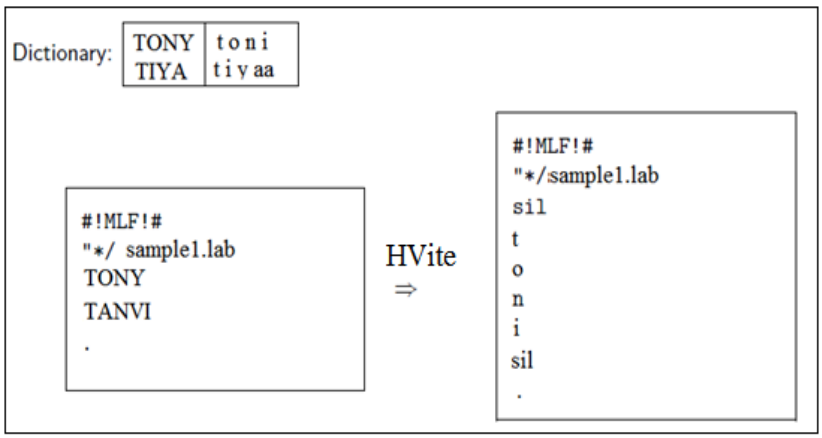


Fig. 3.28: Generation of Aligned.mlf by HVite Tool

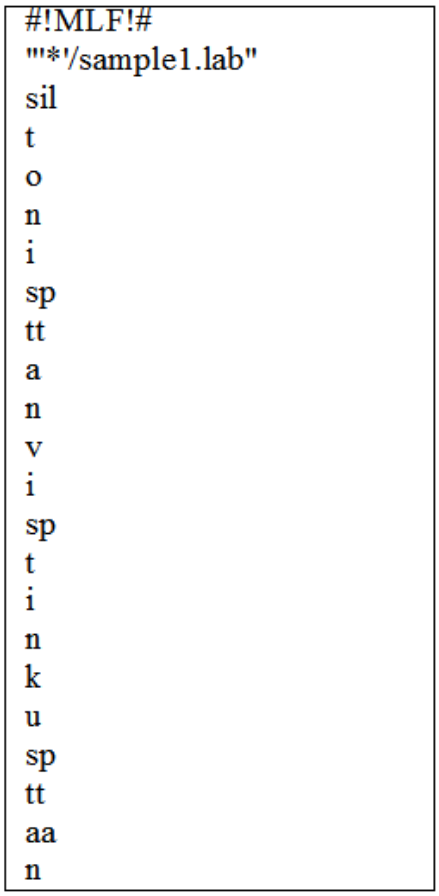


Fig. 3.29: Aligned.mlf File

- **Creating Triphones From Monophones**

  Context-dependent triphones were made, in this step, by cloning monophones and then re-estimating using triphones transcriptions that were created using HLEd tool. The triphones is created from monophones in the form "L-X+R" (where X is a phoneme, L and R are left & right context, respectively) e.g. in word Bharat, for phoneme ‘a’ triphones generated is bh-a+r.

  Triphones transcription lead to an improved level as instead of searching for a monophone alone, it is looked in the “context” of other monophones thus increasing the accuracy due to matching of a specific sequence of 3 sounds together (a triphone), rather than only one sound.

  The monophone transcriptions in the **aligned.mlf** file is converted to triphone transcription **triphones1** file by using **mktri.led** edit script and HLEd tool that creates **wintri.mlf** file along with **triphones1** file which is further used to create **mktri.hed** file using maketrihed script.

```
HLEd -A -D -T 1 -n triphones1 -l '*' -i wintri.mlf mktri.led
aligned.mlf
```

```
perl.../HTK_scripts/maketrihed monophones1 triphones1
```

The output files generated that contained triphone models like “t-o+n” for word “tony”, “tt-a+n” for word “tanvi” etc. were **wintri.mlf** specifying triphone HMM models according to samples and **triphones1** containing list of all triphone models.

```
#!MLF!#
"*/sample1.lab"
sil
t+o
t-o+n
o-n+i
n-i
sp
tt+a
tt-a+n
a-n+v
n-v+i
v-i
sp
t+i
t-i+n
i-n+k
n-k+u
k-u
sp
```

Fig. 3.30: (wintri.mlf) Containing Triphones

```
sil
t+o
t-o+n
o-n+i
n-i
sp
tt+a
tt-a+n
a-n+v
n-v+i
v-i
t+i
t-i+n
i-n+k
n-k+u
k-u
tt+aa
tt-aa+n
aa-n+y
n-y+aa
y-aa
t+ee
t-ee+n
ee-n+aa
```

Fig. 3.31: triphones1 File

```
CL triphones1
TI T_bh {(*-bh+*,bh+*,*-bh).transP}
TI T_a {(*-a+*,a+*,*-a).transP}
TI T_t {(*-t+*,t+*,*-t).transP}
TI T_n {(*-n+*,n+*,*-n).transP}
TI T_aa {(*-aa+*,aa+*,*-aa).transP}
TI T_g {(*-g+*,g+*,*-g).transP}
TI T_r {(*-r+*,r+*,*-r).transP}
TI T_sp {(*-sp+*,sp+*,*-sp).transP}
TI T_ch {(*-ch+*,ch+*,*-ch).transP}
TI T_y {(*-y+*,y+*,*-y).transP}
TI T_s {(*-s+*,s+*,*-s).transP}
TI T_m {(*-m+*,m+*,*-m).transP}
TI T_sil {(*-sil+*,sil+*,*-sil).transP}
TI T_tt {(*-tt+*,tt+*,*-tt).transP}
TI T_j {(*-j+*,j+*,*-j).transP}
TI T_k {(*-k+*,k+*,*-k).transP}
TI T_sh {(*-sh+*,sh+*,*-sh).transP}
TI T_ee {(*-ee+*,ee+*,*-ee).transP}
TI T_l {(*-l+*,l+*,*-l).transP}
TI T_i {(*-i+*,i+*,*-i).transP}
TI T_e {(*-e+*,e+*,*-e).transP}
TI T_u {(*-u+*,u+*,*-u).transP}
```

Fig. 3.32: (mktri.hed) Edit File

The edit script **mktri.hed** contained a clone command CL followed by TI commands to tie all of the transition matrices in each triphones set. The name of the file containing the list of triphones is given to the clone command CL as its argument. For each model of the form a-b+c in this list, monophone b is looked and copied. The name of a macro and a list of HMM components is given to TI command as its argument. The list of items within brackets is the patterns designed to match the set of triphones, right biphones and left biphones for each phone. After that edit script HHEd along with mktri.hed file is used to efficiently clone the models.

```
HHEd -A -D -T 1 -H hmm9/macros -H hmm9/hmmdefs -M
hmm10 mktri.hed monophones1
```

```
~h "n-d+a" <BEGINHMM>
 <NUMSTATES> 5
<STATE> 2
<MEAN> 39  -6.793139e+000 -2.926306e+000 -7.762902e+000…..
<VARIANCE> 39  2.777049e+001 4.440626e+001 5.118520e+001 ….
<GCONST> 1.312633e+002
<STATE> 3
 <MEAN> 39  -6.793139e+000 -1.007505e+001 -4.524404e-001…..
<VARIANCE> 39  2.777049e+001 7.315862e+001 4.440626e+001
5.118520e+001…..
<GCONST> 1.312633e+002
<STATE> 4
<MEAN> 39  -6.793139e+000  -7.762902e+000 -2.883674e+000 -
1.007505e+001 -4.524404e-001….
<VARIANCE> 39  2.777049e+001 7.315862e+001 4.440626e+001
6.700849e+001….
 <GCONST> 1.312633e+002 ~t "T_d"
<ENDHMM>
```

Fig. 3.33: HMM Model of Triphone “n-d+a” For Phone “d”

Once the context-dependent models had been cloned, the new triphone set were re-estimated using HERest to form **hmmdefs** and **macros** file in hmm11 and hmm12 directories.

```
HERest -A -D -T 1 -C config.conf -I wintri.mlf -t 250.0 150.0
3000.0 -s stats -S train.scp -H hmm11/macros –H
hmm11/hmmdefs -M hmm12 triphones1
```

Here **stats** file is also created that specifies the occurrence statistics of triphones in database.

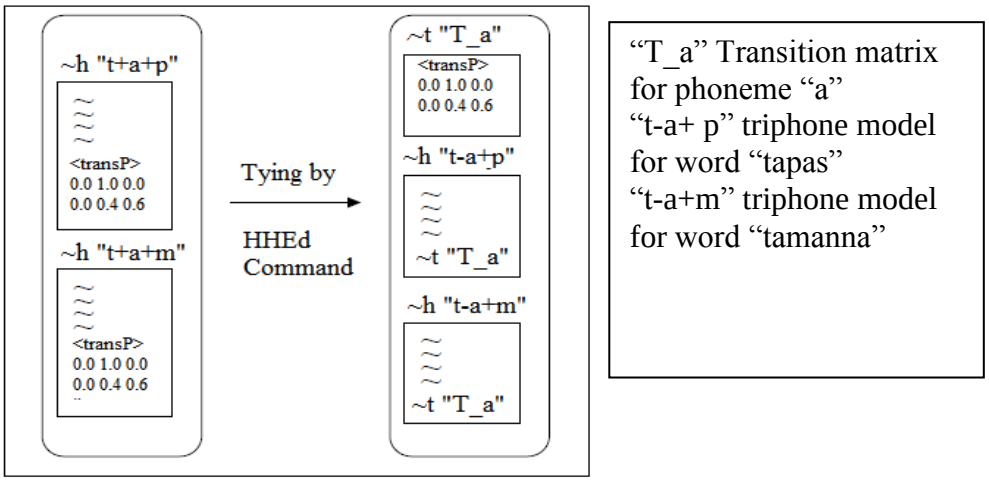


Fig. 3.34: Tying Transition Matrices

- **Creating Tied State Triphones**

  A set of triphones HMMs like "t-a+p" for word "tapas" and "t-a+m" for word "tamanna" with all triphones in a phone set sharing the same transition matrix were created in the previous step. States within these triphones sets were tied so that data could be shared and parameters could be estimated correctly.

  In order to perform tying of states "Decision Tree based Clustering" is used. It is performed by running HHEd tool of HTK. HHEd is used to cluster the states and then each cluster is tied. Decision trees are based on asking questions about the left and right contexts of each triphones and find those contexts which make the largest difference to the acoustics and which should therefore distinguish between the different clusters [28].

For example in fig. states for phoneme “a” are clustered. Words like bharat (bh-a+r), tamanna (tt-a+m) and taran (r-a+n) are considered i.e. all triphone models for phoneme “a”. They are splitted into various clusters according to the questions (QS) defined in tree.hed for the right and left contexts of phoneme. E.g. in word tamanna, triphone model “tt-a+m”, right context (R) is “m” that is defined as nasal in tree.hed and left context (L) is “tt” which is unvoiced consonant as defined in the file. According to these phonetic information about contexts various questions are answered in tree based clustering and clusters of states are produced.

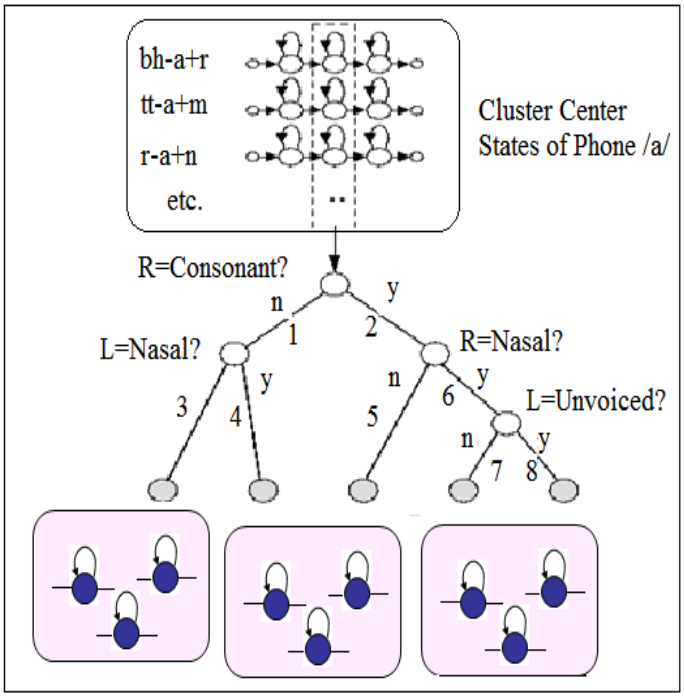


Fig. 3.35: Decision Tree Clustering

To perform Decision Tree Clustering and tie the states, firstly files named **dict-tri** and **fulllist** were created by following command. Initial fullest contains only phonemes used but later on it is modified using **triphones1** file and **fixfullist.pl** script to contain a list of distinct monophones and triphones.

```
HDMan -A -D -T 1 -b sp -n fulllist -g global.ded -l flog dict-tri
../lexicon/list.txt
```

```
perl fixfulllist.pl fulllist1 fulllist
```

After this an edit script **tree.hed** is used that contained the user-defined questions (QS) according to the language and defined by a set of contexts and TB commands specified in **tree.hed** by following HTK command.

```
perl .../HTK_scripts/mkclscript.prl TB 350 monophones0 >>
tree.hed
```

```
QS  "R_NonBoundary"          { *+* }

QS  "R_Stop"                 {*+p,*+ph,*+b,*+t,*+d,*+dd,*+k,*+g,
*+dh,*+kg}

QS  "R_Nasal"                { *+m,*+n }

QS  "R_Unvoiced-All"         {*+p,*+t,*+tt,*+k,*+kh,*+tth,*+chh,
*+ch,*+sil }

QS  "R_NonAffricate"         { *+s,*+sh,*+z,*+f,*+v,*+th,*+dh }

TB 100 "ST_sil_2_"           {("sil","*-sil+*","sil+*","*-sil").state [2]}

TB 100 "ST_bh_3_"            {("bh","*-bh+*","bh+*","*-bh").state
```

E.g. -- “ਪ” in ਤਪਸ
is “*+p” in
R_stop

--“ਠ” in ਠਾਕੁਰ
is “*+tth” in
R_Unvoiced

Fig. 3.36: Tree.hed file

TB command firstly, formed a single cluster of set of states defined by the final argument and then each question in the question set loaded by the QS commands is used to split the pool into two sets based on the context information e.g. for word “bharat” the triphones set is “bh-a+r”, then sets of states were splitted according to acoustic and phonetic information of right context “bh” and left context “r”. This increases the log likelihood of the training data.

Then **tiedlist** file is generated that contains set of tied clusters by following command by using HHEd script.

```
HHEd -A -D -T 1 -H hmm12/macros -H hmm12/hmmdefs -M
hmm13 tree.hed triphones1
```

```
~s "ST_s_2_1"
<MEAN> 39  -6.793139e+000 -2.926306e+000 -5.525350e+000
<VARIANCE> 39  2.777049e+001 7.315862e+001 4.440625e+001
<GCONST> 1.312633e+002
~s "ST_s_3_1"
<MEAN> 39  -6.793139e+000 -2.926306e+000 -5.525350e+000
<VARIANCE> 39  2.777050e+001 7.315862e+001 4.440626e+001
<GCONST> 1.312633e+002
~s "ST_s_4_1"
<MEAN> 39  -6.793139e+000 -2.926306e+000 -5.525350e+000
<VARIANCE> 39  2.777049e+001 7.315863e+001 4.440625e+001
<GCONST> 1.312633e+002
~h "s" <BEGINHMM> <NUMSTATES> 5 <STATE> 2 ~s "ST_s_2_1"
<STATE> 3 ~s "ST_s_3_1" <STATE> 4 ~s "ST_s_4_1" ~t "T_s"
<ENDHMM>
```

Fig. 3.37: hmmdefs file after tying states for phoneme “s”

```
p-tt+i
aa
ai
bh
ch
ee
sh
sp
th
tt p-tt+i
t-aa+ch aa
ch-aa+r aa
a-t+aa t+ai
i-sh+aa sh
l-aa aa
j-o+tt o
aa-th th
i-k+aa k
t-n+aa n
bh+a bh
n-d+a d
k-aa+r aa
n-g+n g
a-m+a m
```

Where

triphones models like p-tt+i is for tt in word tripti

ch-aa+r for aa in word bhattacharya

Fig. 3.38: Tiedlist

The generated hmmdefs and macros files in HMM13 directory were re-estimated two times to finally generate hmmdefs and macros file in HMM15 directory.

```
~o <STREAMINFO> 1 39 <VECSIZE>
39<NULLD><MFCC_D_A_0><DIAGC>
~v "varFloor1"
<VARIANCE> 39  2.777049e-001 7.315862e-001 4.440626e-
001 6.700849e-001 3.530515e-001 5.895303e-001 4.417108e-
001 6.626121e-001 3.455022e-001 5.118520e-001 3.947881e-
001 4.259625e-001 2.465594e+000 1.560931e-002 2.507905e-
002 2.930563e-002 4.227093e-002 2.574000e-002
```

Fig. 3.39: (macros file) HMM15 Directory

```
~o
<STREAMINFO> 1 39
<VECSIZE> 39<NULLD>
<MFCC_D_A_0>
<DIAGC>
~t "T_a"
<TRANSP> 5
0.000000e+000 1.000000e+000 0.000000e+000 0.000000e+000
1.387163e-002 9.861284e-001 0.000000e+000 0.000000e+000.....
~s "ST_a_2_1"
<MEAN> 39
-6.944791e+000 -2.526826e+000 -3.831622e+000 -9.338988e+000
-5.618311e+000 -8.123528e+000 -1.587589e+000 -1.022135e+001....
<VARIANCE> 39
2.094680e+001 5.261876e+001 4.868203e+001 5.497910e+001
2.633384e+001 4.728995e+001 4.244819e+001 5.791674e+001.......
<GCONST> 1.293951e+002
~s"ST_a_3_1"
<MEAN> 39
-7.362495e+000 -2.835286e+000 -3.415716e+000 -8.929062e+000
```

Fig. 3.40: (hmmdefs file) HMM15 directory

These files viz. hmmdefs and macros are further used as an input for testing phase as HMM models of various phones.

CHAPTER -4

TESTING AND RESULTS

After completion of the training phase, the system is tested for certain strings for which it was trained in the training phase of implementation. These words are obtained after processing them in pre-processing stage for titles like Mr., initials like K.K. Gupta, numbers like Tapas123 etc. and then are given as input to the testing phase. In this part the test data i.e. text which is to be mapped to speech signals was given as input along with re-estimated context-dependent HMMs obtained after training phase. According to the phoneme sequence in text labels the context-dependent HMMs were concatenated with the help of HVite tool and word network file wdnet. The main objective of this HMM (Hidden Markov Model) based tool was to generate a sequence of phonemes after selecting from various alternatives pronunciations of the words containing ਤ and ਟ i.e. whether Tony corresponds to ਤੋਨਿ (phoneme sequence is ਤ, ੋ, ਨ, ਿ) or ਟੋਨਿ (phoneme sequence is ਟ, ੋ, ਨ, ਿ) and of the words containing ਅ and ਆ e.g. word “Aalap” (ਆਲਾਪ) corresponds to phoneme sequence (ਆ, ਲ, ਆ, ਪ) or (ਆ, ਲ, ਅ, ਪ) corresponding to the text given as input to the system. So that these phoneme sequences could be concatenated following Concatenative approach to generate speech signals according to the text input which is, in this case, is a caller’s name stored in the phone directory of mobile phone.

This sequence of phonemes of the input text prompts through following steps:-

4.1 TESTING OF HTK

4.1.1 CREATION OF TEST PROMPTS

- **Pre-Defined Grammar**

  Firstly a grammar file called **gram** was created based on the regular language notations that defined a variable called name_n denoting the various names in

the text. This file is used to create word network file **wdnet** by following command.

```
HParse gram wdnet
```

The word network file **wdnet** consisted of a list of nodes and a list of arcs. The nodes represented the words and the arcs represented the transition between the words. Each node and arc definition was written on a single line and consisted of a number of fields. Each field specification consisted of a "name=value" pair.

The word network file was used by HTK tool (HNet) along with dictionary **dict** that provided pronunciations for each word in the network and a set of acoustic HMM phone models to choose the appropriate phoneme sequence according to text given.

```
$name_n=TARPANA|TULSI|TANISH|TUSHAR|TINKU|TAN
YA|TILAK|TIYA|TAPASYA|TAKSHEEL|TANUJA|TITHI|VA
TIKA|TATINI|TRIPTI|TEHAL|TEENA|TAMANNA|TARA|B
HATT|TAPAS|TEJAS|TRISHNA|TUKARAM|TARAN|TRUS
HA|TARACHAND|TANMAY;

(SENT-START ($name_n) SENT-END)
```

Fig. 4.1: (gram) Grammar File

```
VERSION=1.0
N=33   L=59
I=0    W=!NULL
I=1    W=!NULL
I=2    W=SENT-START
I=3    W=TARPANA
I=4    W=!NULL
I=5    W=TULSI
I=6    W=TANISH
I=7    W=TUSHAR
-------
J=0     S=32   E=1
J=1     S=0    E=2
J=2     S=2    E=3
J=3     S=3    E=4
J=4     S=5    E=4
J=5     S=6    E=4
J=6     S=7    E=4
J=7     S=8    E=4

---------
```

Where N = Number of nodes in network
L = Number of arcs in the network
I = Node Number
W = word like Tarpana, Tulsi etc..
J = arc-number
S = Start-node
E = End-node

Fig. 4.2: Word Network File (wdnet)

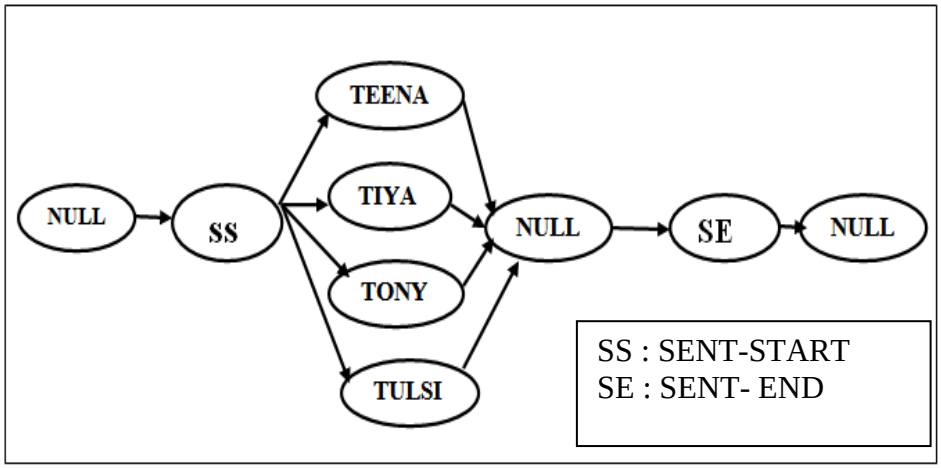


Fig. 4.3: Word Network In wdnet File

- **Test Prompts**

A file called **testprompts.txt** was created manually containing all the test words containing \t\ and \tt\ in 28 test samples testprompts1.txt containing words having \a\ and \aa\ in 45 test samples. Each sample containing one words each. Thus tool is tested for total the words that were already there in training data of training phase i.e. system was already trained on them.

This **testprompts.txt** file was converted to an MLF (Master Label File) file called **testref.mlf** containing all the test samples .lab files viz. test1.lab, test2.lab etc. that HTK can process. It is generated using prompts2mlf script using following perl command.

```
perl ../HTK_scripts/prompts2mlf testref.mlf testprompts
```

```
*/test1 JAWAHAR
*/test2 RUPALI
*/test3 VIKRANT
*/test4 PRAKASH
*/test5 AMAR
*/test6 BHAWAN
*/test7 TARUN
*/test8 JANVI
*/test9 FALGUNI
*/test10 DHARIKA
*/test11 DHVANI
*/test12 RAVI
*/test13 MANAV
*/test14 RATNAKAR
*/test15 MALTI
*/test16 KASANA
```

Fig. 4.4: (testprompts1.txt) File For Words \a\ And \aa\

```
*/test1 TARPANA
*/test2 TULSI
*/test3 TANISH
*/test4 TUSHAR
*/test5 TINKU
*/test6 TANYA
*/test7 TILAK
*/test8 TIYA
*/test9 TAPASYA
*/test10 TAKSHEEL
*/test11 TANUJA
*/test13 TITHI
*/test14 VATIKA
*/test15 TATINI
*/test16 TRIPTI
*/test17 TEHAL
*/test18 TEENA
*/test19 TAMANNA
*/test20 TARA
*/test21 BHATT
*/test23 TAPAS
*/test24 TEJAS
*/test25 TRISHNA   ----- etc.
```

Fig. 4.5: (testprompts.txt) File For Words \t\ And \tt\

```
#!MLF!#
"*/test1.lab"
TARPANA
.
"*/test2.lab"
TULSI
.
"*/test3.lab"
TANISH
.
"*/test4.lab"
TUSHAR
.
"*/test5.lab"
TINKU
.
"*/test6.lab"
TANYA
.
"*/test7.lab"
TILAK
.
"*/test8.lab"
TIYA
.
```

Fig. 4.6: (testref.mlf) Reference File

4.1.2 RECORDING OF TEST DATA

All the test samples in testprompts.txt were recorded i.e. 28 Punjabi words were recorded by a female speaker using Power Sound Editor with following specifications:-

- Sampling Rate 8000 Hz
- Bit Depth 16 bits
- Channel Mono
- Environment Room Environment
- Format Wave (.wav file)

Recorded test files were in form like test1.wav, test2.wav etc.

4.1.3 CODING OF DATA

Feature vectors i.e. Mel cepstral coefficients (MFCC) were extracted from all the recorded .wav files by converting them into .mfc files through HCopy tool of HTK. Files generated are like test1.mfc, test2.mfc etc.

Script file **codetrain.scp** containing paths of all speech files and path of directory in which corresponding mfc files would be created was specified. Configuration file config1.conf was used specifying configuration details like

- The speech signals would be windowed using a 25 ms Blackman window and 10 ms frame period.
- The spectral feature vector would be consisted of 39 mel-cepstral coefficients including the zeroth coefficient (=13) and its delta coefficients (=13) and acceleration coefficients (=13).

MFCC features contained in corresponding mfc files are used by HVite tool further. They are helpful in determining the text word given as test input to the system and then its phoneme sequence will be selected using word network file thus, lastly generating the phoneme sequence.

```
HCopy -A -D -T 1 -C config1.conf -S codetrain.scp
```

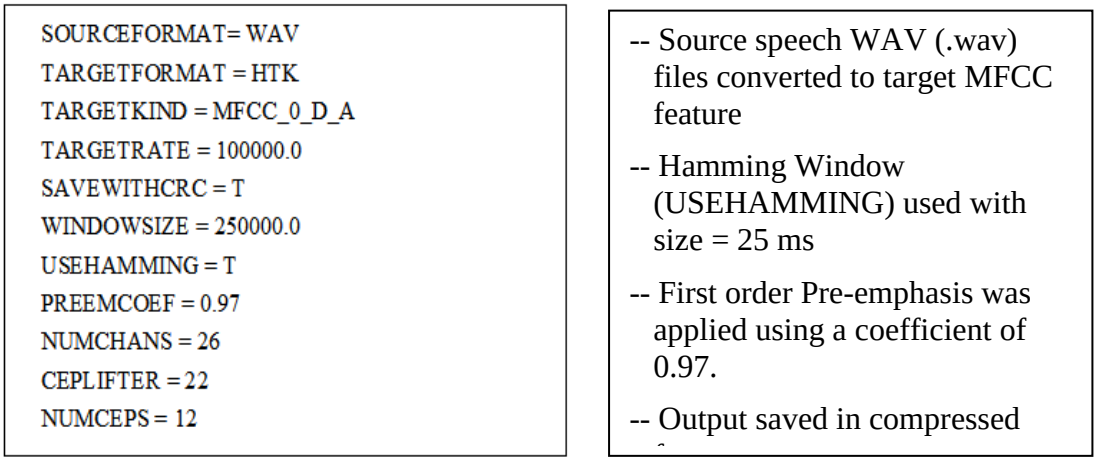


Fig. 4.7: Configuration File (Config.conf)

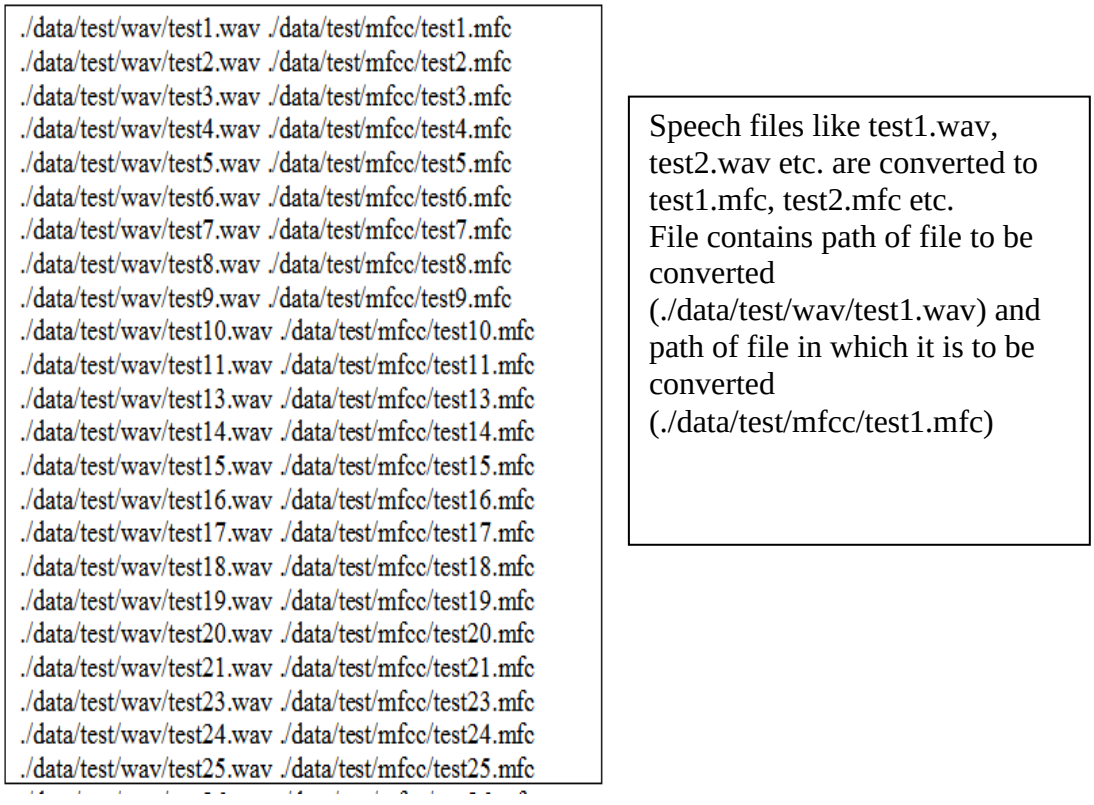


Fig. 4.8: Script File (Test.scp)

4.1.4 TESTING ACOUSTIC MODEL USING HTK

It is the final step of testing phase wherein script file **test.scp** containing path of MFCC files, pronunciation dictionary **dict** or lexicon file **list.txt**, HMM models created in training phase were used to generate **recout.mlf** file that contains all the test words input in the form of .mfc files and **phonemes.txt** file that contain resultant

phoneme sequence for each test word using HTK tool HVite. Configuration file **config.conf** was also used and following command was executed:-

```
HVite -A -D -T 1 -H hmm15/macros -H hmm15/hmmdefs -C
config.conf -S test.scp -l '*' -i recout.mlf -w wdnet
../lexicon/list.txt tiedlist
```

HTK tool called HVite performed the functions provided by HNet and HRec tools of HTK to be invoked from the command line. However, HVite is just a shell containing calls to load the word network, dictionary and models; generate the recognition network and then repeatedly recognize each input utterance. The options

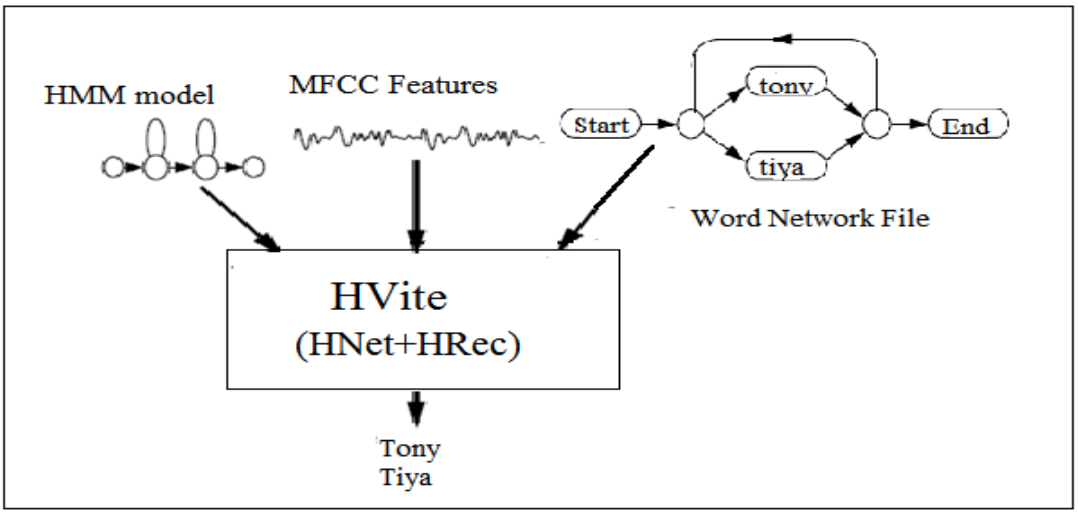
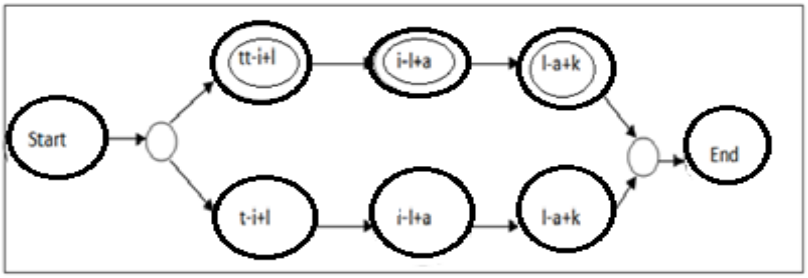


Fig. 4.9: Working of HVite Tool

The dictionary **dict** or **list.txt**, HMM model set generated in training phase i.e. **hmmdefs** and **macros** files of hmm15 directory and word network file **wdnet** were input to the HTK library module HNet whose function was to generate an equivalent network of HMMs. Each word in the dictionary may have several pronunciations and in this case there would be one branch in the network corresponding to each alternative pronunciation. Each pronunciation consisted either of a list of phones or a list of HMM names. In the former case, HNet could optionally expand the HMM network to use word internal triphones.



represents path followed in network for choosing correct pronunciation internally using HNet

Fig. 4.10: Network generated containing different Pronunciation for word "Tilak"

Once the HMM network has been constructed, it can be input to the decoder module HRec and used to recognize speech input thus generating recout.mlf file containing test words.

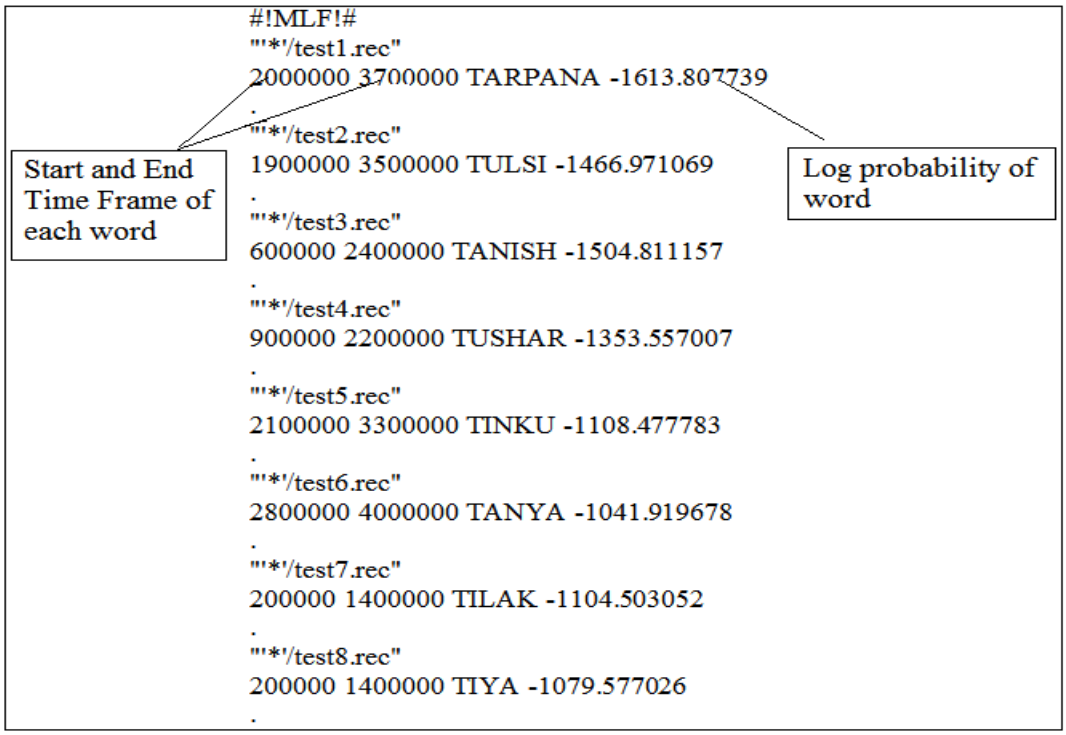


Fig. 4.11: Generated (recout.mlf) file

4.1.5 PHONEME GENERATION

Lastly, phoneme sequence was obtained from the test words in **recout.mlf** file. As a result **phonemes.mlf** file was generated that contained all the phoneme sequences. For this pronunciation dictionary **dict** and edit script mkphones0.led was used as arguments for HLEd tool of HTK.

```
HLEd -A -D -T 1 -l '*' -d dict -i phonemes.mlf mkphones0.led
recout.mlf
```

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

Fig. 4.12: Edit script (mkphones0.led)

In this edit script the expand **EX** command replaces each word in **recout.mlf** by the corresponding pronunciation in the dictionary file **dict**. The **IS** command inserts a silence model sil at the start and end of every utterance e.g. for word “tulsi” generated phoneme sequence is "'*'/test2.lab" sil tt u l s i sil . Finally, the delete DE command deletes all short-pause sp labels.

4.2 RESULTS OBTAINED

After testing HTK for various test samples having words containing ੜ or ੜ and ਅ or ਆ for which system is trained in training phase following results are obtained:-

- Both test 1 with 45 test words and test 2 with 36 words (containing ਅ or ਆ) and words obtained, after testing, with correct and incorrect phoneme sequences are represented by following bar graphs.

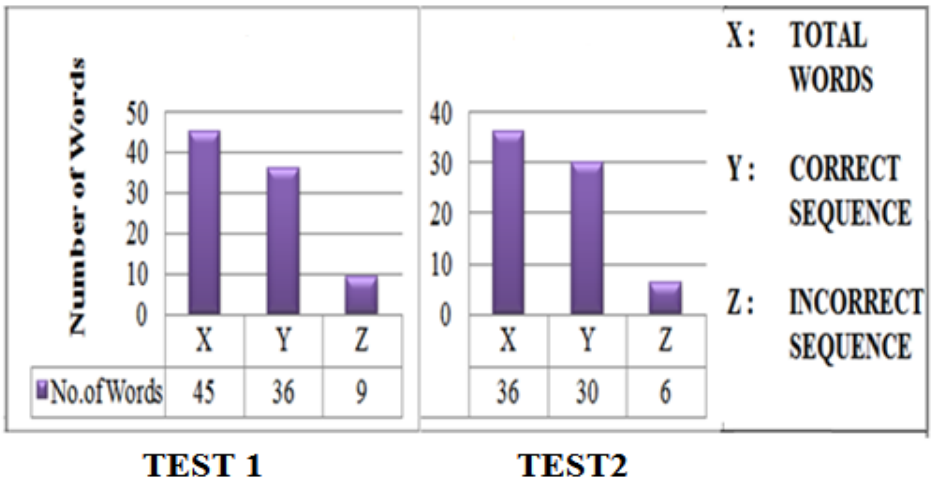


Fig. 4.13: Test Samples for words containing ਅ or ਆ

- Number of words in Test Samples containing ਤ or ਟ with total 28 and 33 words and no of correct and incorrect phoneme sequences obtained are shown by following bar graph.

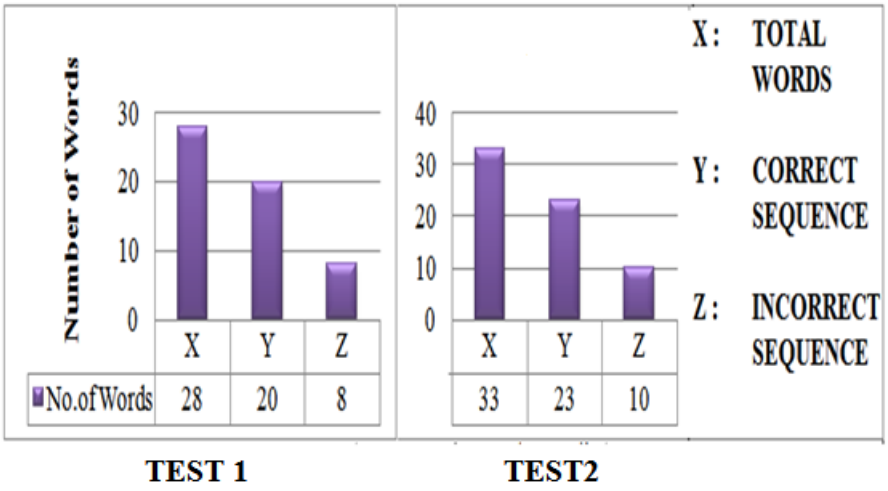


Fig. 4.14 Test Samples for words containing ਤ or ਟ

- Comparisons of overall accuracies for both sets of test i.e. one containing words with ਤ or ਟ and one with ਅ or ਆ is depicted in following cylindrical bar graph.

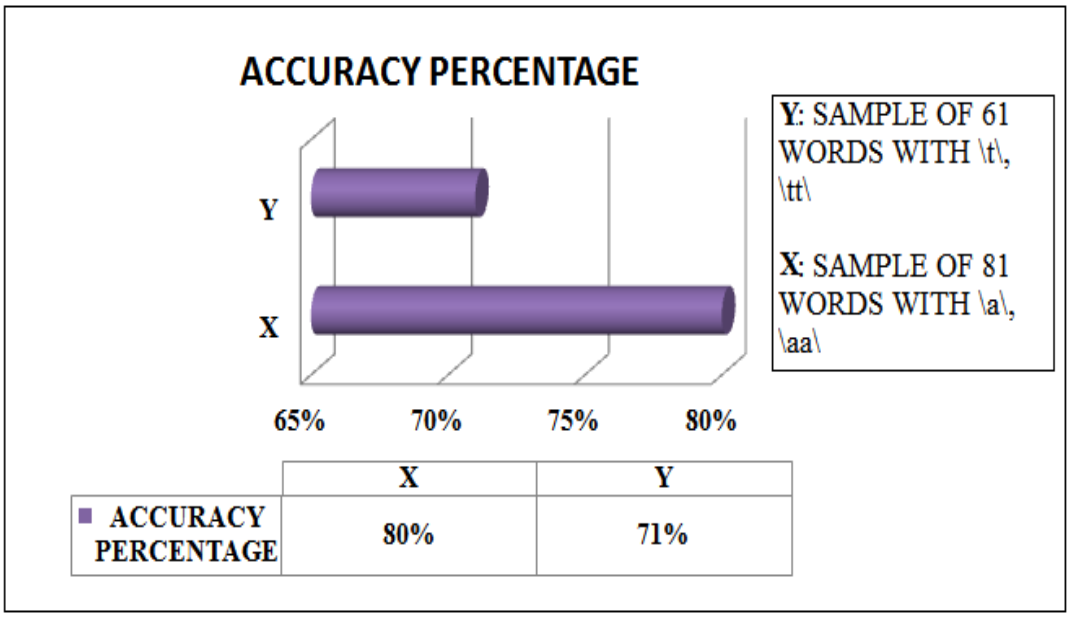


Fig. 4.15: Comparison of Overall Accuracies

As a result of testing and training this whole HTK (HMM Toolkit) based system, a sequence of phonemes were generated at last. These phonemes were further used in

next phase wherein they were concatenated using Concatenative approach in mobile based application developed using .Net platform in Visual Studio to generate natural and intelligible speech sounds.

These signals uttering caller's name in Punjabi language that was stored as English text in phone contacts were obtained.

A MLF (Master Label File) **phonemes.mlf** was generated by HLed tool that contained the correct phoneme sequence, amongst various other alternatives, of the test word contained in **recout.mlf** file which was initially produced by HVite tool of HTK by making use of dictionary **dict** and word network file **wdnet**. Phonemes of different test words were arranged in different .lab files according to the input test samples.

Through HTK correct sequences of phonemes is generated to a greater extent and satisfied results are obtained. Further rule based approach is used to formulate certain rules for generating phoneme sequences of words whose sequences are not correctly produced by HTK.

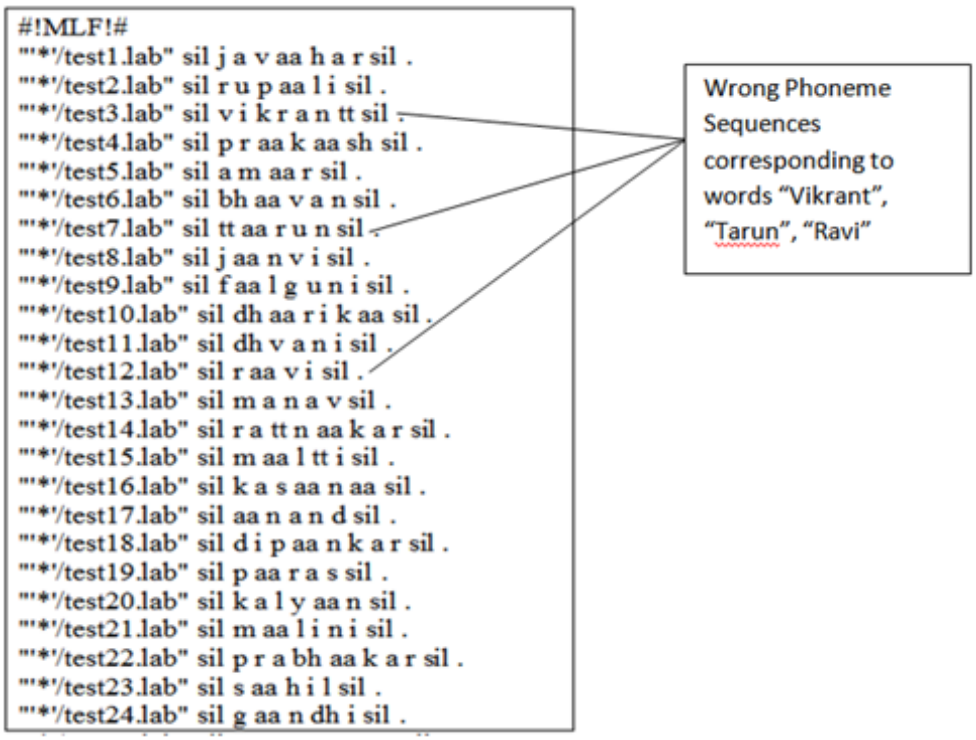


Fig. 4.16 (phonemes.mlf) Phoneme File Generated

CHAPTER -5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

This thesis has presented a new approach to develop TTS (Text-to-Speech) synthesis system that produces Punjabi speech from English text, by describing the segmentation of Punjabi speech corpus using HTK (HMM Toolkit).

Initially, numerous uses of speech synthesizers for uneducated and visually handicapped people and various speech synthesis approaches (Articulatory synthesis, Concatenative Synthesis, Formant Approach) are analyzed and it is being concluded that HTS (HMM based TTS) synthesis approach is a flexible and more adaptive approach for speech synthesis and concatenative approach is useful for producing natural speech sounds. Therefore in order to develop Punjabi TTS to map English text in form of caller's name into Punjabi speech for visually impaired people and for those who are illiterate but can understand spoken Punjabi language, HMM based technique is used to segment speech corpus containing words into smaller units i.e. phonemes that are produced correctly to a greater extent. Further concatenative approach is used to concatenate these phonemes to produce speech.

In order to segment speech corpus into correct sequence of phonemes with the help of hidden markov models and feature parameters extracted from corpus, HTK is used. By analyzing the implementation of HTK in various phases following points are concluded:-

- With the help of HTK, the phoneme sequences are generated by re-estimating HMMs themselves for each phone based on maximum likelihood criteria that provides high accuracy.
- In case of different speaking styles, different speakers, different pitch, emotions etc. a larger database is required in concatenative approach for each

speaker but in HMM based approach these attributes can be obtained in speech by various adaptation techniques.

- Use of context-dependent triphones models improves performance as it is easier to search for monophones based on right and left contexts instead of searching it alone.

- HTK uses Viterbi decoding to find most probable state sequence of phonemes through given feature vectors and uses spectral features (MFCC) to re-estimate parameters of acoustic models instead of using vocal tract frequencies (as in Formant Synthesis) and modeling speech production system directly (as in Articulatory Synthesis).

- TTS system thus developed by using these phoneme sequences generate natural and intelligible Punjabi speech sounds that is helpful for people especially those who are sightless and those belonging to Punjab region and understands Punjabi instead of English.

- Segmentation through HTK involves large number of steps and creation of various intermediate files. Processing of various files and data is tedious and time consuming.

- Due to the absence of proper infrastructure like windows mobile SDKs, studio environment for recording voices by professionals etc., the efficiency of system degraded.

- Command line interface of HTK though provides information about all intermediate steps but is not very user-friendly as compared to graphical user interface.

- Concatenative approach of synthesis is though has a constraint of larger databases and is inefficient in case of different speaking styles but is very easy to follow.

- Due to small speech corpus comparatively less efficiency is achieved that can be improved by using and annotating larger database covering a larger

phoneme set of Punjabi through more words so that more improvised triphones models can be generated.

5.2 FUTURE SCOPE

TTS systems due to their usage and functionality are an important area of research. The TTS system developed using concatenation of phonemes obtained by HMM based segmented speech corpus requires more improvements and development to inculcate more functionality.

- Prosody viz. emotions like anger, happiness, sadness etc. and variations viz. male/ female/child voice are not analyzed in the present work. These need to be inculcated so as to produce high quality speech having more naturalness.

- In next phases, system can be trained and used to remove conflicts regarding phoneme sequences of words containing ਧ, ਢ and ਥ, ਠ and efficiency can be further improved by by improvising context-dependent phone models via recording, annotating more Punjabi speech data and applying filters using custom rules/ procedures.

- This application can also be extended for other mobile functions like reading out text messages etc. and for web browsers to make them more interactive and enhance man-machine communication.

- HTS approach for developing TTS systems can be extended for various other Indian languages and for implementing changes in voice characteristics and speaking styles of synthesized speech with the help of speaker adaptation technique developed for speech recognition.

- Sound signals produced in the end are in wave format. These can be extended to other formats like mp3, .wmp (Window Media Player) etc. as required by user with the help of other techniques.

- The speech synthesizer can be integrated with Punjabi OCR (Online Character Recognition) output to read handwritten text also which will be scanned and converted to speech.

REFERENCES

1. Allen J, Hunnicutt S, Klatt D. *From text-to-speech: the MITalk system*. Cambridge University Press, Inc., Cambridge, 1987.

2. Basu, A., Sen, D., Sen, S., Chakraborty, S., An Indian Language Speech Synthesizer – Techniques and Applications. in *Proceedings of National Systems Conference*, (2003), 217-223.

3. Blunsom, P., Hidden Markov Models, The University of Melbourne, Department of Computer Science and Software Engineering, 19th August 2004, URL:http://www.cs.mu.oz.au/460/2004/materials/hmm-tutorial.pdf

4. Borkar, S.P. 2006. *Text To Speech System for Konkani (Goan) Language*. Master's Thesis. Rajarambapu Institute of Technology.

5. Carlson, R., Models of Speech Synthesis. in *Colloquim on Human-Machine Communication by Voice*, (1993), 8-9.

6. Dutoit, T., 1996. High Quality Text-To-Speech Synthesis: an Overview. *Journal of Electrical & Electronics Engineering*, 17(1), 25-37.

7. Donovan, R.E., Woodland, P.C., 1999. A hidden Markov-model-based trainable speech synthesizer. *Computer Speech and Language*, 00, 1–19.

8. Fant, G. *Acoustic theory of speech production.* Mouton & Co, The Hague, 1960.

9. Gera, P. 2006. *Text-To-Speech Synthesis for Punjabi Language*. M.Tech Thesis. Thapar University.

10. Kaur, A. and Singh, T., 2010. Segmentation of Continuous Punjabi Speech Signal into Syllables. *Lecture Notes in Engineering and Computer Science*, 2186(1), 598-601.

11. Kesarkar, M.P. 2003. *Feature Extraction for Speech Recognition*. Seminar Report. Indian Institute of Technology.

12. King, S., 2011. An introduction to statistical parametric speech synthesis. *Sadhana - Engineering Science,* 36(5), 837-852.

13. Klatt D., 1987. Review of Text-to-Speech Conversion for English. *Journal of the Acoustical Society of America*, 82 (3), 737-793.

14. Kumar, K. and Aggarwal, R. K., 2011. Hindi Speech Recognition System using HTK. *International Journal of Computing and Business Research*, 2(2), 2229-6166.

15. Lemmetty, S. 1999. *Review of Speech Synthesis Technology*. Master's thesis. Helsinki University of Technology.

16. Maia, R., Zen, H., Tokuda, K., Kitamura, T. and Resende Jr., F., Towards the development of a Brazilian Portuguese text-to-speech system based on HMM. in *Proceedings of Eurospeech*, (2003), 2465–2468.

17. Martincic-Ipsic, S. and Ipsic, I., 2006. Croatian HMM-based speech synthesis. *Journal of Computing and Information Technology*, 14 (4), 307–313.

18. Masuko, T. 2002. *HMM based speech synthesis and its applications.* Ph.d Thesis. Tokyo Institute of Technology.

19. Prudon, R. and d'Alessandro, C., A selection/concatenation TTS synthesis system: Databases development, system design, comparative evaluation. in *4th Speech Synthesis Workshop*,(2001).

20. Rabiner, L.R. and Juang, B.H. 1986. An Introduction to Hidden Markov Model. *IEEE ASSP Magazine,* 3(1), 4-15.

21. Raitio, T. 2008. *Hidden Markov Model Based Finnish Text-to-Speech System Utilizing Glottal Inverse Filtering*. M.Sc. Thesis. Helsinki University of Technology.

22. Shirbahadurkar, S.D. and Bormane, D.S., Marathi Language Speech Synthesizer Using Concatenative Synthesis Strategy (Spoken in Maharashtra,

India). in *Second International Conference on Machine Vision*, (2009), 181-185.

23. Singh, P. 2005. *Development of A Punjabi Text-To-Speech Synthesis System*. M.Tech Thesis. Punjabi University.

24. Singh, P. and Lehal, G. S., Text-to-Speech Synthesis system for Punjabi language. in *Proceedings of International Conference on Multidisciplinary Information Sciences and Technologies,* (2006), 388-391.

25. Tokuda, K., Zen, H. and Black, A.W., An HMM-based speech synthesis applied to English. in *IEEE Workshop in Speech Synthesis*, (2002).

26. Yoshimura, T., Tokuda, K., Masuko, T., Kobayashi, T. and Kitamura, T., Duration modeling for HMM-based speech synthesis. in *Proceedings of International Conference on Spoken Language Processing*, (1998), 29–32.

27. Yoshimura, T., Tokuda, K., Masuko, T., Kobayashi, T. and Kitamura, T., Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis. in *Proceedings of Eurospeech*, (1999), 2347–2350.

28. Young, S., Evermann, Gales, M., Hain, T., G., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povay, D., Valtchev, V. and Woodland, P. *The HTK Book Version 3.4*. Cambridge University Press, Cambridge, 2006.

29. Zen, H., Toda, T., Nakamura, M. and Tokuda, K., An overview of Nitech HMM-based speech synthesis system for Blizzard Challenge 2005. in *Proceedings of Interspeech2005 (Eurospeech)*, (2005), 93-96.

RESEARCH PAPERS

RESEARCH PAPER ACCEPTED

Goel, A., Bansal, D., Jindal, K., 2012. Grapheme to Phoneme Conversion for Punjabi Language. *International Journal of Science, Technology & Management*, 3(1).

RESEARCH PAPER COMMUNICATED

- Bansal, D., Goel, A., Jindal, K., 2012. Punjabi Speech Synthesis System Using HTK. communicated in *The International Journal of Information Science & techniques*.
- Bansal, D., Goel, A., Jindal, K., 2012. Punjabi Speech Synthesis System Using HTK. communicated in *Speech Communication*.