A Unified Approach to Speech Synthesis in Indian Languages

A THESIS

submitted by

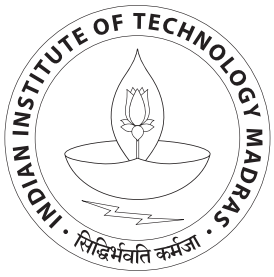
ARUN BABY

for the award of the degree

of

MASTER OF SCIENCE

(by Research)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS

December 2018

THESIS CERTIFICATE

This is to certify that the thesis titled **A Unified Approach to Speech Synthesis in Indian Languages**, submitted by **Arun Baby**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Science (by Research)**, is a bonafide record of the research work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Hema A. Murthy
Research Guide
Dept. of Computer Science
& Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 10 Dec, 2018

ACKNOWLEDGEMENTS

This thesis has been an enriching learning experience for me at academic and personal levels. I am extremely thankful to my advisor Prof. Hema A Murthy, for her sustained support and guidance throughout the period of the study. Discussions with her have been both motivating and thought provoking. Her constructive comments and attention to details have contributed to making this study more nuanced.

I would like to especially thank Jeena J Prakash, Anju Leela Thomas, Anusha Prakash, Rupak Vignesh, Nishanthi N L and other members of SMT Lab for their invaluable help, especially for conducting listening tests. They kept me sane and always encouraged me to look on the bright side of life.

I take this opportunity to thank my parents, P P Baby and Mercy Baby for their support and encouragement.

A work of this kind would not have been possible without the financial aid of the Ministry of Electronics and Information Technology (MeitY), Government of India. I am grateful for this. I would like to thank IIT Madras for giving me an opportunity to conduct this study.

Arun Baby

ABSTRACT

KEYWORDS: Text-to-speech synthesis, Indian languages, unified framework, spectral cues, deep neural networks, segmentation

India is a country with 22 official languages (written in 13 different scripts), 122 major languages and 1599 other languages. These languages come from $5-6$ different language families of the world. It is only about 65% of this population that is literate, that too primarily in the vernacular. Speech interfaces, especially in the vernacular, are enablers in such an environment. Building text-to-speech (TTS) systems for such a diverse country necessitates a unified approach. This research work aims to build Indian language TTS systems in a unified manner by exploiting the similarities that exist among these languages. Specifically, the focus is on two components of the TTS system, namely, text parsing and speech segmentation.

Parsing is the process of mapping graphemes to phonemes. Indian languages are more or less phonetic and have about $35-38$ consonants and $15-18$ vowels. In spite of the number of different families which leads to divergence, there is a convergence owing to borrowings across language families. A Common Label Set (CLS) is defined to represent the various phones in Indian languages. In this work, a uniform parser is designed across all the languages capitalising on the syllable structure of these languages.

Segmentation is the process of finding phoneme boundaries in a speech utterance. The main drawback of the Gaussian mixture model - hidden Markov model (GMM-HMM) based forced-alignment is that the phoneme boundaries are not explicitly modeled. State-of-the-art speech segmentation approach for speech segmentation in Indian languages is hybrid segmentation which uses signal processing cues along with GMM-HMM framework. Deep neural networks (DNN) and convolutional neural networks (CNN) are known for robust acoustic modelling. In this work, signal processing cues, that are agnostic to

speaker and language, are used in tandem with deep learning techniques to improve the phonetic segmentation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ASR	Automatic Speech Recognition
CART	Classification And Regression Tree
CLS	Common Label Set
CNN	Convolutional Neural Network
DMOS	Degradation Mean Opinion Score
DNN	Deep Neural Network
GD	Group Delay
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
HTS	Hidden Markov model-based Speech Synthesis System
IVS	Inherent Vowel Suppression
MFCC	Mel Frequency Cepstral Coefficients
MGC	Mel Generalized Cepstral coefficients
MLSA	Mel Log Spectrum Approximation
PC	Pairwise Comparison
PLP	Perceptual Linear Prediction
SBSF	Sub-Band Spectral Flux
SF	Spectral Flux
SPSS	Statistical Parametric Speech Synthesis
STE	Short-Term Energy
SVM	Support Vector Machine
TTS	Text-To-Speech
USS	Unit Selection Synthesis
UTF	Unicode Transformation Format
WSF	Window Scale Factor

CHAPTER 1

Overview of the Thesis

1.1 Introduction

India is the second most populous nation with over 1.29 billion people. It has 22 official languages (written in 13 different scripts), 122 major languages, with over 1600 languages/dialects [Wikipedia, 2018b]. Further, it is only about 74% of this population that is literate, that too primarily in the vernacular [Wikipedia, 2018d]. Only 12.16% is literate in English thus marginalizing most of the Indian society [Wikipedia, 2018c]. Speech interfaces, especially in the vernacular, are enablers in such an environment. In this thesis, the focus of effort is on developing text-to-speech (TTS) systems for Indian languages.

The objective of this work is to build technology that will enable the building of TTS systems in any Indian language quickly. Text to speech synthesis is the process of converting an arbitrary input text to its corresponding speech output. The state-of-the-art TTS systems in Indian languages use syllables or phonemes as subword unit. Three major components involved in building a TTS system are text parsing, speech segmentation, and speech modeling. The text processing component converts graphemes into a sequence of phonemes. The segmentation component converts the speech to accurate time-aligned sub-word units. The speech generation component uses the produced sequence of sub-word units and the segmented speech to generate the speech waveform. Determining the appropriate sequence of sounds and segmenting the speech into accurate sub-word units are very crucial for generating natural and intelligible speech. The objective of this work is to build technology that will enable the building of robust high-quality TTS systems quickly, by focusing on two major components, text parsing, and speech segmentation.

1.2 Overview of the work

In this thesis, different methods to improve the TTS systems are devised. Since parsing and segmentation are the two components which determine the naturalness and intelligibility of the TTS systems, the thesis concentrates on these sub-systems.

Parsing is the process of mapping graphemes (written text) to a sequence of phonemes. As discussed, Indian languages come from 5-6 different language families of the world. Most of these languages have their own script in Unicode transformation format (UTF-8). This makes parsing for text to speech systems for Indian languages a difficult task as each language needs a different parser. However, in spite of the number of different families which leads to divergence, there is a convergence owing to borrowings across language families. Most importantly Indian languages are more or less phonetic and consist broadly of about 35-38 consonants and 15-18 vowels. Although Indian languages are phonetic, i) agglutination[1] leads to addition of graphemes that do not have a corresponding sound in the waveform, ii) schwa deletion[2] mid word is common, which leads to graphemes that do not relate to the acoustics of the signal. Parsing primary addresses these issues. In a previous work, a common label set (CLS) is developed to represent the various phones in Indian languages [Ramani et al., 2013]. In the current work, an attempt is made to unify these languages based on the existing similarities. A uniform parser is designed across many Indian languages. The proposed parser converts UTF-8 text to CLS, applies parsing rules and generates the corresponding phoneme sequences.

Automatic detection of phoneme boundaries is an important sub-task in building speech processing applications, especially TTS systems. Segmentation of speech into accurate time-aligned phonetic transcriptions plays a vital role in building robust speech systems. Manual labeling for a huge multi-lingual corpus is time-consuming and error-prone which warrants automatic procedures. Machine learning approaches require a huge amount of data to learn boundaries accurately. However, Indian languages are digitally low resource. Signal processing cues, which are agnostic to speaker and language, are

[1]The process of combining words that are formed by stringing together morphemes, detailed in Section 3.6.5.

[2]The implicit mid-central vowel, in each consonant of the script, is obligatorily deleted in certain context while uttering, detailed in Section 3.6.1.

good at finding syllable boundaries with less data. In this thesis, a hybrid approach which combines the power of deep neural networks and signal processing cues is discussed.

1.3 Scope of the thesis

Text parsing and speech segmentation are crucial for most of the speech processing applications. This is especially important in case of text-to-speech (TTS) systems. Even though this can be applied in many speech applications[3], this work focuses on generic domain TTS systems for Indian languages. The TTS systems developed in this thesis are based on unit selection synthesis (USS) and hidden Markov model based speech synthesis (HTS). These techniques are detailed in Section 2.4.

1.4 Key contributions

The major contributions of this work are as follows:

- A unified parser which can parse words across different Indian languages to a common label set. This can be extended to any language by mapping letters of alphabet to common label set and defining language-specific rules for the new language.
- Improving the speech segmentation accuracy using deep neural networks in tandem with spectral cues for Indian languages TTS systems using:
  - Iterative boundary correction at utterance level
  - Boundary correction at sub-utterance level

1.5 Organisation of the thesis

The thesis is organised as follows. Chapter 2 gives an overview of TTSes for Indian languages. Chapter 3 details the need for a unified approach to parsing and how it is developed. In Chapter 4, segmentation of speech signals and the state-of-the-art speech

[3]The performance of automatic speech recognition system for Indian languages has improved with the use of unified parser and neural network based speech segmentation [Baby et al., 2018].

segmentation approaches for Indian languages are discussed. Chapter 5 details the proposed approach for segmentation using deep neural networks (DNN) in tandem with spectral cues. Chapter 6 presents conclusions and scope of future works.

CHAPTER 2

Text to Speech Synthesis Systems for Indian Languages

2.1 Introduction

Speech has evolved as the most prominent means of communication between humans and computers as a mode of interaction [Shrishrimal et al., 2012]. Human beings have always been fancied by the idea of creating machines that can communicate with them. Since 1960's, research has been done to develop systems that can understand human speech. Speech technology played a crucial role in the development of numerous domains like education, agriculture, healthcare, and government services [Eskenazi, 2009, Plauche et al., 2006, Durling and Lumsden, 2008]. This is predominantly useful in a multilingual society like India which has great linguistic diversity.

The objective of TTS systems is to convert arbitrary text into corresponding speech output. There are two types of TTS systems- domain specific, and vocabulary independent. In case of domain specific, the sentences/words to be synthesised should be in-domain: for example, railway broadcast. In case of vocabulary independent, the input can be any arbitrary text. This thesis works with vocabulary independent TTS systems.

TTS system building consists of data collection, text parsing, speech segmentation, and speech modeling. TTS systems require a large amount of training data. The first step in creating a TTS system is data collection, also known as corpus creation. The training data consists of speech wave files along with the corresponding text transcription. The process of corpus creation is explained below.

2.2 Corpus creation

Corpus is the machine-readable form of a large collection of structured text in written or spoken form [Dash and Chaudhuri, 2001]. The importance of language corpora has been recognised for a long time in many countries [Conrad, 1999, Wichmann and Fligelstone, 2014]. The amount of work in speech domain for Indian languages is comparatively lower than that of other languages. Building a corpus for Indian languages is a time taking process because of lack of digital resources and is difficult because of its linguistic diversity. However, there exists a lot of scope in developing language corpora for Indian languages. The information acquired from the corpora will not only provide advanced resources for TTSes, languages processing tools, etc. but also will be useful for educational purposes and various domains of language research.

The procedures for text collection, text correction (if any), the methodology followed for selecting the speakers and details of the recording process are discussed below.

2.2.1 Text selection/correction

A huge text corpus is a very crucial resource for preparing the training data for building TTS. Collecting transcribed data for Indian languages, which are in general digitally low resource languages, is a herculean task. To accomplish this task, initially, text in various Indian languages are collected from newspapers, websites, blogs, etc with the help of web crawlers. Furthermore, text from different domains like children's stories, literature, science, tourism, etc. are also manually collected to achieve a good coverage. The collected text is manually corrected to get rid of transcription errors if any. Care has taken to ensure that the chosen text is easy to read, covers the most commonly used words and phrases in a language and has maximum syllable coverage. Also code-switched sentences are avoided during text selection. Unlike English, agglutination is very common in Indian languages. We avoid words that are longer than three syllables. This is primarily to avoid the issue of vowel shortening due to agglutination. The collected data is used to record speech. The list of words from the collected text is also used to

generate a pronunciation dictionary used for building speech synthesis system.

2.2.2 Speaker selection

The next phase, after text collection/selection, is to record the data. Two native voice talents (1 male and 1 female speaker) are selected, for each of the regional languages. Proper instructions are given to these speakers in such a way that there are minimal pronunciation errors. Multi-speaker recording for a given language, gender, and type (native/English) will lead to variations in pitch, speed, speech style, tempo, and amplitude. Single speaker data limits the variations and change in voice quality, which is crucial for building a robust TTS system. Apart from these aspects, appropriate voice talent whose voice seems pleasant to listen, as well as amenable to signal processing is chosen. For the ease of speaker, care is taken in every session to keep the context of the text co-relative (unchanged) so that switching is avoided.

2.2.3 Recording

The corrected text[1] is used for recording. The recording is carried out in a special environment which is free from noise and echo. Further, to avoid the fatigue of the speaker, a break is given every 45 minutes. Later the recorded sentences are split at sentence level. Also, measures are taken to maintain same conditions and voice characteristics across the multiple recording sessions. Hence, the type of recording is mono, with a sampling rate of 48 kilohertz (kHz) and the number of significant bits per sample is 16. Non-conforming sentences are re-recorded. The recorded speech files are stored in .wav format in the database to ensure maximum quality.

[1]The text will be around $4000 - 6000$ sentences which is equivalent of 10 hours of speech data.

2.3 Overview of TTS

Once the speech corpora is created, a TTS system is developed using the corpora, that can artificially synthesize any text. Text pre-processing, text parsing, speech segmentation, training and synthesizing are the different phases of developing a TTS system. Figure 2.1 shows the block diagram of a TTS system. The main parts of TTS systems are parser, segmentation unit, and the TTS modeling unit.

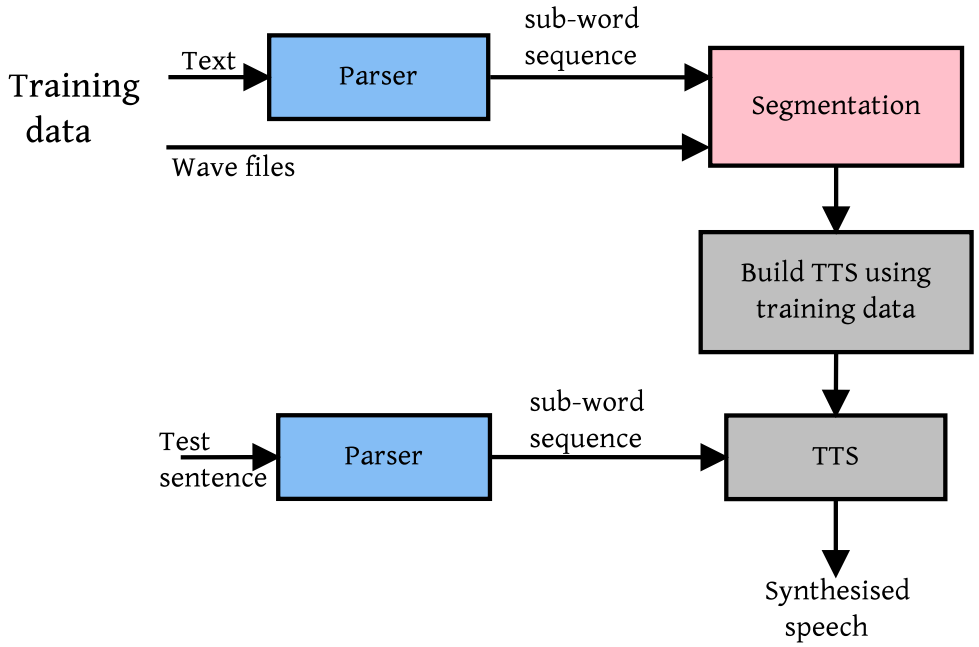


Figure 2.1: Block diagram of text-to-speech synthesis system

As discussed earlier, the training database consists of speech utterances and text transcriptions at the sentence level. However, TTS systems are built at sub-word unit level, mostly phones. The parser, also known as the front-end processor, converts graphemes to phonemes. It is very crucial to obtain proper grapheme to phoneme sequence as the correctness of TTS output depends on this. The general approach is to develop individual parsers for each language. This is a tedious task and requires language experts to figure out the rules initially.

Segmentation is the process of finding the boundaries of phones or syllables in the speech waveform. It takes the phonetic sequence as given by the parser along with the corresponding speech utterance and provides the time-aligned phonetic sequence. The

process can be performed manually or automatically. In case of automatic approaches, hidden Markov models (HMM) are commonly used to obtain appropriate alignments. These boundaries may not be very accurate.

Once the segmentation is performed, TTS system can be built using the segmented data which in turn can be used to synthesize artificial speech. Two approaches - unit selection synthesis (USS) and HMM-based speech synthesis system (HTS) for Indian languages, are discussed in detail in Section 2.4. During synthesis, text sentence is parsed and the phonetic sequence is given to the TTS system to synthesize the speech output.

2.4 State-of-the-art techniques to build TTS systems in Indian languages

The two state of the art techniques for building Indian language TTSes are unit selection synthesis (USS) [Hunt and Black, 1996] and statistical parametric speech synthesis (SPSS)[2] [Zen et al., 2009]. These techniques are discussed briefly in the following subsections.

2.4.1 Unit selection synthesis (USS)

In case of USS, the segmented data after annotation is stored in a database. For synthesis, first, the sub-word sequence is obtained using a parser. The sub-words from the database are concatenated according to the cost criteria, which is a combination of target and concatenation costs[3] [Hunt and Black, 1996]. Syllables are observed to be the best sub-word unit for Indian languages [Kishore and Black, 2003, Patil et al., 2013].

The synthesis quality of USS is natural but discontinuities at the concatenation points are perceivable. Another disadvantage of USS is the database size which is normally huge and has to be stored during synthesis as well.

[2]HTS is the most common approach of SPSS for Indian languages

[3]Target cost depends on the difference between the target unit and the unit in database while concatenation cost depends on quality of concatenation.

2.4.2 HMM-based speech synthesis system (HTS)

In case of parametric synthesis, the model is described as being parametric because it describes the speech using specific parameters. This is different from USS where the speech is obtained from the stored database. The most popular parametric method is HTS. Phone-based HTS systems are used in common for Indian languages.

HTS can produce intelligible speech with small amount of data, owing to the HMM based approach learning a generative model. In USS we need an example for every context. HTS uses a source-filter model for speech production [Fant, 1968]. Two models, acoustic and duration, are trained in HTS systems. The features used for acoustic model in HTS are spectral and excitation parameters. 105-dimensional spectral parameters that contain the mel-generalized cepstral (MGC) features (35) along with their velocity and acceleration values are used. $\log f0$ values are used for excitation parameters. These parameters are modeled using HMMs. The model parameters are estimated using the maximum likelihood criterion:

$$\hat{\lambda} = \arg \max_{\lambda} p(O|W, \lambda), \tag{2.1}$$

where λ represents the model parameters, $\hat{\lambda}$ the re-estimated parameters, O the extracted features from training data and W the transcriptions corresponding to the training data.

To incorporate the continuity of speech output, pentaphone models are trained[4]. Phonetic labels for the context information is obtained using the Festival framework [Black et al., 1998]. The context-dependent HMMs are trained using context independent HMMs that are trained initially. These context-dependent monophones are clustered for re-estimating because of the training data sparsity. The duration and acoustic parameters are modeled separately.

During synthesis, labels are obtained using the festival framework which is then used to find the appropriate context-dependent HMM sequence. Duration of a phone is obtained from the corresponding duration model. Spectral and excitation parameters are generated

[4]5-state single mixture HMMs are most commonly used for pentaphone modeling. For Indian languages, each having around 50 phonemes, the number of distinct models can go up to 125k.

such that the output probability is maximised:

$$\hat{o} = \arg \max_{o} p(o|w, \hat{\lambda}), \tag{2.2}$$

where o represents the speech parameters, $\hat{o}$ the re-estimated speech parameters, $\hat{\lambda}$ represents the model, and w the transcription of the test sentence.

Using these obtained features, speech is reconstructed with the help of mel log spectral approximation (MLSA) filter [Imai et al., 1983].

For HTS-STRAIGHT systems, band-aperiodicity (BAP) parameters [Kawahara et al., 2001] are also modeled along with MGC and $\log f0$ features. STRAIGHT is a speech analysis, modification, and re-synthesis system [Kawahara et al., 1999]. For TTS applications, STRAIGHT is used as a VOCODER [Dudley, 1939].

Since the modeling of the parameters is performed in HTS systems, the size of the TTS systems is small compared to USS systems. Also, the discontinuities in synthesised speech which is predominant in USS is not there is HTS systems as the parameters are modeled.

2.5 Challenges in Building TTSes for Indian languages

Indian languages are digitally low resource. Building TTS systems for low resource languages is a very difficult task. For languages like English and Chinese, the amount of digital data available is large [Godfrey et al., 1992, Paul and Baker, 1992, Schultz et al., 2013]. This accounts for robust TTS systems for English and Chinese [Rabiner, 1989, Hinton et al., 2012, Yuan et al., 2013]. While building speech systems for Indian languages, the scarcity of data is a primary concern. Attempts have been made to bootstrap training data for low resource languages [Cui et al., 2012]. Lack of accurate phone level alignment of the training data is another concern. Since the amount of data is less, machine learning models perform poorly.

Building a separate TTS system for each Indian language is time-consuming and

expensive. The dataset prepared can have transcription errors and speakers may not utter the speech correctly all the time. Due to the continuity of speech, re-syllabification can occur at times. Parsing of text to obtain the phoneme sequence that matches the acoustics of the speech signal is a challenge. Obtaining accurate phone level speech alignment from the data is also crucial in building robust TTS systems. Many of these challenges may seem to be easy for humans, but while training a machine these are difficult challenges. Moreover, lack of adequate amount of accurately annotated data makes it even more tough.

The objective of this work is to exploit similarities across languages to aid faster system building. Owing to the geographical proximity of languages and intermixing among cultures, there is significant borrowing across languages. A natural design choice is to make language-independent modules as much as possible. The thesis mainly focuses on improving the accuracy of parsing and segmentation for Indian language TTS systems.

2.6 Summary

In this chapter, an overview of the TTS systems is discussed. Different phases like corpus creation, TTS building and the challenges for TTS systems in Indian languages are detailed. The state of the art techniques for Indian language TTS systems is also discussed. In the next chapter, a unified approach in parsing languages is detailed.

CHAPTER 3

A Unified Approach to Parsing Indian Languages

3.1 Introduction

The objective of text to speech (TTS) synthesis system is to convert an arbitrary input text to its corresponding speech output. Text processing and speech generation are two major components of a TTS system. The text processing component converts graphemes into a sequence of phonemes while the speech generation component uses the produced sequence of phonemes to generate speech waveform. Determining the appropriate sequence of sounds is very crucial for natural and intelligible speech generation. Table 3.1 shows some examples of syllable and phone level parsing for different languages. The phoneset used in the table is detailed in Section 3.4.

Table 3.1: Sample parsing

Word	**Syllables**	**Phones**
बुताना	बु ता ना	b u t aa n aa
अक्षरधाम	अक् ष र धाम	a k sx a r a dh aa m
കൂടിയാകുമ്പോൾ	കൂ ടി യാ കുമ് പോൾ	k uu tx i y aa k u m p oo ln
தரையெங்கும்	த ரை யெங் கும்	t a r ai y e ng k u m

Traditional approaches in converting text to speech for a given language make use of language specific parsers. Such approaches use specific rules of a given language and build parsers that are highly customised. This makes the task of creating individual parsers for new languages difficult.

Parsers that work for more than one language focus on structurally related languages such as English and French or English and German [Copestake and Flickinger, 2000]. Bilingual parsers built in Indian multilingual context are also available [Raina et al., 2004]. This work introduces a unified parser that can handle Indian languages which are free-word-order and morphologically rich. The main challenges are finding the rules

for different languages and incorporating the context-sensitive rules. The unified parser systematically identifies the invariant properties of Indian languages first. The UTF-8 text is converted to a sequence of labels. A CLS is first defined across all the languages. Rules that are peculiar to a language are treated as exceptions. Lex and Yacc [Levine et al., 1992] stands in good stead to build rule-based language parsers as these employ rule-based method for token matching. This work tries to capture the similarities and resolves the differences in rules across multiple Indian languages so that lexical rules can handle occurrences of all native sentences and pass it to a synthesizer.

3.1.1 Characteristics of Indian Languages

Most of the Indian languages can be broadly classified into two language families:

- Indo-Aryan languages
- Dravidian languages

Indo-Aryan is the largest and is spoken mostly in North India while Dravidian is predominant in South. These classes of languages share some common features. The geographical proximity of the regions, where these languages have been spoken, have resulted in significant borrowings too [Prakash et al., 2014]. Indian languages are characterized by character set termed as aksharas [Raghavendra et al., 2008a]. Aksharas are the fundamental linguistic units of the writing system in Indian languages [Lavanya et al., 2005]. Using the properties of aksharas, syllable boundaries can be marked at vowels at regular intervals for a given sequence of phones. This finding is typically followed in building TTS systems for Indian languages [Kishore et al., 2002].

Indian languages are syllable-timed and a large number of syllables are common across Indian languages [Raghavendra et al., 2008a]. Approximating to the nearest syllable is possible even if the syllable as such is not available [Raghavendra et al., 2008b]. Accounting for the acoustic-phonetic properties of different languages, this thesis primarily focuses on the generation of phonemes sequences of the form C*VC*, where C is a consonant and V is a vowel.

3.2 Related Work

Traditional approaches include the use of pronunciation dictionaries and language specific parsers. pronunciation dictionaries are created manually. But this has the drawback that a lot of manual intervention is needed and is prone to human errors. Moreover, the vocabulary is open but a dictionary is closed. This makes it not scalable. Also, it fails with out-of-domain words. It is difficult to get all the words in a language mapped in a pronunciation dictionary. Separate dictionaries are needed for all languages. A lot of language expertise is required to get all the words in a pronunciation dictionary.

Language-specific rule-based parsers are developed by abstracting the rules in a language. The main drawback is that it is specific to one language. Moreover, the parsers developed can have rules that are contradictory. This results in erroneous parsing of a subset of words. An example in Hindi where words are parsed to:
1) ताजमहल(t-aa-j-a-m-a-h-a-l-a) → t-aa-**j-m-a**-h-a-l Correctly parsed
पागलपन(p-aa-g-a-l-a-p-a-n-a) → p-aa-**g-l-a**-p-a-n Wrongly parsed
2) ताजमहल(t-aa-j-a-m-a-h-a-l-a) → t-aa-**j-a-m**-h-a-l Wrongly parsed
पागलपन(p-aa-g-a-l-a-p-a-n-a) → p-aa-**g-a-l**-p-a-n Correctly parsed

Due to the contradictory rules, the words can be either parsed to 1 or 2 as shown above. The application of the same rule in both words results in wrong parsing of one of these. In both cases, a subset of words will get parsed wrongly. This is mainly due to left-to-right processing of the words which is explained in Section 3.6.1.

3.3 Phonetics

Phonetics is the scientific study of human speech sounds. It has three branches based on production (articulatory phonetics), transmission (acoustic phonetics) and perception (auditory phonetics) of speech sounds. Articulatory phonetics, a subfield of phonetics, explains how speech is produced using different physiological structures. The main two classes of sounds are vowels and consonants.

Speech sounds of any language can be identified by place and manner of articulation [Wikipedia, 2018a]. Consonants can be mainly classified according to the manner of articulation as stops, nasals, fricatives, and affricates and place of articulation as bilabials, dentals, alveolar, palatal, velar etc. [Ladefoged and Johnson, 2014, Wikipedia, 2018e]. Figure 3.1 shows an example with fricative(s), affricate(j), stop consonant(k), and nasal(n) (in green, violet, orange, and yellow colors respectively) with its corresponding waveform and spectrogram. From the spectrogram, it is clear that the acoustic properties of these sounds are very different.

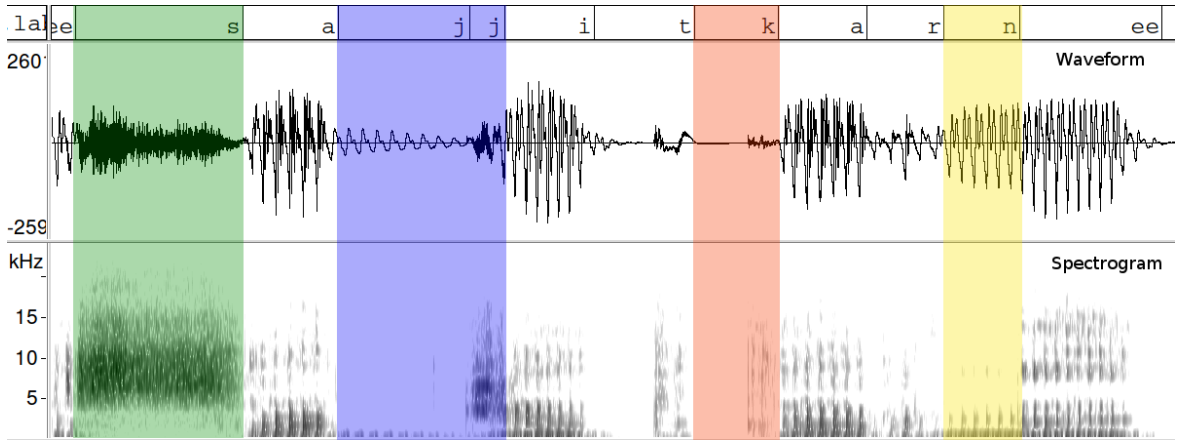


Figure 3.1: An example of Hindi utterance सज्जित करने showing different categories of consonant sounds. Fricative(s), affricate(j), stop consonant(k), and nasal(n) are marked in green, violet, orange, and yellow colors respectively.

Acoustic similarity across languages is used to create a common representation for phones in different languages. This is detailed in Section 3.4.

3.4 Common Label Set (CLS)

The acoustic similarity among similar sets of phones of different languages suggests the possibility of a compact and common set of labels [Ramani et al., 2013, Singh, 2006]. The common label set is defined using the Latin 1 script. The common label set uses a standard set of labels for speech sounds that are commonly used in Indian languages. The notations of labels and rules for mapping are detailed in [Ramani et al., 2013]. The

CLS is used extensively in the development of a unified parser. A subset of CLS[1] is shown in Figure 3.2.

	Label	IPA	Hindi	Marathi	Bengali		Tamil	Malayalam	Telugu
					P	G			
Vowels	a	a	अ	अ	–	–	அ	അ	అ
	ax	ɔ	–	ऑ	অ	অ	–	–	–
	aa	a:	आ	आ	আ	আ	ஆ	ആ	ఆ
	i	ɪ, i	इ	इ	ই,ঈ	ই	இ	ഇ	ఇ
	ii	i:	ई	ई	–	ঈ	ஈ	ഈ	ఈ
	rq	-	ऋ,ৠ	ऋ,ৠ	ঋ,ৡ	ঋ,ৡ	–	ഋ	ఋ,ౠ
Stop Consonants	c	ʧ	च	च	চ	চ	ச	ച	చ
	ch	ʧʰ	छ	छ	ছ	ছ	–	ഛ	ఛ
	cx	ʄ	–	च	–	–	–	–	–
	jx	ʤ	–	ज	–	–	–	–	–
Semi-Vowels	y	j	य,य़	य,य़	য়	য	ய	യ	య
	l	l	ल	ल	ল	ল	ல	ല	ల
	lx	ɭ	–	ळ,ৰ	–	–	ள	ള	ళ
	w	ʋ	व	व	–	ওয়	வ	വ	వ
Fricatives	sh	ʃ	श	श	শ,ষ	শ	–	ശ	శ
	s	s	स	स	স	স	ஸ	സ	స
	khq	x	ख़	ख़	–	–	–	–	–
	z	z	ज़	ज़	ज़	–	–	–	–
	f	f	फ़	फ़	–	ফ়	ஃப	–	–

Figure 3.2: A subset of common label set[2]

3.5 Motivation

Since there are many languages in India, developing a separate parser for each language is not a feasible solution. However, the structural similarity inside a language family can be explored to create abstract rules across languages in the same language family. Despite having different scripts there exists a relationship between the orthography and sound that is common in Indian languages. Exploiting this fact a common label set is introduced. In a previous work, a CLS is developed by identifying the acoustic similarity across languages. Using the CLS as baseline, abstracting rules across languages can be exploited to create a generic parser. An alternate approach which will improve the

[1]The complete set is available at https://www.iitm.ac.in/donlab/tts/cls.php

parsing accuracy can be defined to handle the problem with left-to-right parsing (detailed in Section 3.6). Also, unifying the rules across languages will make scaling to a new language easy. This results in a reduction in the amount of language expertise required.

3.6 Unified Parser

The primary task of a parser is to segment the text for speech systems. However, the parser cannot handle raw text as is available on news websites, blogs, documents etc. Normalising the text input by removing the unwanted characters like special characters and emoticons is essential. Once the text is standardized, the next step is language identification. The first character of the word is taken and compared with the Unicode range to detect the language. Identifying whether the language belongs to Indo-Aryan or Dravidian is also vital owing to vast differences in pronunciation. For parsing the word, the sequence of graphemes is mapped to labels in CLS. Having mapped the input to CLS, the next step is to obtain appropriate pronunciation for each of these labels. Although Indian languages are more or less phonetic, occasionally, the one-to-one correspondence between spoken and written form is absent. These exceptions are handled by rules detailed in Section 3.6.4.

The main issue with parsing is the identification of vowel deletion points, syllable boundaries and the manner of applying rules. The unified parser uses the following set of rules.

3.6.1 Schwa deletion rules

Phonetically, schwa is a short neutral vowel sound /*a*/ which is associated with a consonant. For Indo-Aryan languages, the implicit mid-central vowel (schwa), in each consonant of the script, is obligatorily deleted in certain context while uttering. This is known as schwa deletion or inherent vowel suppression (IVS). For example, in the Hindi word कलम, even though it is written as k-a-l-a-m-a, the pronunciation is k-a-l-a-m. Identifying which schwas are to be deleted and which are to be retained makes the process

of schwa deletion complex. This is obvious for a native speaker, but for machine processing this decision depends on language-specific rules. IVS rules are performed on free consonants/semivowels (FCS) in a word. FCS refers to the consonants/semivowels in a word that do not have a vowel sound adjacent in the written form. For example, in the Hindi word कलम(written as k-a-l-a-m-a) *ka* and *ma* are free consonants and *la* is a free semivowel. The traditional rules followed for schwa deletion are given below.

1. Characters present in the first position of a word, never undergo IVS.
   Hindi : कस्कर(k-a-s-k-a-r-a) → **k-a**-s-k-a-r
   Bengali : পরিমাণ (p-a-r-i-m-aa-nx-a) → **p-a**-r-i-m-aa-nx
   Gujarati : મળેલો (m-a-lx-ee-lo) → **m-a**-lx-ee-l-o

2. Characters in final position always undergo IVS.
   Hindi : कहन(k-a-h-a-n-a) → k-a-h-a-**n**
   Bengali : পঙ্কজ (p-a-ng-k-a-j-a) → p-a-ng-k-a-**j**
   Gujarati : મહાન (m-a-h-aa-n-a) → m-a-h-aa-**n**

3. No two successive characters undergo IVS. Hindi : कसरत(k-a-s-a-r-a-t-a) → k-a-**s-r-a**-t
   Gujarati : બરકત(b-a-r-a-k-a-t-a) → b-a-**r-k-a**-t
   Bengali : মনসকতা(m-a-n-a-s-a-k-a-t-aa) → m-a-**n-a-s-k-a**-t-aa
   Gujarati : વનસ્પતિના(v-a-n-a-s-a-p-a-t-i-n-aa) → v-a-**n-a-s-p-a**-t-i-n-aa

4. No two vowels come together.

5. The remaining FCS in a word that is not processed by rules 1 and 2 is processed in left-to-right order. IVS occurs for an FCS if its successor in the word is (i) not the last character of the word or (ii) a vowel other than $'a'$.

Issues with left-to-right parsing

Application of left-to-right order of processing leads to erroneous parsing of a subset of words. Example: Application of the rule yields the following output.

ताजमहल(t-aa-j-a-m-a-h-a-l-a) → t-aa-j-m-a-h-a-l Correctly parsed

पागलपन(p-aa-g-a-l-a-p-a-n-a) → p-aa-**g-l-a**-p-a-n Wrongly parsed (correct parsing is p-aa-**g-a-l**-p-a-n)

ताजमहल is parsed correctly whereas पागलपन is not. Clearly this rule is inadequate to parse both words.

Proposed rules

This work proposes 2 new rules - AB (1) and AB (2) - to solve such parsing problems. These rules mainly work on FCS to parse the words accurately.

AB (1) rule: A free semivowel (eg: ya, ra, la etc) at the second position of a word starting with a vowel never undergoes IVS whereas a free consonant (eg: ka, ca, ta etc) at the second position of a word starting with a vowel always undergoes IVS.

Hindi words : अकबर and असफल follows v-cv-cv-cv structure, but having a free consonant and a free semivowel in its second position respectively.

But अकबर (a-k-a-b-a-r-a) is parsed to a-**k-b-a**-r (vc-cvc)

and असफल (a-s-a-f-a-l-a) is parsed to a-**s-a-f-a**-l (v-cv-cvc)

AB (2) rule: The focus of this rule is on the substring of the word which is not processed by rules 1, 2 and AB (1). The inherent vowel sounds in this substring are named unmarked schwa. The rule proposes lexicographically ordered processing of FCS in this substring. An FCS is processed only if it is preceded by an unmarked schwa and succeeded by a vowel, vowel sound or unmarked schwa in the transliterated form. In this case, the predecessor (unmarked schwa) is deleted. If the successor is an unmarked schwa, it is marked as non-deletable in further iterations (marked schwa). Application of AB rule parses पागलपन and ताजमहल correctly. The process is illustrated in Tables 3.2 and 3.3. In Table 3.3, a^* represents unmarked schwa and â represents marked schwa.

Table 3.2: Pass 1 - Apply Rules 1, 2 and AB (1)

Word	**Transliterated String**	**Rule 1**	**Rule 2**	**AB(1)**
ताजमहल	taa j*a* m*a* h*a* l*a*	NA	taa j*a* m*a* h*a* l	NA
पागलपन	paa g*a* l*a* p*a* n*a*	NA	paa g*a* l*a* p*a* n	NA
अकबर	a k*a* b*a* r*a*	NA	a k*a* b*a* r	a k b*a* r
असफल	a s*a* f*a* l*a*	NA	a s*a* f*a* l	a sa f*a* l

Table 3.3: Pass 2 - Apply AB (2) Rule

Substring	**Iteration 1**			**Iteration 2**			**Iteration 3**		
(unmarked)	**char**	**Action**	**Output**	**char**	**Action**	**Output**	**char**	**Action**	**Output**
ja^* ma^* ha^*	j	No	ja^* ma^* ha^*	m	yes	j mâ ha^*	h	No	j mâ ha^*
ga^* la^* pa^*	g	No	ga^* la^* pa^*	p	yes	ga l pâ	l	No	ga l pa^*
ba^*	b	No	ba^*	-	-	-	-	-	-
sa^*	s	No	sa^*	-	-	-	-	-	-

3.6.2 Geminate correction rules

The term geminate in phonology refers to a long or doubled consonant sound, such as the /*kk*/ in the Hindi word पक्का that contrasts phonemically with its shorter or singleton counterpart पका . Such contrasts occur frequently in Indian languages. There exist other phonetic cues to geminates besides consonantal duration such as pitch and intensity differences. However, this work does not focus on the phonological behavior of geminates. The focus is to keep the geminates together, that is, they are always grouped as a syllable, as the sound is distinct. Samples of geminate correction rules are given below.

Hindi

पक्का (p a)(**k k aa**) - geminate *ka*
पका (p a)(**k aa**) - single *ka*

Tamil

பட்டம் (p a)(**tx tx a m**) - geminate *tx*
படம் (p a)(**tx a m**)- single *tx*

3.6.3 Syllable parsing rules

Though each sound is mapped to a corresponding label in the common label set, the label set does not handle the implicit /*a*/ sound associated with each consonant of the script. Hence, a separate rule is written to add the /*a*/ sound to the labels of all consonants without a vowel modifier associated with it. Thereafter, schwa deletion is performed for Indo-Aryan languages alone. For Dravidian languages schwa deletion rules are not applied. The processed input text is split into a set of sub-syllables, both at vowel and halant[3] positions. These sub-syllables are processed in last to first manner, to ensure

[3]A notation used in most writing systems of the Indian subcontinent to signify the lack of an inherent vowel.

that all the consonantal units are suffixed by a vowel. Necessary correction (if required) is performed subsequently i.e, if the current unit does not possess a vowel sound, it is appended to the previous unit. For example, ताजमहल is syllabified as (t aa j)(m a)(h a l). This rule is significant in particular for chillaksharas in Malayalam that do not possess an inherent vowel. This rule is also considered while grouping geminates as syllables, as the first occurrence of the consonant does not possess an inherent vowel.

3.6.4 Language-specific rules

Not all words will get correctly parsed after applying the above rules. This is due to the fact that each language has certain specific rules which cannot be generalized. These language-specific rules are applied during parsing to improve the accuracy further. A few examples of such rules for Tamil are shown in Figure 3.3.

Grapheme	Phoneme in CLS	Rule	Phoneme in actual pronunciation	Example
க	k	If previous and next grapheme is a vowel	g	ஆகாயம் (aa**g**ayam)
ட	tx	If previous grapheme is ண	dx	வேண்டும் (weenx**dx**um)
ச	c	If previous grapheme is ஞ	j	பஞ்சம் (panj**j**am)
ம	p	If previous grapheme is ம	b	குடும்பம் (kudxum**b**am)
க	k	If previous grapheme is ங	g	திங்கள் (ting**g**alx)

Figure 3.3: Language specific rules for Tamil

3.6.5 Agglutination

Agglutination is the process of combining words that are formed by stringing together morphemes. The stringing of words is carried out without changing the morphemes in either spelling or phonetics. Languages that use the property of agglutination are called agglutinative languages. The unified parser handles agglutinative words that are common in Dravidian languages since it employs a rule-based approach.

Tamil

வந்துக்கொண்டிருக்கிறான் → வந்து கொண்டு இருக்கிறான்

w-a-nd-d-u-k-k-o-nx-dx-i-r-u-k-k-i-rx-aa-n → w-a-nd-d-u k-o-nx-dx-u i-r-u-k-k-i-rx-aa-n

Malayalam

വന്നുകൊണ്ടിരിക്കുന്നു → വന്നു കൊണ്ട് ഇരിക്കുന്നു

w-a-n-n-u-k-o-nx-tx-i-r-i-k-k-u-n-n-u → w-a-n-n-u k-o-nx-tx i-r-i-k-k-u-n-n-u

3.7 Experiments and results

3.7.1 Dataset

The dataset used[4] is released as part of resources for Indian languages [Baby et al., 2016b]. Text to speech synthesis systems are built using the language specific parsers and unified parser for 11 Indian languages detailed in Table 3.4.

Table 3.4: Dataset

Language	**Type**	**Duration (in hrs)**
Bengali	Male	6.05
Bodo	Female	4.00
Gujarati	Male	4.92
Hindi	Male	5.03
Kannada	Male	3.01
Malayalam	Male	5.70
Manipuri	Male	6.61
Marathi	Female	4.80
Rajasthani	Male	5.82
Tamil	Male	4.30
Telugu	Male	4.20

[4]Available at https://www.iitm.ac.in/donlab/tts/database.php

3.7.2 Pairwise comparison test

Pairwise Comparison (PC) tests are performed by an average of 12 native listeners to evaluate the performance of the unified parser approach. The listener listen a set of 15 sentence pairs. Each pair is synthesised using TTS systems created using the unified parser and native parser. Listeners can give preference to either one of these systems or as equal (if both sounds similar). PC tests reveal the effectiveness of the unified parser. As can be seen from Figure 3.4, in most cases the unified parser and native parser have the same preference. Occasionally there is a preference for the unified parser.

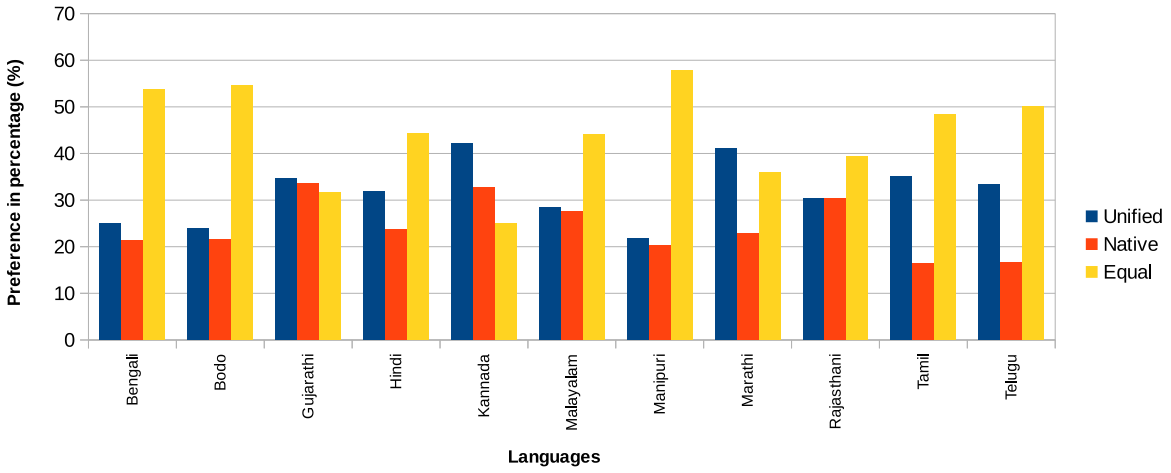


Figure 3.4: Pairwise Comparison test results

Experimental results show that the unified parser is more robust and accurate compared to current systems that require similar supervision. This work is also the first attempt to introduce a unified approach, using the common label set, for parsing Indian languages. Unlike previous systems that require manually-constructed rules, this system requires much less knowledge of the native languages and can be easily scaled to other languages. To build TTS systems for a new language, the mapping from grapheme to CLS needs to be identified. Existing language-specific rules can be adapted. For example, Hindi and Rajasthani follow mostly the same set of rules.

3.8 Multilingual synthesis using unified parser

The unified parser parses Indian language text to CLS. This can be exploited in building multilingual TTS systems. Also, data from different languages can be pooled to build much robust speech systems. This is more relevant for Indian languages since code-switching and code-mixing are predominant across Indian languages.

3.9 Summary

The unified parser is a step towards building an efficient pronunciation generator for Indian languages. Although the objective was primarily to unify various Indian language parsers, it is observed that the unified parser is more robust than custom built parsers. In order to scale unified parser to a new language, one needs to identify the language family, borrow the standard rule set. Exceptions may be handled by incorporating language-specific rules.

CHAPTER 4

Segmentation of Speech Signals

4.1 Introduction

Segmentation is the process of finding the time aligned sequence of phones/syllables in the speech utterance. Figure 4.1 shows the segmentation of the Hindi utterance उस रामा into phones and syllable units. The syllable boundaries are marked in red and phone boundaries are marked in blue.

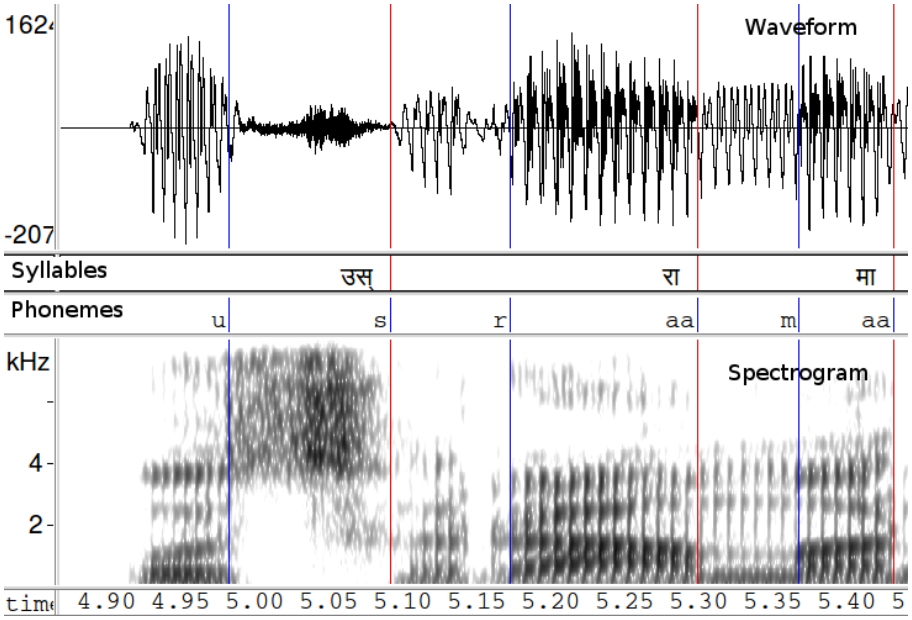


Figure 4.1: Sample segmentation

Segmentation of speech into accurate time-aligned phonetic transcriptions plays a vital role in building robust speech systems, including statistical parametric speech synthesis (SPSS) systems, as the duration of HMM states is explicitly modeled and generated during synthesis [Zen et al., 2009]. The widely used HMM-based Viterbi forced alignment is not ideal for speech synthesis [Sethy, 2002, Black and Kominek, 2009] as the location

of the phoneme boundary is not used as a criterion for estimation of parameters. It often requires manual correction after the forced alignment. Manual labeling for a huge multi-lingual corpus is time-consuming and error-prone which warrants automatic procedures that are better than Viterbi force-aligned HMM segmentation. There have been many attempts at improving the accuracy of HMM segmentation. In [Kim and Conkie, 2002], a spectral transition measure is used to correct boundaries having abrupt spectral changes. In [Black and Kominek, 2009], the boundaries are iteratively moved forward or backward by one frame, depending upon the direction in which frame classification accuracy increases.

Wherever hand-labeled[1] data is available, for example, the TIMIT corpus [Zue et al., 1990], machine learning models have been trained to learn the boundaries [Hinton et al., 2012]. For example, [Lo and Wang, 2007] use support vector machine (SVM) and [Lee, 2006] uses a multi-layer perceptron to refine the HMM boundaries. The best-reported results on TIMIT database use a fusion of multiple acoustic front-ends (i.e. systems based on mel-frequency cepstral coefficients (MFCC), perceptual linear predictive (PLP), etc.), on top of boundary correction models such as neural networks and single-state HMMs, thereby improving the segmentation accuracy to 96.7% within a tolerance of 20 ms [Stolcke et al., 2014]. However such hand-labeled data is not available for Indian languages. Further, even in such a carefully designed and labeled corpus vowel deletion has been observed [Golda Brunet and Murthy, 2017].

Accurate phonetic segmentation becomes a problem when only the phoneme sequences are available and not their boundary locations. Signal processing cues that are agnostic to the speaker can be used to get syllable boundaries [Prasad et al., 2004]. Signal processing cues result in false alarms but seldom introduce deletions when the parameters are chosen such that the boundaries are overestimated. Phonetic transcription can be used in tandem with signal processing cues to eliminate insertions. Signal processing cues along with HMM-based alignment has been used for segmenting speech data in TTS systems for Indian languages [Shanmugam and Murthy, 2014b].

[1]Manual labeling performed by experts

4.2 Role of segmentation

Text pre-processing, text parsing, speech segmentation, training and synthesizing are the different phases of developing a TTS system. In text parsing phase, the parser converts text sentences into a sequence of sub-word units, syllables or phones. During segmentation, the boundaries of these sub-word units are obtained from the speech utterance. TTS systems are trained thereafter using these obtained boundaries. In USS, during training, the boundary information is used to organise the speech database into a suitable structure for easy retrieval of sub-word units that match a particular context. Classification and regression tree (CART) is the most commonly used structure [Riley, 1991]. During synthesis, appropriate sub-word units, with the aid of context information, are concatenated to get the synthesized speech utterance. In HTS, during the training phase, the boundary information is used for modeling sub-word units. During the synthesis phase, the speech is synthesized for a given text using the trained sub-word models. The goodness of a TTS system is evaluated based on the naturalness and intelligibility of the synthesized voice, which in turn is related to the trained sub-word models (in HTS) and the CART (in USS). Hence, in both systems, the quality of synthesis depends on the boundaries of sub-word units. Robust sub-word unit models can be built during the training phase, if a large amount of training data is available[2] with machine learning techniques.Being statistical modeling techniques, the boundaries are accurate when a large amount of data is available. In the absence of hand-labeled data, segmentation boundaries of the sub-word units become crucial in determining the quality of synthesized speech. Since Indian languages are digitally low resource languages, accurate segmentation of speech data into sub-word units becomes an important sub-task in building TTS systems.

The success of a good TTS system depends on the segmentation of the speech data which in turn depends on the availability of: a) a good phone recognizer b) manually labeled corpora c) a large amount of accurate parallel text and speech corpora. For high resource languages like English, although phoneme level transcriptions are not available, accurate sentence level transcriptions are available. In addition to this, a huge speech

[2]The training data of a TTS system consist of text sentences and the corresponding recorded speech utterances.

corpus is also available [Godfrey et al., 1992]. Machine learning algorithms like support vector machines (SVMs) [Lo and Wang, 2007] and multi-layer perceptrons [Lee, 2006] are trained on this corpus to learn phone boundaries. Using boundary information from any one of these approaches, ASR systems are built for English. These automatic speech recognition (ASR) systems are used for phone segmentation in English TTS systems [Schwarz et al., 2006]. In [Kominek and Black, 2004], Kominek and Black introduced a new speech database for English (CMU ARCTIC database) that better suits the requirement of speech synthesis. This database consists of around two hours of speech data. The phone labels are obtained using CMU-Sphinx train tool [Huang et al., 1993]. These labels are then verified and corrected manually, and USS voices are built [Kominek et al., 2003]. Since the database is manually annotated, the quality of synthesized voice is good. Apart from using phone recognizers or manually labeled corpora, various semi-automatic, and automatic segmentation algorithms are available for different languages [Lee et al., 1990, Wilpon et al., 1990, Brognaux and Drugman, 2016, Young et al., 2002, Prasad et al., 2004, Shanmugam, 2015]. Most of these perform HMM-based forced Viterbi alignments, which is followed by either manual correction of phone boundaries or the use of other techniques for correcting the HMM boundaries.

4.3 Segmentation for Indian languages

Speech segmentation framework in Indian language TTS systems include both semi-automatic and automatic segmentation approaches. In semi-automatic methods, segmentation is performed using some automated approach, but with the aid of manual intervention. This automated method could be machine learning algorithms or signal processing based techniques. Gaussian mixture model-HMM-based *bootstrap* segmentation (GMM-HMM-BS) [Lee et al., 1990, Wilpon et al., 1990, Young et al., 2002], *group delay* based segmentation [Prasad et al., 2004] are two such approaches. The boundaries given by these methods are not accurate and hence requires manual intervention to make the boundaries accurate. Automatic segmentation of speech utterances is mostly based on GMM-HMM *flat start* (GMM-HMM-FS) segmentation. In this approach, segmentation

is performed automatically without any manual intervention. However, the boundaries given by this method is not very accurate. Another automatic approach, hybrid segmentation, uses signal processing techniques in tandem with GMM-HMM-based flat start segmentation [Shanmugam, 2015]. Various automatic and semi-automatic approaches used for speech segmentation are discussed in the following subsections.

4.3.1 GMM-HMM flat start approach (GMM-HMM-FS)

GMM-HMM is one of the widely accepted approaches for phone segmentation in Indian languages. In this method, the speech utterance is initially divided into segments of equal length, where the number of segments is exactly same as the number of phones that make up the utterance. This initial alignment is used for training GMM-HMM monophone models. Separate models are trained for each phone in the phoneset. The parameters, mean and variance, of every state in all monophone HMMs are initialized with global mean and variance. Baum-Welch embedded re-estimation is then performed to update the parameters of each HMM. This process is repeated iteratively, and the optimal phone boundaries are obtained by performing forced Viterbi alignment using the updated parameters of monophone HMMs. This method gives accurate phone/syllable boundaries if data is available in abundance. Otherwise, manually segmented data is required to get accurate boundaries as discussed in [Lee, 2006, Kim and Conkie, 2002, Ogbureke and Carson Berndsen, 2009, Yuan et al., 2013]. But for low resource languages like Indian languages manually labeled data are not available and hence flat start boundaries are approximate and non-accurate [Shanmugam, 2015].

4.3.2 GMM-HMM bootstrap approach (GMM-HMM-BS)

GMM-HMM-based speech segmentation gives better phone boundaries with better initial alignments. GMM-HMM-FS segmentation does not provide robust phone models because the initial alignments are not good. Hence, another approach, called GMM-HMM bootstrap segmentation (GMM-HMM-BS) is adopted for building better phone models. In this approach, to get better initial phone alignments, a small amount of speech data (for

example, 5 minutes) is selected and labeled manually. This data is used for building the initial monophone HMM models. The data is selected in such a way that all the phones of a given language are covered. Baum-Welch embedded re-estimation is then performed on the rest of the data to refine the phone models. The phone models are refined iteratively until the segmentation becomes satisfactory. Finally forced Viterbi alignment is used to get the phone boundaries using the refined phone models. Hence the obtained phone boundaries are better than that of GMM-HMM-FS segmentation, but yet, they are not very accurate. These boundaries obtained are adequate for ASR systems, but they are not good enough to build high-quality TTS systems. The boundaries obtained with bootstrap segmentation is shown in Figure 4.2. It is seen from the figure that the bootstrap (BS) boundaries are better than flat start (FS) boundaries, nevertheless, they are not yet accurate.

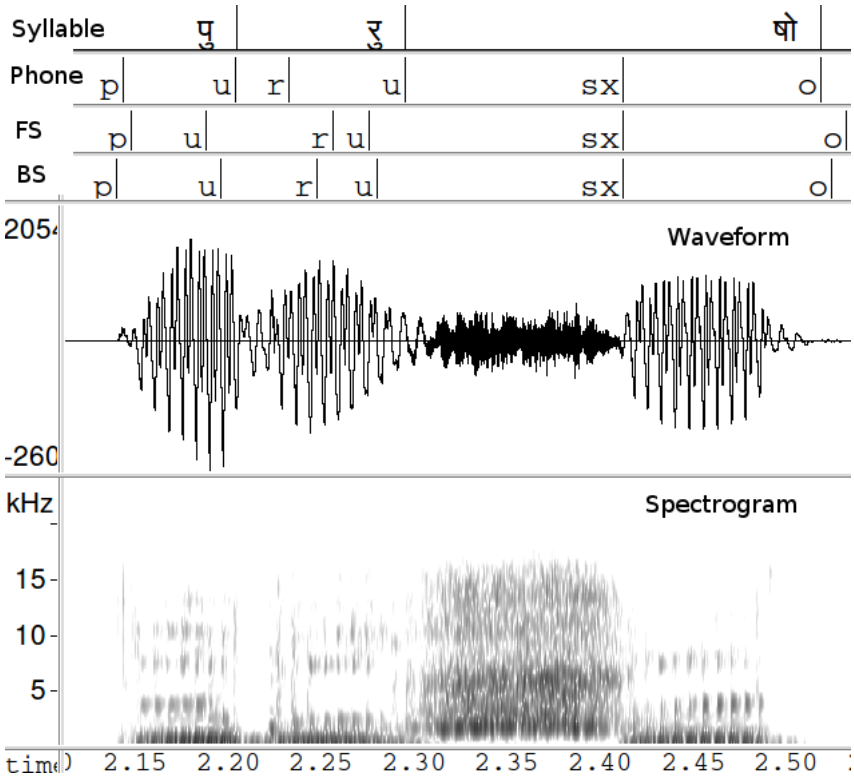


Figure 4.2: Bootstrap segmentation. The first panel shows the syllable transcription (in UTF-8 encoding), the second panel shows the actual phone boundaries, the third panel shows the phone boundaries obtained with flat start segmentation (5 hours of data) and the fourth panel shows the phone boundaries obtained with bootstrap segmentation (5 hours of data, out of which 10 minutes was manually labelled).

4.3.3 Group delay based semi-automatic approach (GDS)

In this method, speech segmentation is performed at syllable level. The syllable boundaries are identified with the aid of signal processing cues, followed by a manual correction. This approach is widely used in syllable based TTS systems [Pradhan et al., 2013], and phone-based TTS systems which perform embedded re-estimation on corrected syllable boundaries to get phone models [Shanmugam and Murthy, 2014a].

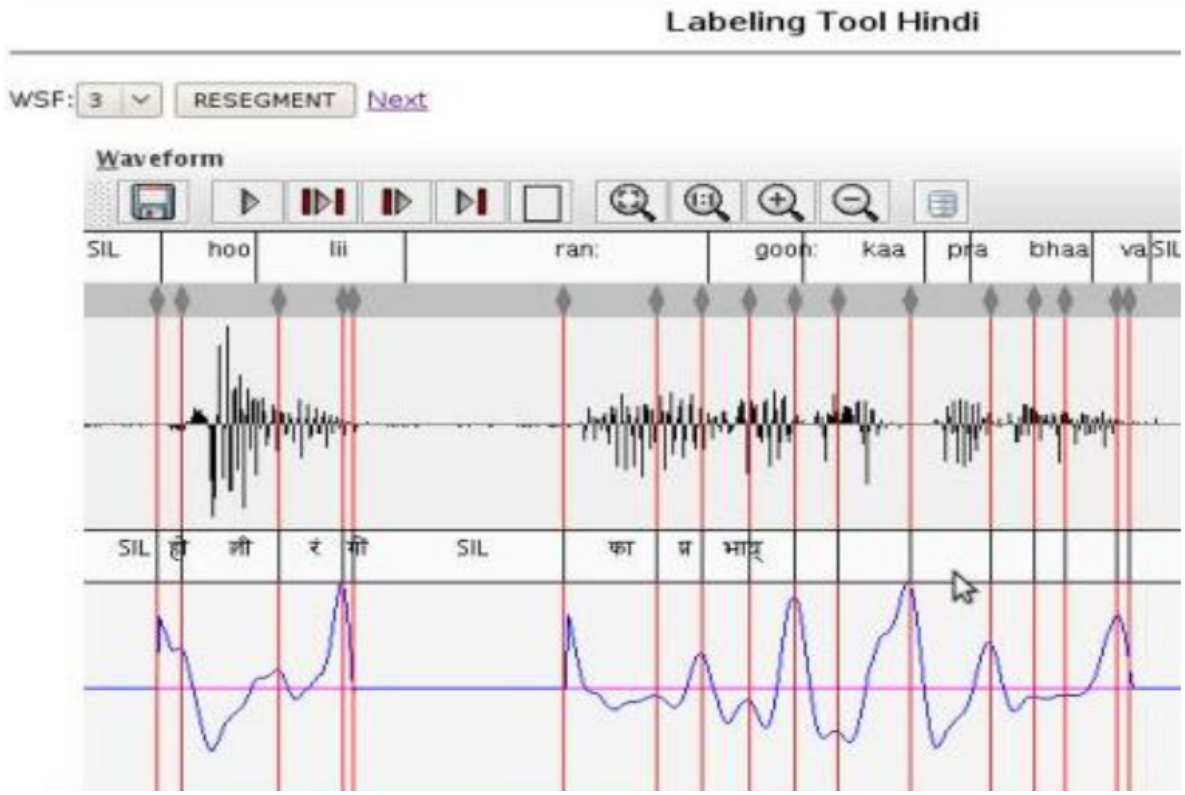


Figure 4.3: Semi-automatic labeling tool for manual correction of syllable boundaries obtained with GD segmentation. The tool has an option to play the waveform fully or as segments, and to insert, delete, or move the syllable boundaries. It also provides the facility to play with a tuning parameter called window scale factor (WSF) (discussed in Section 4.4) to control the number of syllable boundaries given by GD.

Group delay (GD) of short-term energy (STE) is used widely to identify the syllable boundaries from speech signals without any phonetic transcriptions [Prasad et al., 2004]. This process is detailed in Section 4.4. After getting the syllable boundaries, the boundary sequence is mapped directly to syllable sequence from the text. This leads to two types of errors namely, (a) insertion error and (b) deletion error. *Insertion error* occurs when an

additional boundary is obtained within a syllable and *deletion error* occurs when a syllable boundary is missed out. Since the boundary sequence is mapped directly to syllable sequence, the presence of insertion or deletion errors leads to incorrect boundaries of other syllables also. Hence, the boundaries given by this method is often corrected manually to get accurate boundaries. A semi-automatic labeling tool is developed [Deivapalan et al., 2008], to correct the boundaries obtained from GD segmentation as shown in Figure 4.3.

4.3.4 Automatic hybrid segmentation (GMM-HMM-BC)

In this method, signal processing cues are used in tandem with HMM-based forced Viterbi alignment to get accurate syllable/phone boundaries. HMMs take text transcription as input, and hence they do not have any insertion/deletion errors. However, phone boundaries given by this method is not accurate [Sethy, 2002], compared to the GD syllable boundaries. Hence the syllable boundaries obtained from HMM-FS are corrected to the closest GD syllable boundaries iteratively. After obtaining accurate boundaries, Baum-Welch embedded re-estimation is performed within a syllable to get monophone models. Using these models, forced Viterbi alignment is then performed. This is the state-of-the-art segmentation approach for Indian language TTS systems [Shanmugam and Murthy, 2014a].

Hybrid segmentation depends on acoustic cues for improving the speech segmentation. It uses GD of short-term energy and sub-band spectral flux for the performance enhancement. A number of syllable boundaries are corrected based on the rules developed from empirical analysis of the acoustic cues. This is detailed in Section 4.4.

4.4 Importance of acoustic cues

Phones are the most common subword unit for speech modeling. But in most of the cases, only sentence level transcription is available for training the models [Godfrey et al., 1992, Baby et al., 2016b,Hirsch and Pearce, 2000]. Obtaining accurate phone level alignment is a difficult task. Manual alignment is not only time-consuming but also inconsistent as it

is difficult to perceive a phone in isolation. A common set of fundamental units that can be defined "universally" across all spoken languages [Siniscalchi et al., 2013]. For Indian languages syllables are found to be robust units [Patil et al., 2013, Lakshmi and Murthy, 2006].

4.4.1 Syllable as an alternative to phone

Syllable, the fundamental unit of speech production can be used as an alternative to the phone. Syllables have typical spectral and temporal characteristics, are much longer in duration (about 150ms) and can be detected using signal processing cues. Syllables are also closely related to human speech perception and articulation [Ganapathiraju et al., 2001]. Analysis of pronunciation variation at syllable level is observed to be more systematic [Greenberg, 1999]. Syllable is found to be a robust subword unit for Indian languages [Patil et al., 2013, Pradhan et al., 2015, Lakshmi and Murthy, 2006]. Syllable modeling results in the reduction of model parameters as context dependencies are less important for syllable models compared to that of tri-phone models [Tachbelie et al., 2012, Tachbelie et al., 2014].

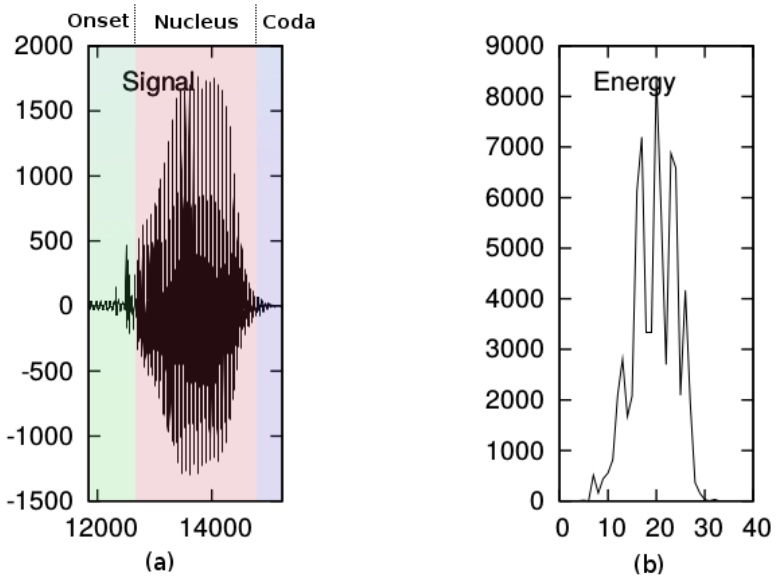


Figure 4.4: Waveform and energy plots of a syllable. X-axis correspond to time in samples at a sampling rate of 16 kHz (fig a). The energy plot shows the corresponding number of frames in a syllable (fig b).

A syllable consists of one or more phones and is of the form C*VC*, where C is a consonant and V is a vowel. A syllable consists of three parts - onset, nucleus, and coda, as shown in Figure 4.4(a). The onset and coda consist of consonants whereas nucleus is a vowel. Syllable boundary detection from the acoustic waveform is comparatively easy. Group delay (GD) based techniques are proven to give robust syllable boundaries for Indian languages [Murthy and Yegnanarayana, 1991, Murthy and Yegnanarayana, 2011, Shanmugam and Murthy, 2014b, Nagarajan and Murthy, 2004, Prasad et al., 2004].

Signal processing cues are agnostic to speakers and languages. Most widely used signal processing cue for obtaining syllable boundaries in ASR and TTS systems is short-term energy (STE) [Murthy and Yegnanarayana, 2011, Prasad et al., 2004]. Sub-band spectral flux (SBSF) is used for detecting fricative, affricate, and nasal boundaries [Shanmugam, 2015].

4.4.2 Short-term energy (STE) for syllable boundary detection

Owing to co-articulation in continuous speech, it is more difficult to distinguish phone transitions than syllable transitions [Shanmugam, 2015]. It is easier to obtain syllable boundaries than phone boundaries. The region of vowels in syllables corresponds to more energy and duration than that of consonants. Boundaries of syllables correspond to low energy region. Short term energy (STE) can be used as an acoustic cue to obtain syllable boundaries. STE function $E[m]$ where $m = 1, ..., M$ is calculated from the given speech utterance $x[n]$ as:

$$E[m] = \sum_{m=1}^{M} (x[n].w[m-n])^2 \tag{4.1}$$

where $w(n)$ presents the windowing function of finite duration and m represents the frame shift

But STE cannot be used directly due to local fluctuations in energy (Figure 4.4(b)). But a smoothed version of STE can be used to detect syllable. Figure 4.5 shows a speech waveform along with the smoothed version of the STE. Since the fluctuations in STE are smoothed with GD processing, each of the valleys resembles syllable boundaries. This

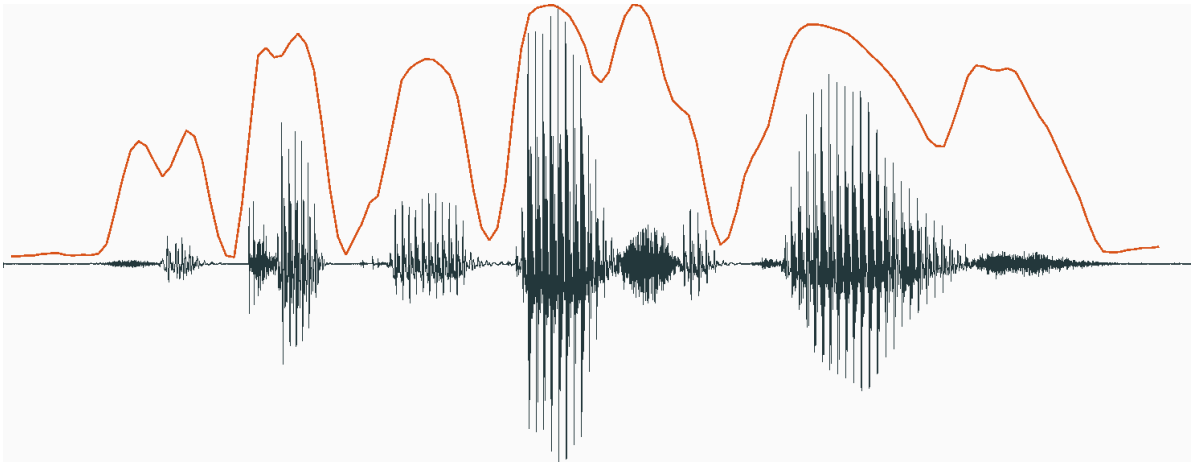


Figure 4.5: Speech waveform with smoothed STE

can be used as a cue to detect many syllable boundaries accurately.

STE can be made to resemble magnitude spectrum of any real signal, and GD based processing can be applied to obtain the syllable boundaries. GD function can be applied to minimum phase signals only. Hence, the signal is made minimum phase by processing in the root cepstral domain (inverse DFT of the short-term energy function) [Lim, 1979, Nagarajan et al., 2003] and then GD function is applied on this minimum phase signal. In [Prasad et al., 2004], the use of minimum phase group delay functions in finding syllable boundaries for speech recognition is proposed.

This GD based algorithm is agnostic to text transcription as the boundaries are obtained directly from the waveform independent of the transcription. The number of syllable boundaries given by the algorithm depends on the size of the Hanning window chosen in the cepstral domain [Prasad et al., 2004], which in turn depends on a parameter called window scale factor (WSF). WSF is inversely proportional to the syllable rate of the utterance. Figure 4.6 shows the GD of the STE of a part of a Tamil utterance for various WSF values- $10, 3.4$, and 1 - in the three panes below the waveform. The syllable boundaries are marked by orange line. From the figure, it can be observed that WSF of 3.4 gives good syllable boundaries.

Examples of syllable boundaries obtained with GD of STE are shown in Figures 4.7, 4.8, 4.9, and 4.10 for the languages Hindi, Bengali, Telugu, and Tamil respectively. GD of STE gives peaks at the location of syllable boundaries, irrespective of language. However,

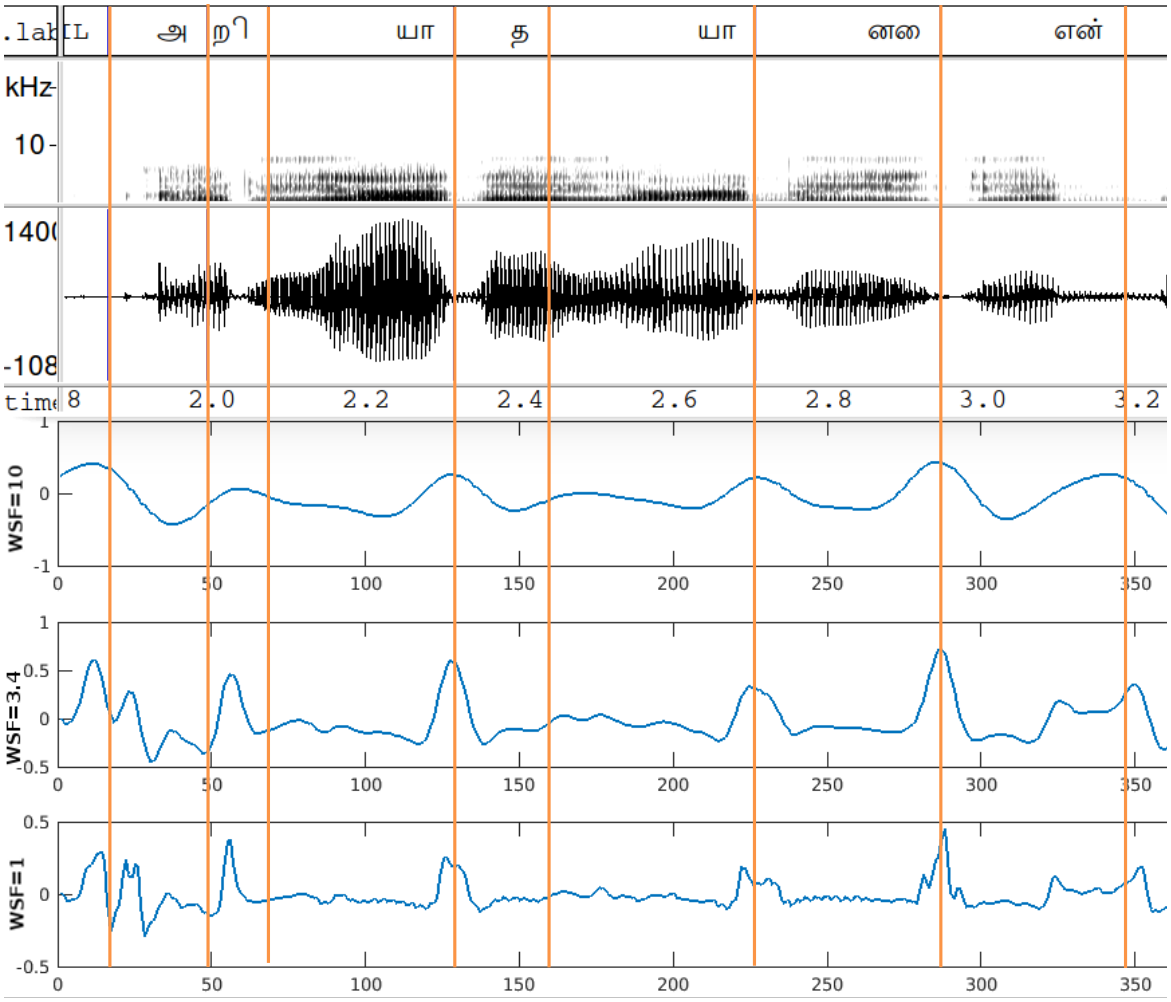


Figure 4.6: GD boundaries for different WSF values

GD of STE fails to identify syllable boundaries in a few cases. This is discussed in detail in Section 4.4.4.

4.4.3 Sub-band spectral flux (SBSF) for phone transitions

Based on extensive experimentation, it is observed that sibilant fricatives and affricates have prominent energy in higher frequency bands, and nasals have prominent energy only in lower frequency bands. Spectral change as a function of time can be used to detect phone boundaries when the phone transition is accompanied by significant change in spectral characteristics [Kim and Conkie, 2002, Rabiner and Juang, 1993]. Spectral flux (SF), which is the Euclidean distance between the normalized power spectrum of a speech frame and normalized power spectrum of the previous frame, gives a measure of

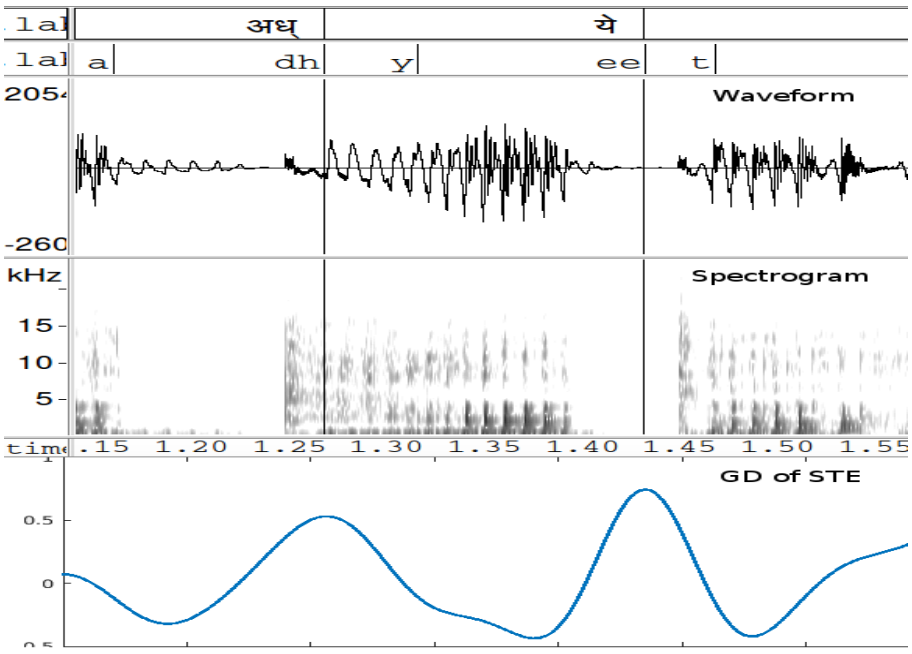


Figure 4.7: An example of a Hindi utterance that shows syllable boundary detection with GD of STE. The first and second panels show the syllable and phone transcriptions respectively. The boundary of the syllables *adh*, and *yee* corresponds to a peak in the GD of STE.

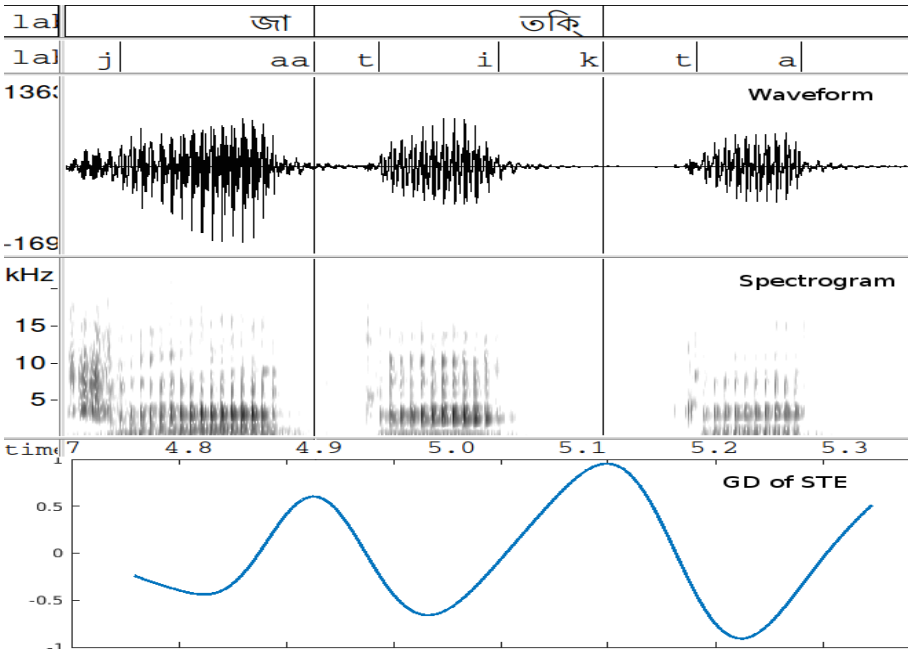


Figure 4.8: An example of a Bengali utterance that shows syllable boundary detection with GD of STE. The first and second panels show the syllable and phone transcriptions respectively. The boundaries of the syllables *u* and *jaa*, and *jaa* and *tik* corresponds to a peak in the GD of STE.

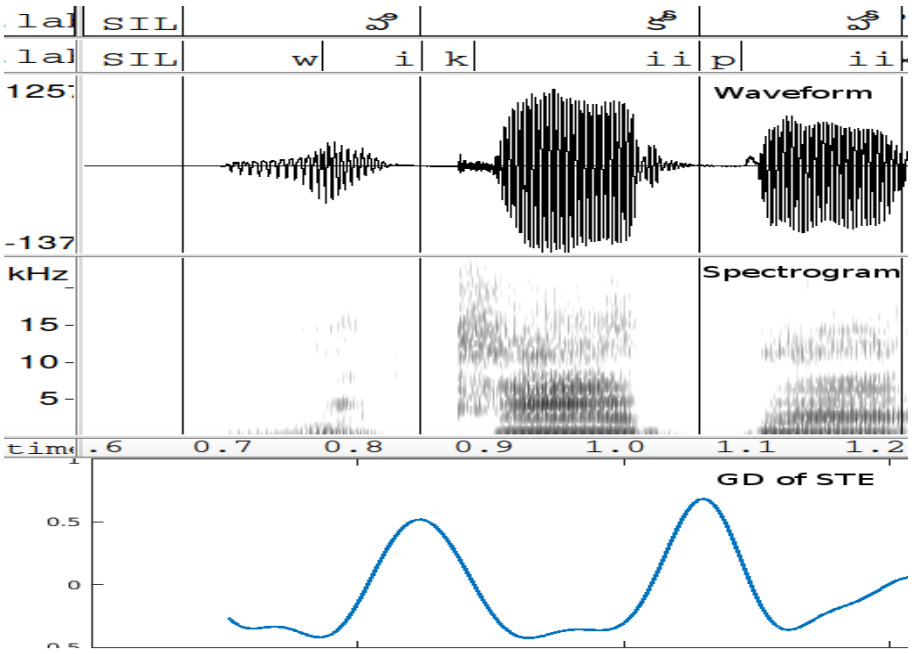


Figure 4.9: An example of a Telugu utterance that shows syllable boundary detection with GD of STE. The first and second panels show the syllable and phone transcriptions respectively. The boundary of the syllables *wi*, and *kii* corresponds to a peak in the GD of STE.

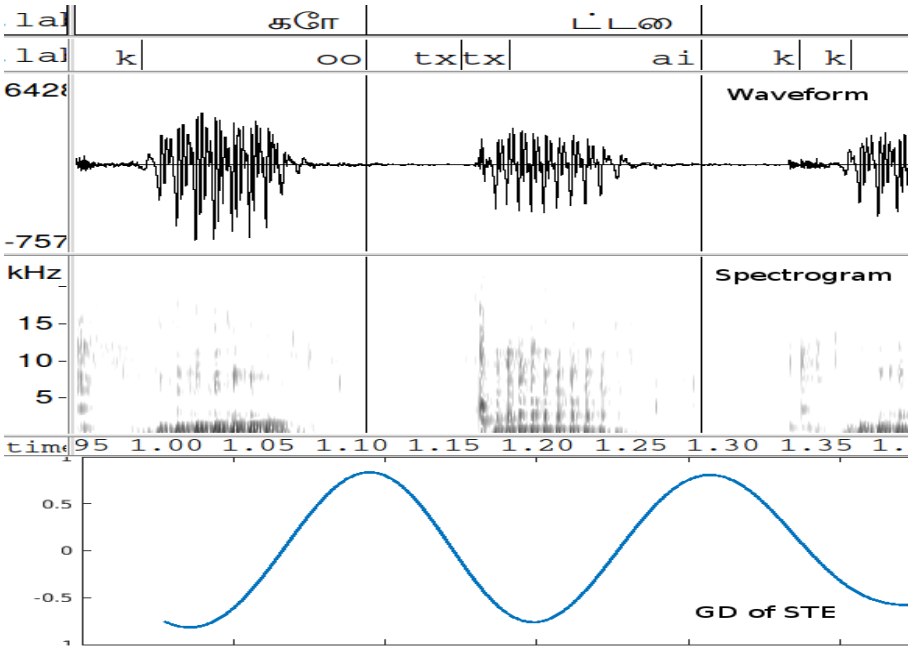


Figure 4.10: An example of a Tamil utterance that shows syllable boundary detection with GD of STE. The first and second panels show the syllable and phone transcriptions respectively. The boundary of the syllables *koo*, and *txtxai* corresponds to a peak in the GD of STE.

spectral change. Spectral flux for n^{th} frame (SF_n) and $n-1^{th}$ frame, where E_n is the energy of n^{th} frame, is given by the equation:

$$SF_n = (E_n - E_{n-1})^2 \tag{4.2}$$

The peaks in the spectral flux correspond to phone boundaries. This property of spectral flux can be used for obtaining the phone boundaries of sibilant fricatives, and affricates [Shanmugam, 2015]. Phone boundaries are characterized by energy changes in different bands of the spectrum [Kim and Conkie, 2002]. Sub-band spectral flux (SBSF) is computed by dividing the normalized power spectrum into four bands uniformly, and finding the squared difference between the band energy of a frame with that of the previous frame as given by the equation:

$$SBSF_n = \sum_{i=1}^{4}(E_n[i] - E_{n-1}[i])^2 \tag{4.3}$$

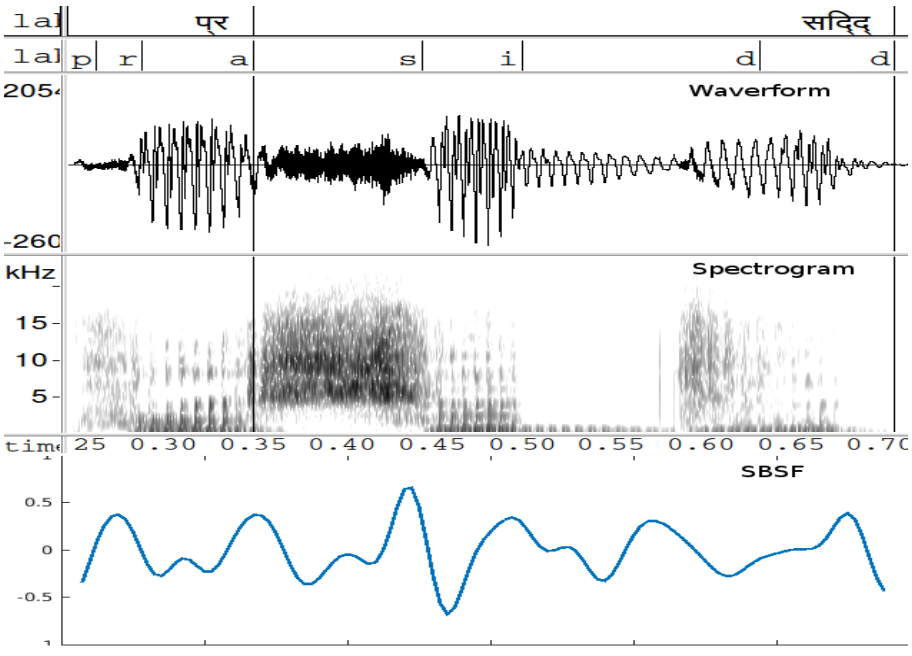


Figure 4.11: An example of a Hindi utterance that shows syllable boundary detection with SBSF. The first and second panels show the syllable and phone transcriptions respectively. SBSF gives a peak near the boundary of the syllables *pra*, and *sid* (first phone of the syllable *sid* is a fricative *s*).

SBSF gives peak at locations where there is a significant change in spectral character-

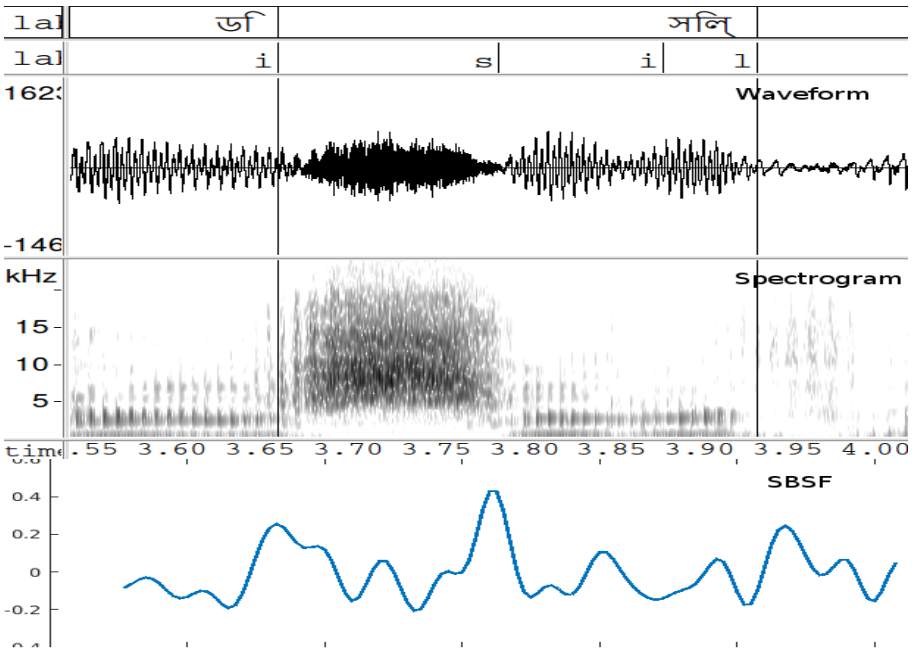


Figure 4.12: An example of a Bengali utterance that shows syllable boundary detection with SBSF. The first and second panels show the syllable and phone transcriptions respectively. SBSF gives a peak near the boundary of the syllables *dxi*, and *sil* (first phone of the syllable *sil* is a fricative *s*).

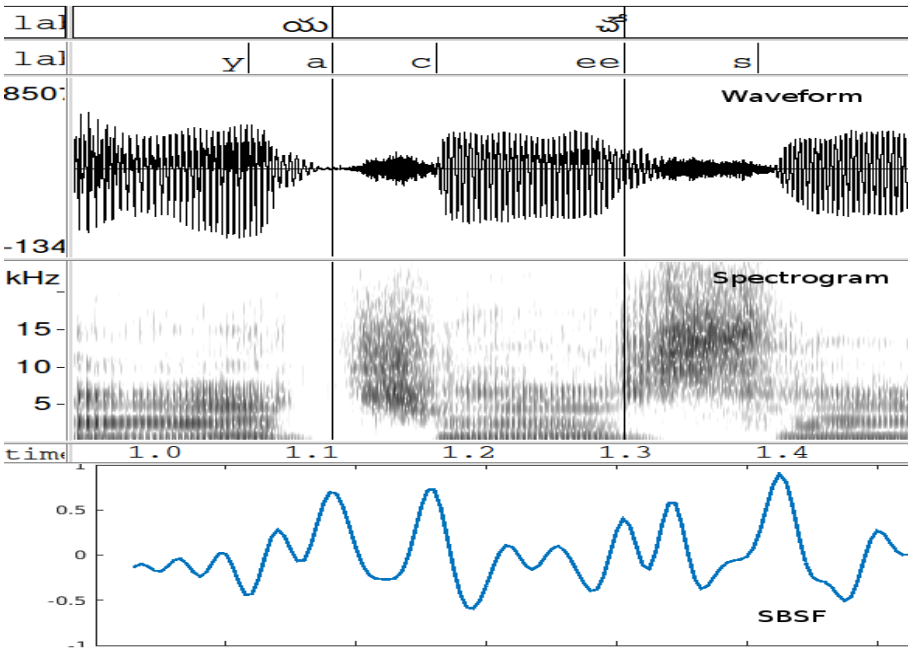


Figure 4.13: An example of a Telugu utterance that shows syllable boundary detection with SBSF. The first and second panels show the syllable and phone transcriptions respectively. SBSF gives a peak near the boundary of the syllables *ya*, and *cee* (first phone of the syllable *cee* is an affricate *c*).

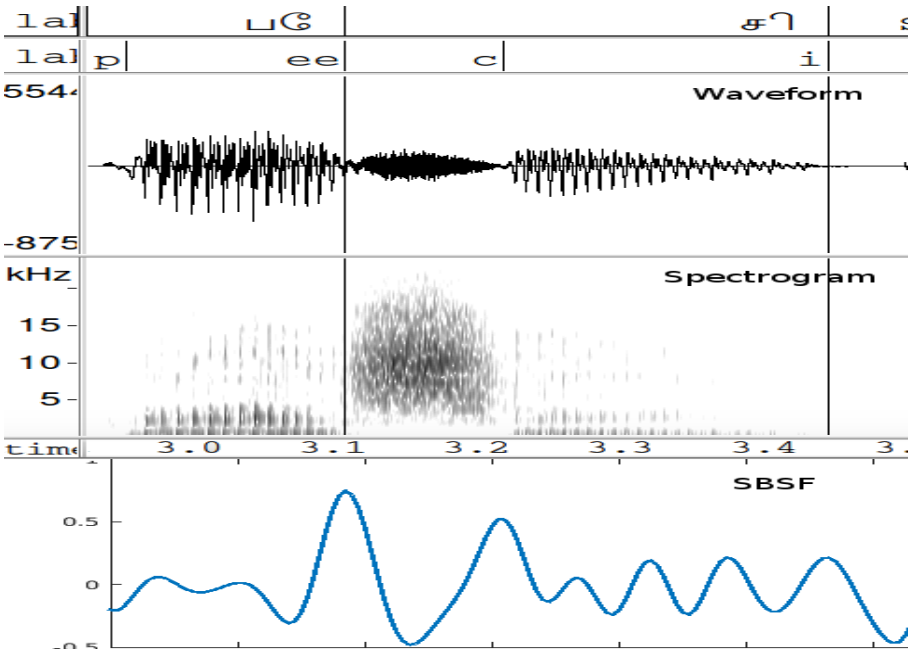


Figure 4.14: An example of a Tamil utterance that shows syllable boundary detection with SBSF.The first and second panels show the syllable and phone transcriptions respectively. SBSF gives a peak near the boundary of the syllables *pee*, and *ci* (first phone of the syllable *ci* is an affricate *c*).

istics. However, it is observed that it gives peaks consistently for sibilant fricatives and affricates. Hence, SBSF can be used for finding the boundaries of syllables which start or end with sibilant fricatives or affricates. The boundary detection algorithm based on SBSF function is detailed in [Shanmugam, 2015]. Figures 4.11, 4.12, 4.13, and 4.14 shows examples of syllable boundaries obtained with SBSF for the languages Hindi, Bengali, Telugu, and Tamil respectively.

4.4.4 Rules for boundary correction

With extensive experimentation using GD segmentation algorithm, it is observed that the boundaries obtained from GD processing of STE and SBSF do not yield accurate boundaries always [Shanmugam, 2015]. The boundaries given by GD of STE will not be accurate if (a) a syllable has a semi-vowel or an affricate at its beginning and (b) a syllable has a fricative or nasal at its beginning or end. An example for this is illustrated in Figure 4.15. In Figure 4.15, the GD of STE does not show any peak at the boundary between the syllables तम्(*tam*) and अग्(*ag*). The presence of the nasal *m* in the syllable

tam caused a dip in GD function. SBSF fails to find phone transitions if both the phones are fricatives or affricates. An example for this is illustrated in Figure 4.16. The boundary between the syllables नॉस्(*naxs*) and फ़ी(*fii*) is not detected using SBSF since both the last phone of *naxs* (fricative *s*) and the first phone of *fii* (fricative *f*) have prominent energy in higher frequency bands, and SBSF cannot capture changes in spectral characteristics.

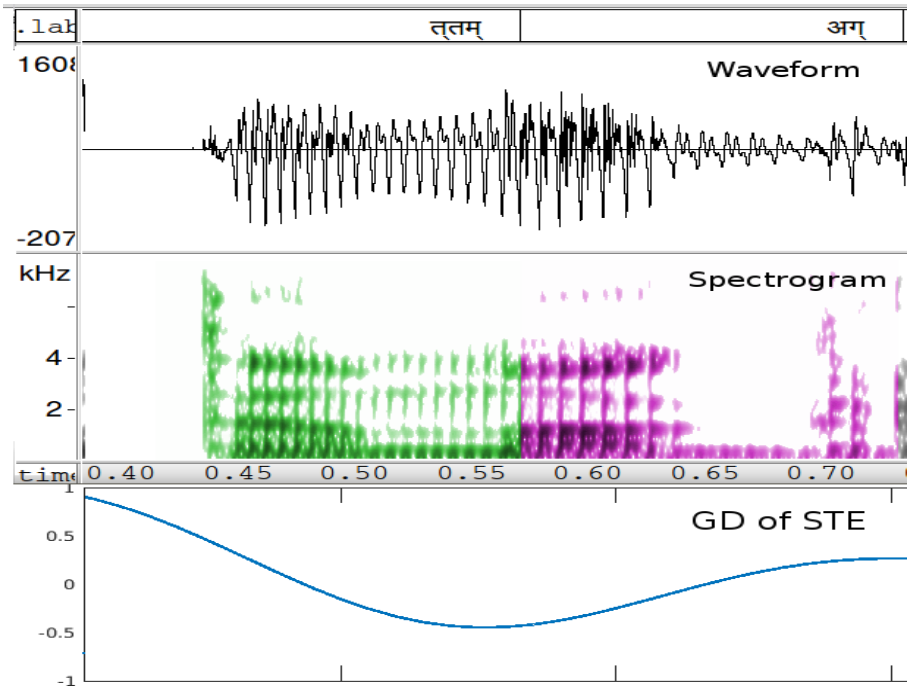


Figure 4.15: An example of a Hindi utterance that shows a case where GD of STE will not give good syllable boundary. STE will not capture the boundary between syllables *tam* and *ag*.

Hence, even though STE and SBSF give syllable boundaries, they do not yield accurate boundaries always. Therefore, in this work, two boundary marking rules are used to identify the correct boundaries given by spectral cues. The following rules are applied to determine if the boundary between two syllables syl_1 and syl_2 are correct [Shanmugam, 2015]:

- **Rule 1:** The boundary between the syllables syl_1 and syl_2 is marked as correct using STE, if the end-phone (ɛ́) of syl_1 is not a fricative or nasal, and the beginning-phone (ɓ) of syl_2 is not a fricative, affricate, nasal or a semi-vowel.
- **Rule 2:** The boundary between syl_1 and syl_2 is marked as correct using SBSF, if the ɛ́ of syl_1 or the ɓ of syl_2, but not both, is a fricative or an affricate.

A sample segmented speech utterance with syllable boundaries obtained with GD of

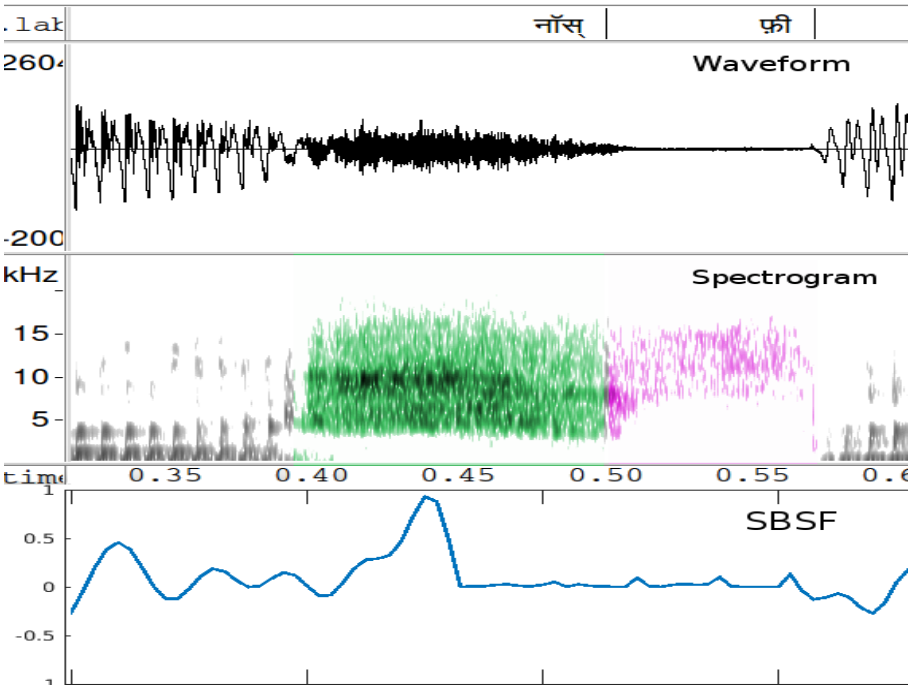


Figure 4.16: An example of a Hindi utterance that shows a case where SBSF fails to give good syllable boundary. SBSF will not capture the boundary between syllables *naas* and *free*.

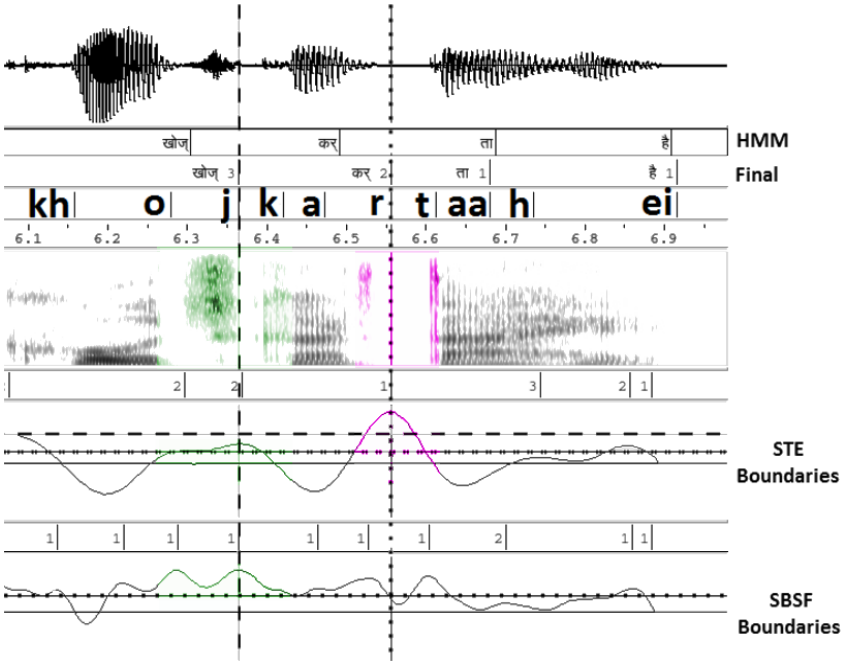


Figure 4.17: An example of a Hindi utterance that shows boundary correction using STE and SBSF.

STE and SBSF based on these boundary correction rules are shown for part of a Hindi utterance in Figure 4.17. The syllable boundaries obtained with GD of STE is denoted

with the number *2* and those obtained with SBSF is denoted with the number *3* beside the syllable transcription (in Figure 4.17). The number *1* denotes that the boundaries are not corrected. In the figure, the boundary between syllables खोज्(*khoj*) and कर्(*kar*) is corrected using SBSF (*Rule 2*), and the boundary between syllables कर्(*kar*) and ता(*taa*) is corrected using STE (*Rule 1*).

4.5 Summary

This chapter briefly reviews the speech segmentation processes. The importance of accurate speech segmentation for building TTS systems are discussed. Various approaches used for segmentation for Indian languages are also detailed. The importance of spectral cues for speech segmentation is discussed. The next chapter presents details about neural network based techniques and the proposed approach of using deep learning techniques in tandem with signal processing cues.

CHAPTER 5

Deep Neural Networks in Tandem with Spectral Cues for Speech Segmentation

5.1 Introduction

In India, spoken languages belong to several language families, the major ones being Indo-Aryan and Dravidian. Nevertheless, the amount of available digital resources in terms of parallel speech and text corpora is very small [Post et al., 2012, Joy et al., 2014]. There is no single language spoken in the entire country. Hence, in the context of speech synthesis, separate TTS systems are needed for these languages.

However, building TTS systems require huge amount of data for training robust machine learning models. Collecting huge amount of data for such large number of languages is a tedious and expensive task. Unlike the English language, manually labeled speech corpora, or good generic ASR systems are not available for phone segmentation of Indian languages. The use of language-independent phone recognizers also failed for these languages. The phone boundaries given by the state-of-the-art language-independent phone recognizer by the Brno University of Technology (BUT), *the BUT phone recognizer* [Schwarz et al., 2006], for a Hindi speech utterance is shown in Figure 5.1. Comparison between first panel (phone boundaries given by the BUT recognizer) and second panel (actual phone boundaries) shows the presence of insertion error (phone *m* between the phones *i* and *d*) and substitution errors (substitution of phone *a* with phone *i*, phone *d* with phone *u*, and phone *k* with phone *g*) [1]. Hence, as discussed in Section 4.2, better phone boundaries become mandatory to develop robust TTS systems.

[1]Insertion error occurs when an additional phone boundary, which is not present, is detected. Deletion error occurs when an actual phone boundary is not present in the alignment. Substitution error occurs when all the boundaries are captured properly but a phone is recognized wrongly as another phone.

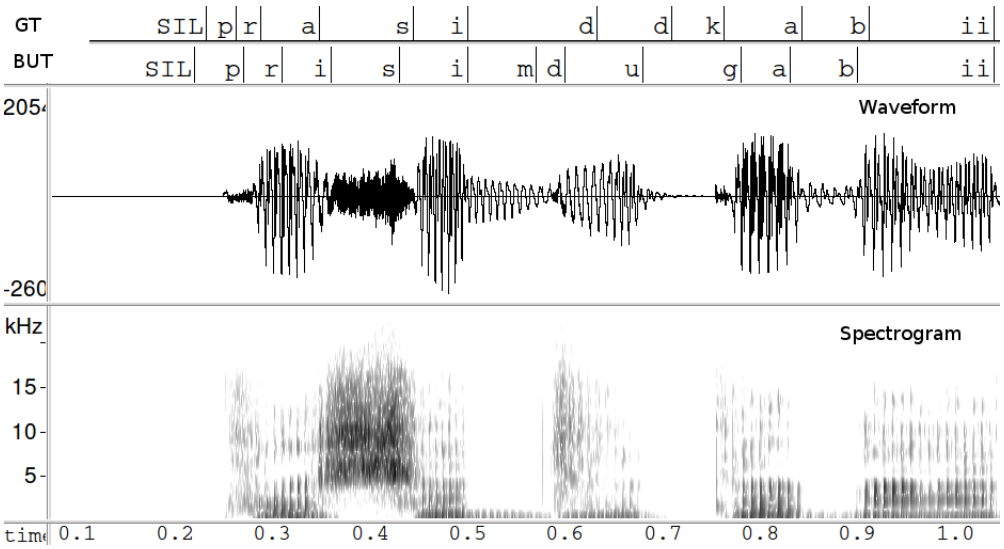


Figure 5.1: Phone boundaries obtained for a Hindi speech utterance using BUT phone recognizer. The first panel shows the actual phone boundaries (denoted as ground truth (GT), the second panel shows the phone boundaries obtained with BUT recognizer. Comparison between first and second panels shows the presence of insertion and substitution errors.

Manual annotation of phone boundaries is a difficult task. This requires native listeners of each language, who are also phoneticians, to mark the boundaries accurately in the speech utterances. This is tedious and time-consuming. Moreover, it is difficult to get experts for each language. However, for some of these languages, a small amount of manually labeled data is available. But the machine learning models, trained with less data, will not provide good phone boundaries. Phone segmentation in Indian language TTS systems is performed with the aid of various automatic and semi-automatic phoneme segmentation algorithms. These approaches include machine learning techniques like HMM flat-start segmentation [Lee et al., 1990, Young et al., 2002], HMM bootstrap segmentation [Brognaux and Drugman, 2016, Young et al., 2002], signal processing techniques like group delay based segmentation [Prasad et al., 2004], and hybrid segmentation algorithm [Shanmugam and Murthy, 2014b], which uses signal processing cues along with HMM-based forced Viterbi alignment, as discussed in Chapter 4.

5.2 Neural networks for phone modeling

In recent years, deep learning techniques like deep neural networks (DNN) and convolutional neural networks (CNN) are widely applied in ASR systems. The DNN/CNN modeling techniques outperform the GMMs in acoustic modeling as they can handle highly non-linear relationships between input and output [Hinton et al., 2012]. Nevertheless, these neural network techniques are not used for speech segmentation in the context of speech synthesis of Indian languages. In this work, an attempt is made to exploit the discriminative power of deep neural networks for phone segmentation in Indian language TTS systems.

In all the segmentation algorithms discussed in Section 4.3, GMMs are used for acoustic modeling within an HMM state. GMM decides the posterior probability of how well each HMM state fits a frame of acoustic coefficients. This posterior probability is converted to likelihood scores for performing the forward-backward algorithm during forced Viterbi alignment. For automatic speech recognition (ASR) systems DNNs outperformed GMMs in acoustic modeling. The inclusion of DNN/CNN improved the overall performance of ASR systems. In this work, GMMs are replaced by DNNs, and these systems are referred to as DNN-HMM systems [Hinton et al., 2012]. Instead of DNNs, CNNs are used for acoustic modeling and the systems are referred to as CNN-HMM systems [Abdel Hamid et al., 2012, Golik et al., 2015]. This motivated the implementation of DNN-HMM/CNN-HMM for speech segmentation approach in TTS systems.

5.2.1 Deep neural network (DNN)

DNNs are neural networks with more than two fully connected hidden layers of nodes between the input and output layers. A DNN is trained in two stages using the features[2] derived from speech data. In the first stage, an unsupervised restricted Boltzmann machine (RBM) is trained using the features. The RBM learns weights and the structures in input speech data. In the second stage, a neural network is initialized with the weights

[2]The most common training feature is filter bank energies.

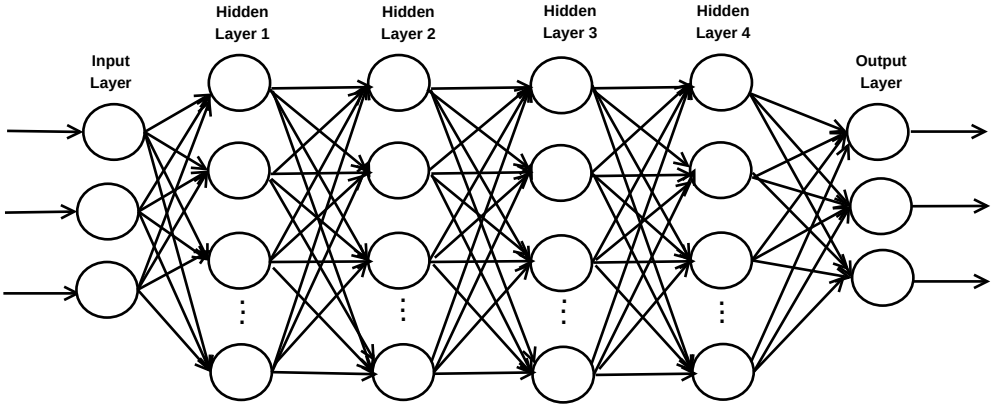


Figure 5.2: Schematic diagram of DNN

learned from the RBM. The block schematic of a DNN with 4 hidden layers is shown in Figure 5.2.

5.2.2 Convolutional neural network (CNN)

Unlike DNNs, CNNs are the neural network architectures with convolutional layer (convolutional layer + pooling layer (subsampling layer)) followed by few fully connected hidden layers. CNN reduces correlations among the different dimensions and performs convolution in the frequency domain. The block schematic of a CNN with 2 convolution and subsampling layers and 4 hidden layers is given in Figure 5.3.

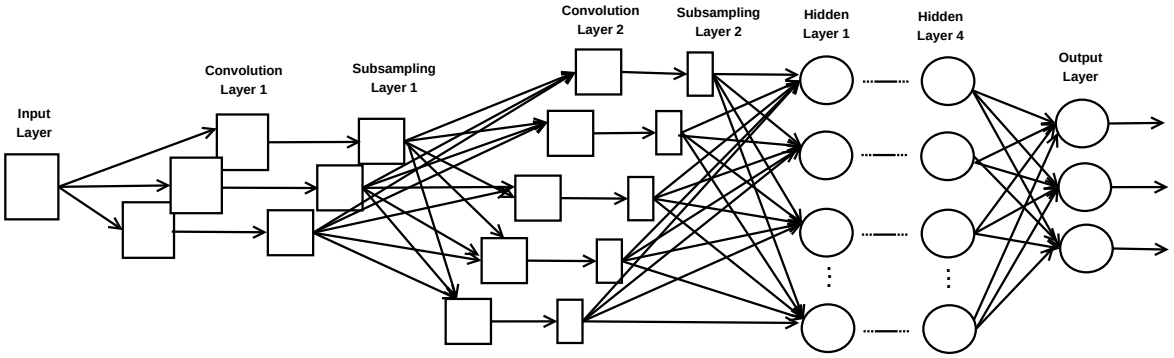


Figure 5.3: Schematic diagram of CNN

5.2.3 DNN with flat-start initialisation

Unlike GMM-HMM with flat start initialization, DNN/CNN-HMM flat start initialization will not give good phone boundaries. The phone boundaries obtained with DNN-HMM flat start for a Hindi utterance is shown in Figure 5.4. Hence, DNN/CNN-HMM based phone segmentation is performed with the initial phone models obtained from GMM-HMM segmentation.

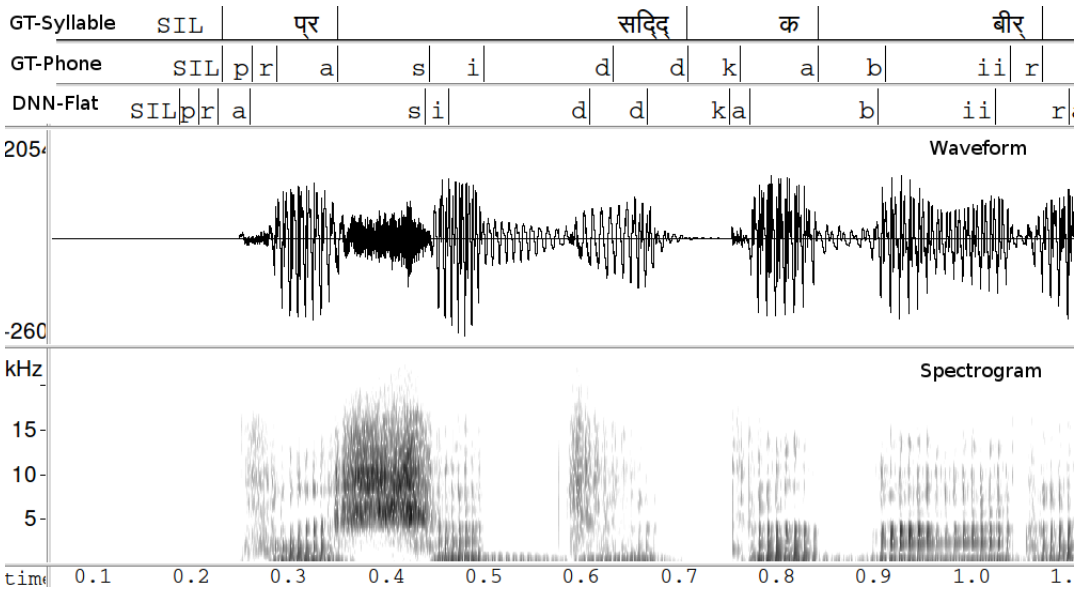


Figure 5.4: DNN flatstart segmentation

5.3 Motivation

Signal processing techniques, which are agnostic to the speaker, are capable of detecting accurate syllable boundaries from the speech data [Prasad et al., 2004]. These techniques are used widely in Indian languages for obtaining syllable boundaries for both ASR and TTS systems [Golda Brunet and Murthy, 2017, Janakiraman et al., 2010]. Syllable, the fundamental unit of speech production, can be detected using signal processing cues. The signal processing cues, group delay of short-term energy (STE) and sub-band spectral flux (SBSF) are capable of capturing the transitions between syllables. However, the boundaries obtained do not use the transcription.

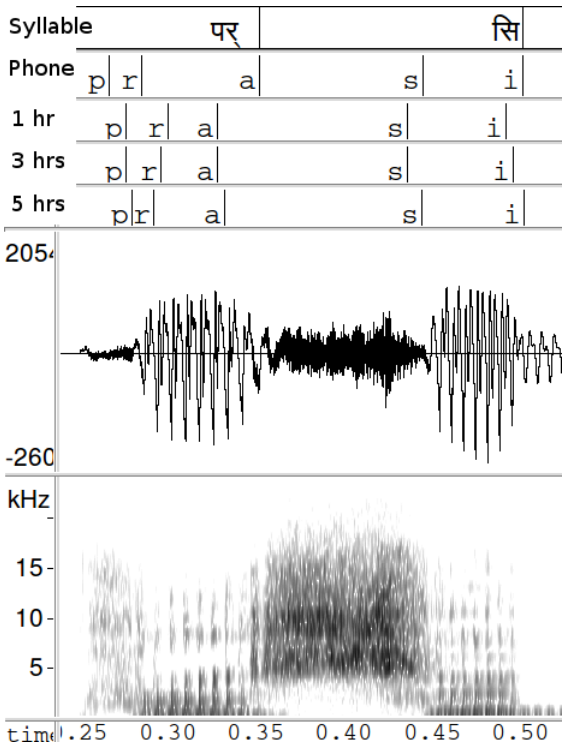


Figure 5.5: HMM flat start segmentation with different hours of training data. The first panel shows the syllable transcription (in UTF-8 encoding), the second panel shows the actual phone boundaries, the third, fourth, and fifth panels show the phone boundaries obtained with 1 hour, 3 hours, and 5 hours of data respectively.

On the other hand, machine learning approaches use text transcriptions also for speech segmentation. The number of syllable boundaries given will be based on the number of syllables present in the text. The accuracy of the boundaries depends on the amount of data used for training. Smaller the amount of data, poorer is the segmentation. This is illustrated in Figure 5.5, which shows the phone alignment obtained with 1 hour, 3 hours, and 5 hours of training data. The alignment becomes better with the increase in training data. But even with 5 hours of data, the boundaries obtained are not accurate. With any amount of data, signal processing cues can be used along with these machine learning algorithms to correct the boundaries given by the latter.

Large amount of data to train robust neural network models is not available for Indian languages. A hybrid approach which uses the deep neural networks in tandem with signal processing cues can improve the neural network models significantly for low resource languages. The proposed approaches which combine neural networks and signal

processing are detailed in the next section.

5.4 Proposed approaches

The phone boundaries obtained with DNN-HMM/CNN-HMM systems are better than those given by GMM-HMM systems. But some of the boundaries given by signal processing cues are even better than the DNN-HMM/CNN-HMM phone boundaries. Hence, signal processing cues are used to correct these boundaries. Similar to GMM-HMM framework, GD of STE and SBSF are used to correct the DNN-HMM/CNN-HMM phone boundaries, after applying the boundary correction rules as mentioned in Section 4.4.

The boundary corrections are applied to DNN-HMM/CNN-HMM systems in two different methods. In the first proposed method, the boundary correction is performed as an iterative process. The DNN-HMM/CNN-HMM systems are trained first to obtain phone boundaries, which are then corrected using signal processing cues. The number of iterations is set to 8 empirically. In the second proposed method, the speech utterance is first spliced into sub-utterances at the locations of phone boundaries that are corrected using signal processing cues. The DNN-HMM/CNN-HMM training is then performed on these sub-utterances to obtain better phone boundaries. This is a non-iterative procedure.

5.4.1 Signal processing cues in tandem with DNN/CNN-HMM: Iterative approach

The GD of STE and SBSF give accurate syllable boundaries for a subset of syllables as discussed in Section 4.4. The locations of these syllable boundaries which are accurate are obtained prior to DNN/CNN segmentation by performing GMM-HMM flat start segmentation followed by correction based on GD of STE and SBSF as shown in Figure 5.6 (Block I). The boundaries of the last phone of the corrected syllable boundaries are marked as GD corrected phone boundaries. A syllable to phone dictionary (syldict in Figure 5.6) is used to map from syllable to phoneme sequence.

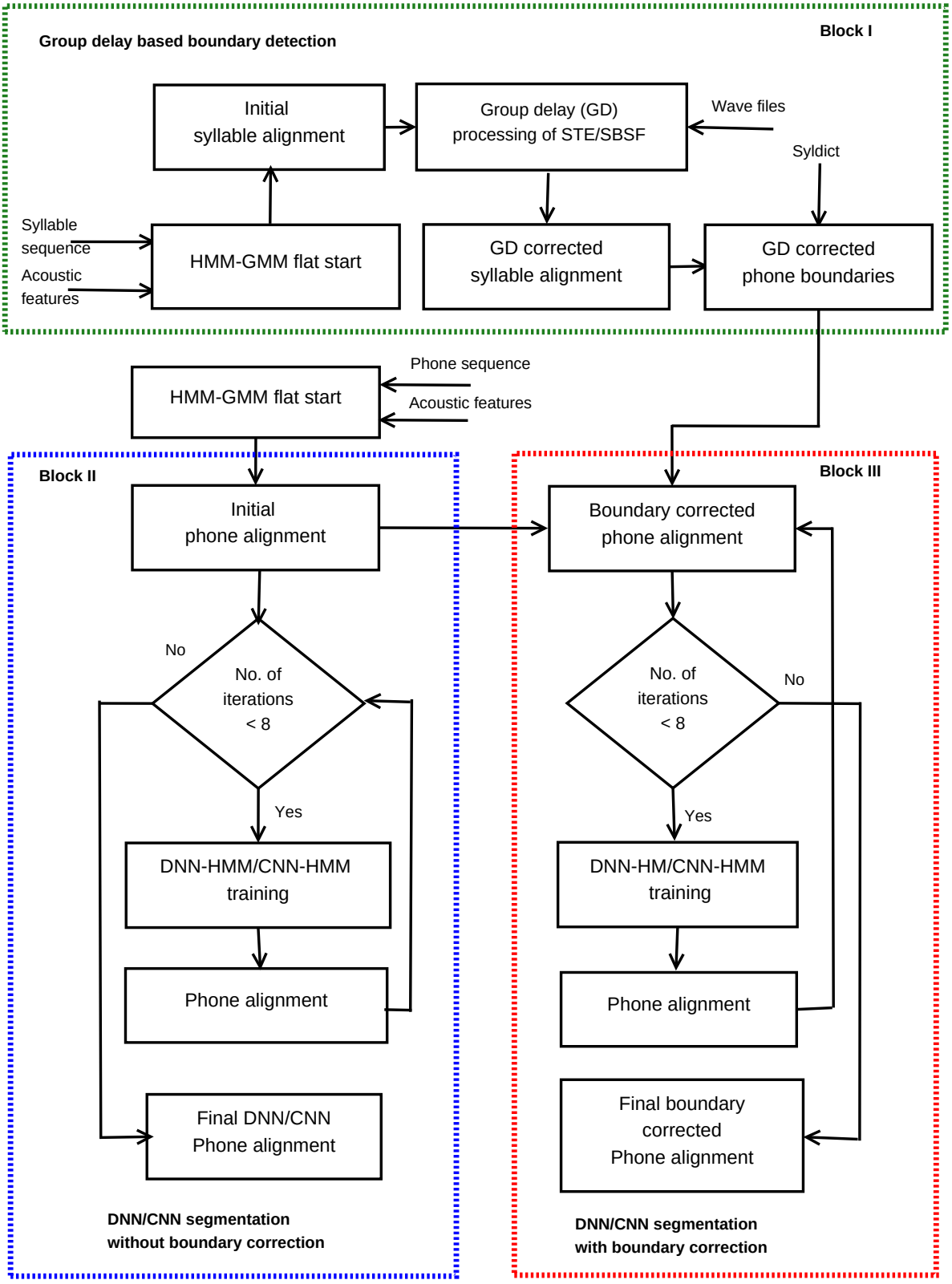


Figure 5.6: Block diagram of DNN/CNN algorithm with and without iterative boundary correction

The block diagram showing the DNN-HMM/CNN-HMM with boundary correction is given in Figure 5.6 (Block III). The phone boundaries from GMM-HMM-FS segmentation is used as the initial phone alignments for DNN/CNN training. Since DNN/CNNs learn alignments better with better initial alignments, these initial alignments are corrected using GD corrected phone boundaries before training these systems. The boundary of the phones corresponding to GD corrected phones is corrected. Other phone boundaries are kept as such. This corrected alignment is used for training the systems. Training is performed iteratively on the entire speech utterance. In each iteration, the phone boundaries obtained are corrected, either forward or backward, using GD corrected phone boundaries. After the last iteration, phone boundaries obtained from DNN/CNN-HMM are corrected again to get the final phone alignment.

5.4.2 Signal processing cues in tandem with DNN/CNN-HMM at sub-utterance level: Non-iterative approach

HMM phone models are robust when they are used for alignment on short utterances [Shanmugam, 2015]. Hence, in this approach, DNN-HMM/CNN-HMM segmentation at sub-utterance level instead of the entire utterance is performed. Initial alignment for DNN-HMM/CNN-HMM systems is obtained from GMM-HMM-FS segmentation as in the case of the iterative approach. The speech utterances are then spliced into sub-utterances at the location of GD-corrected phone boundaries. DNN-HMM/CNN-HMM training is performed on each sub-utterance rather than the entire utterance. The re-estimation is restricted to a much smaller sub-utterance which improve the phone models and in turn improves the boundaries. The block diagram of the whole procedure is shown in Figure 5.7.

5.5 Experimental setup

This section details the database used for the experiments, and the proposed segmentation algorithms based on DNN and CNN systems.

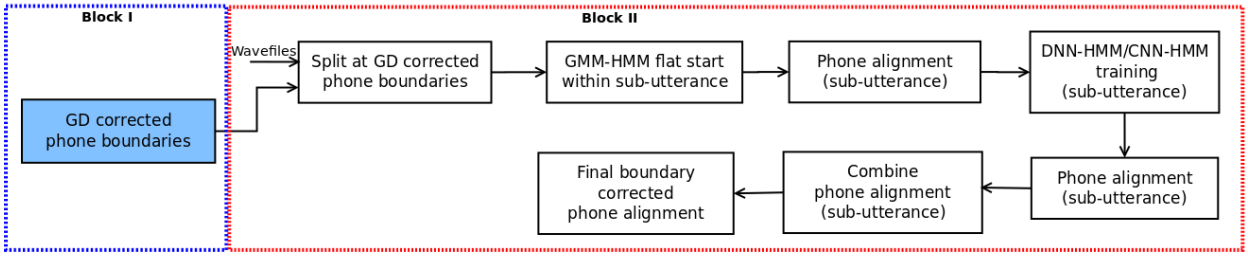


Figure 5.7: Block diagram of DNN-HMM/CNN-HMM segmentation with boundary correction: Non-iterative approach. The *GD corrected phone boundaries* are obtained using the procedure shown in Block I of Figure 5.6.

5.5.1 Dataset

The database consists of text sentences and the corresponding spoken waveforms. A subset of the Indic database[3] is used for the experiments [Baby et al., 2016b]. The details of the database are given in Table 5.1. The unified parser for Indian languages is used for grapheme to phoneme conversion of the native text [Baby et al., 2016a].

Table 5.1: Dataset

Language	Gender	Duration (in hrs)	No. of Utterances	No. of Distinct Phones
Bengali	Male	5.00	3093	49
	Female	5.00	3248	49
Gujarati	Male	5.00	1317	48
	Female	5.00	883	48
Hindi	Male	5.00	2192	55
	Female	5.00	2144	55
Kannada	Male	3.44	1289	48
	Female	3.83	1229	48
Malayalam	Male	5.00	3063	50
	Female	5.00	3524	55
Manipuri	Male	5.00	4458	33
	Female	5.00	4801	31
Rajasthani	Male	5.00	3108	50
	Female	5.00	2354	50
Tamil	Male	5.00	2073	34
	Female	5.00	1633	35
Telugu	Male	5.00	1833	47
	Female	5.02	831	47

[3]Available at https://www.iitm.ac.in/donlab/tts/database.php

5.5.2 Segmentation

Segmentation of speech data is performed using the baseline system and proposed approaches - without boundary correction and with boundary correction based on signal processing cues. The alignments given by the systems without boundary corrections are compared with the alignments given by the systems with boundary corrections. GMM-HMM-based approaches (GMM-HMM-FS and GMM-HMM-BC) are performed using HTK toolkit [Young, 1993]. DNN/CNN based approaches are performed using Kaldi toolkit [Povey et al., 2011]. The frame size of 25 ms with an overlap of 4 ms is used for processing speech data for all the experiments.

Baseline systems

1. **GMM-HMM flat start segmentation (GMM-HMM-FS)**

   The state mean and variance of all monophone HMM models are initialized with global mean and variance. 39-dimensional mel-frequency cepstral coefficients (MFCC) features are used as the feature for training HMMs. Vowels, consonants, and pauses are modeled as 5-state 2 mixtures, 3-state 2 mixtures, and 1-state 2 mixtures respectively. These values are obtained empirically.

2. **DNN-HMM flat start segmentation (DNN-HMM-FS)**

   All monophone HMMs are modeled using 5 states. The number of mixtures in each state differs for different phones, which is obtained from the data. 40-dimensional filter bank features are used as input for DNN-HMM. The neural network configuration used is given in Table 5.2.

3. **DNN-HMM/CNN-HMM segmentation without boundary correction (DNN-HMM/ CNN-HMM)**

   DNN-HMM segmentation without boundary correction is performed as an iterative process. The number of iterations is set to 8 empirically. The initial alignment is given by GMM-HMM flat start. Vowels, consonants, and silences are modeled with 5 states HMMs and a varying number of mixtures. 90% of the data is used for training and 10% is used for validation. The neural network configuration used is given in Table 5.2.

4. **GMM-HMM flat start segmentation with boundary correction (GMM-HMM-BC)**

Initial alignment for training this system is obtained from HMM-GMM flat start segmentation. 39-dimensional mel-frequency cepstral coefficients (MFCC) features are used for training HMM models. Vowels, consonants, and pauses are modeled as 5-state 2 mixtures, 3-state 2 mixtures, and 1-state 2 mixtures respectively. These values are obtained empirically. The boundary correction rules are applied based on threshold values of GD of STE and SBSF. Correction based on GD processing of STE is performed with two thresholds. A threshold of 0.5 is used if the beginning phone of the second syllable is an unvoiced stop consonant. A threshold value of 0.2 is used if end phone of the first syllable is an unvoiced stop consonant. Correction based on SBSF is performed with a threshold of 0.3. A threshold based on duration is also maintained for boundary correction. Boundary correction is performed only if the duration of present syllable and subsequent syllable is greater than 100 ms. This configuration is obtained from [Shanmugam, 2015].

Proposed systems

1. **DNN-HMM/CNN-HMM segmentation with iterative boundary correction (DNN-HMM-BC/CNN-HMM-BC)**

   The number of states and mixtures used for each monophone HMMs, the features used, and the configurations used for DNN and CNN are similar to that of DNN-HMM/CNN-HMM approach. The initial flat-start alignment from GMM-HMM is corrected using the spectral cues detailed in Section 4.4. This alignment is fed into DNN for training. After each iteration, the alignment obtained from the DNN is corrected using the spectral cues and again fed back into the DNN. The number of iterations is set to 8. After the 8^{th} iteration the phone boundaries obtained from DNN are again corrected with GD corrected phone boundaries. The neural network configuration used is given in Table 5.2.

2. **DNN-HMM/CNN-HMM segmentation with non-iterative boundary correction (DNN-HMM-BC-split/CNN-HMM-BC-split)**

   The syllable boundaries are detected using GD of STE and SBSF as mentioned in Section 4.4. The input speech files are spliced at these boundaries to obtain the new training data (at sub-utterance level). Initial flat-start alignment is obtained using GMM-HMM. This alignment is fed to the neural network for training. The configuration of DNN-HMM/CNN-HMM and the boundary correction criteria are same as that of DNN-HMM-BC/CNN-HMM-BC. The neural network configuration used is given in Table 5.2.

The neural network configuration used for the experiments is given in Table 5.2. This is from the recipe for Kaldi as obtained from [Povey et al., 2011].

Table 5.2: DNN and CNN configurations used for the experiments

	DNN	**CNN**
No. of Hidden Layers	6 (fully connected layers)	6 (2 convolution + 4 fully connected layers)
Feature	filter bank features	filter bank features + pitch
Dimension	40	40+3
Spliced frames	11	11
Convolution window size	-	8
Pooling window size	-	3 with overlap
Method	stochastic gradient descent and back propagation	stochastic gradient descent and back propagation
Mini-batch size	256	256
Feature map size	-	1^{st} Layer - 256 ; 2^{nd} Layer - 128

5.6 Result analysis

The various speech segmentation approaches described in Section 5.5.2 are analyzed in this section. Different segmentation approaches include the techniques without boundary correction (GMM-HMM-FS, DNN-HMM, and CNN-HMM), and those with boundary corrections (GMM-HMM-BC, DNN-HMM-BC, CNN-HMM-BC, DNN-HMM-BC-split, and CNN-HMM-BC-split). The phone boundaries obtained using the frameworks without boundary correction is compared with those using boundary correction.

Samples of segmentation using different approaches are detailed in Section 5.6.1. Two evaluation metrics are used for comparing different methods, namely, (1) number of boundaries detected and (2) DMOS subjective evaluation of the TTS systems built using each of these segmentation approaches. A significant improvement in segmentation accuracy is observed for the approaches with boundary correction using signal processing techniques. These are detailed in Sections 5.6.2 and 5.6.3.

5.6.1 Segmentation samples

Improvement in segmentation with signal processing based boundary correction approaches is illustrated with an example from Hindi and Tamil dataset below.

Hindi

Figures 5.8, 5.9, and 5.10 shows phone boundaries obtained for a syllable शोध् (shodh) with GMM-HMM (with and without boundary correction), DNN-HMM (with and without boundary correction) and CNN-HMM (with and without boundary correction) respectively. The actual syllable and phone boundaries are marked in the figures as ground truth (GT-syllable and GT-phone in the figures). It is observed that in all frameworks, GMM-HMM, DNN-HMM, and CNN-HMM, the syllable and phone boundaries improved with the use of signal processing based boundary correction.

In Figure 5.8 in case of the flatstart (GMM) boundary, both beginning and end phone boundary of the syllable is wrong (shown in red). The boundaries obtained using hybrid segmentation (GMM-BC) is much better (shown in blue). Many phone boundaries are detected correctly. Still, the end boundary of the syllable, *dh*, is not correct. In both Figure 5.9 and 5.10 similar observations can be made. In case of neural network systems (DNN and CNN), both the boundaries are detected wrongly. When iterative boundary correction is employed (DNN-BC, CNN-BC), the beginning boundary is detected correctly. And in case of the sub-utterance level approaches (DNN-split, CNN-split) both the begin and end boundary are getting detected correctly.

Tamil

Figures 5.11, 5.12, and 5.13 shows phone boundaries obtained with different approaches for the syllable து (tu). Similar to that of Hindi utterance, the approaches without boundary correction give both begin and end boundaries wrong. The signal processing based boundary correction (GMM-HMM-BC, DNN-HMM-BC, and CNN-HMM-BC), give the end boundary of the syllable correct. However, the beginning boundary of the syllable is wrong. The non-iterative boundary correction at sub-utterance level (DNN-HMM-BC-split and CNN-HMM-BC-split), gives both the boundaries correctly.

From these figures, it is clear that the proposed systems outperform the baseline systems. Also, many other phones boundaries detected are much better compared to the

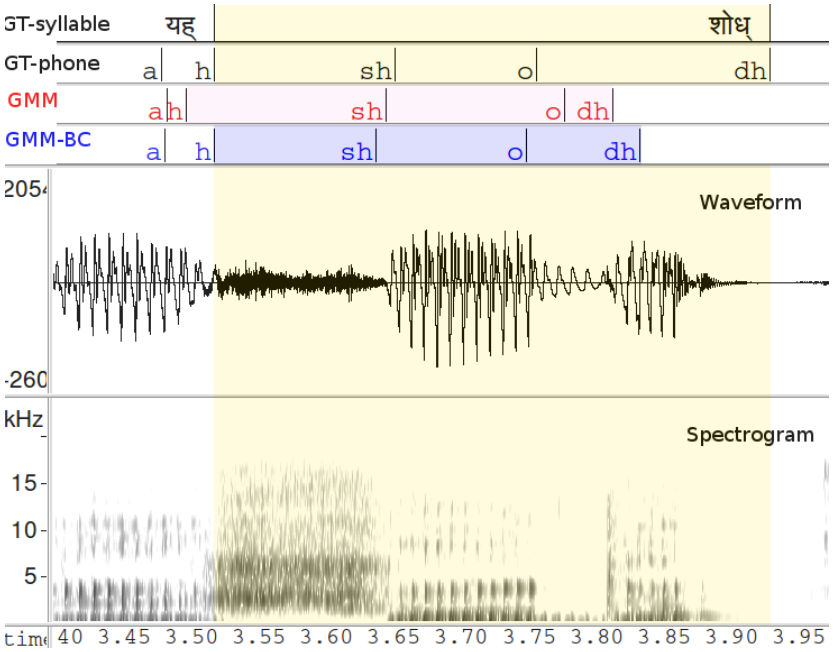


Figure 5.8: An example from part of a Hindi utterance, where syllable शोध् (shodh) is highlighted, with phone boundaries obtained using GMM-HMM and GMM-HMM-BC.

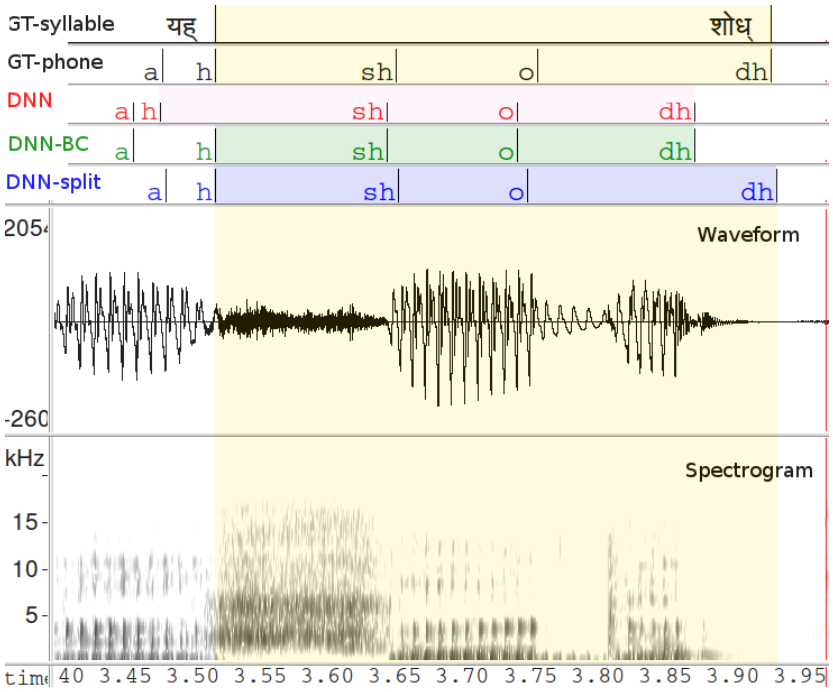


Figure 5.9: An example from part of a Hindi utterance, where syllable शोध् (shodh) is highlighted, with phone boundaries obtained using DNN-HMM and DNN-HMM with boundary correction.

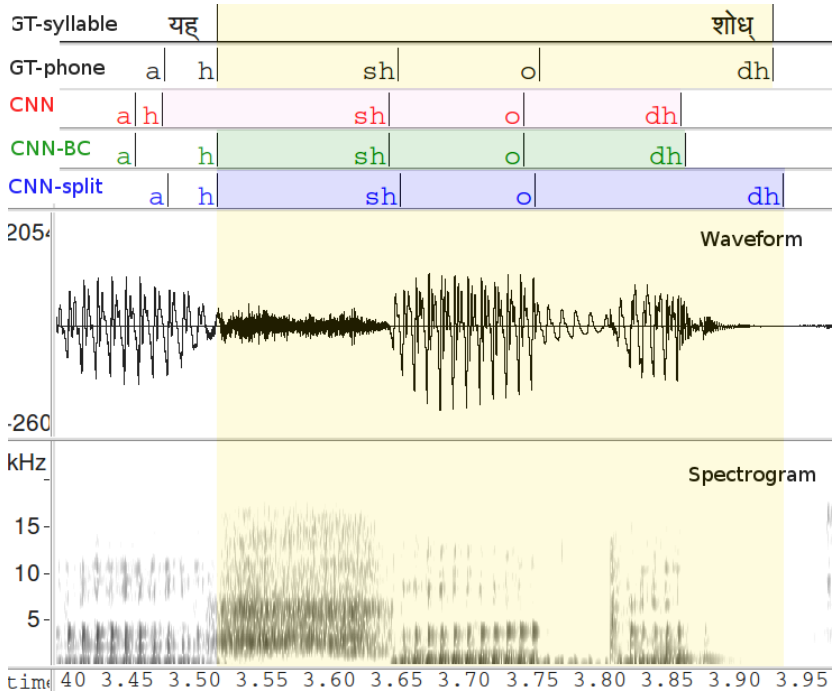


Figure 5.10: An example from part of a Hindi utterance, where syllable शोध् (shodh) is highlighted, with phone boundaries obtained using CNN-HMM and CNN-HMM with boundary correction.

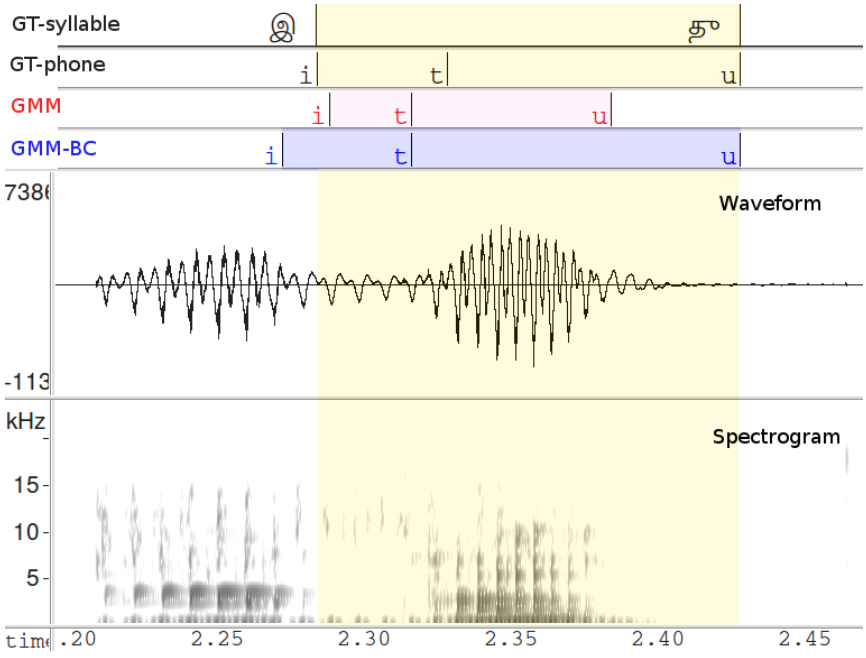


Figure 5.11: An example from part of a Tamil utterance, where syllable தூ (tu) is highlighted, with phone boundaries obtained using GMM-HMM and GMM-HMM with boundary correction.

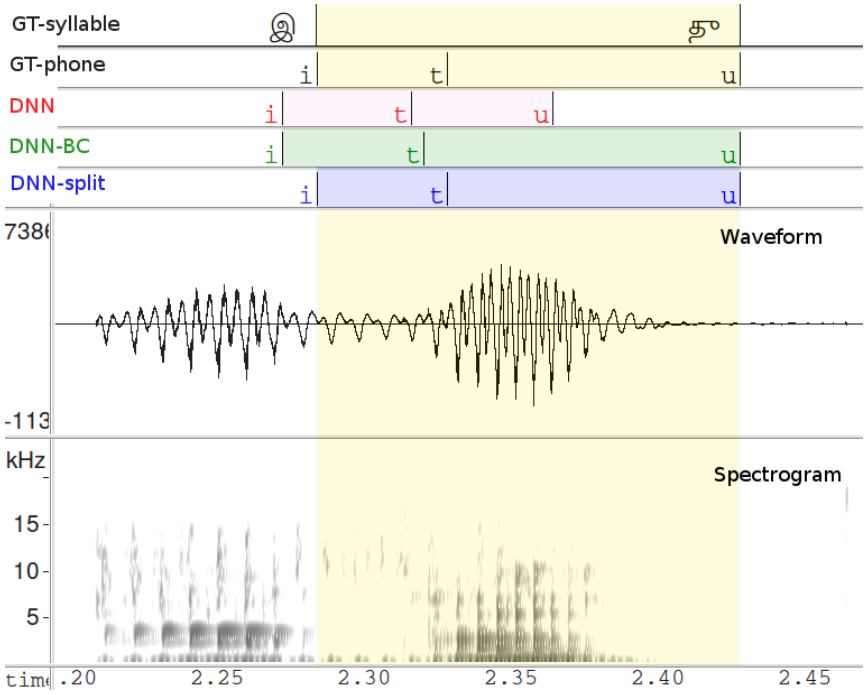


Figure 5.12: An example from part of a Tamil utterance, where syllable தூ (tu) is highlighted, with phone boundaries obtained using DNN-HMM and DNN-HMM with boundary correction.

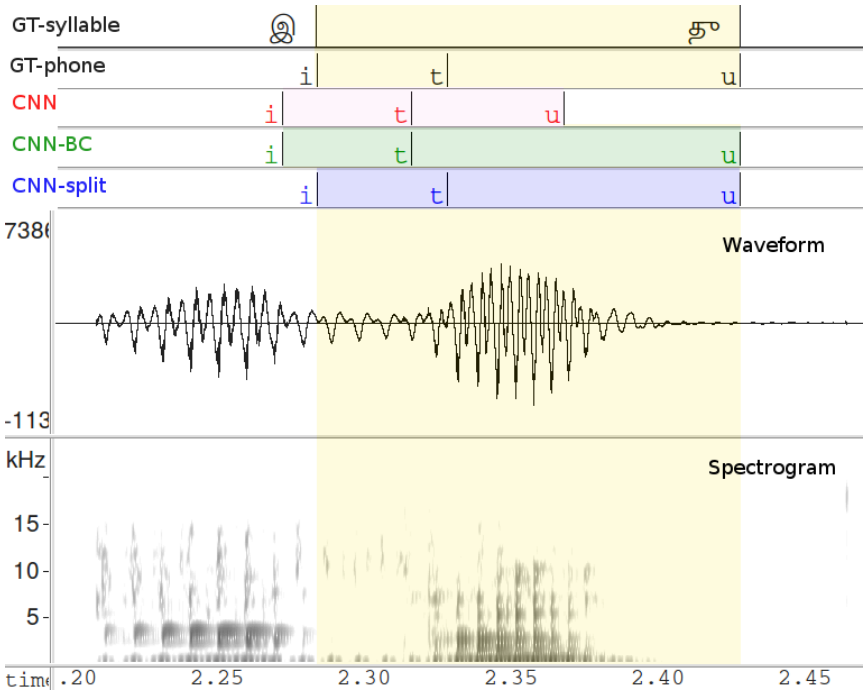


Figure 5.13: An example from part of a Tamil utterance, where syllable தூ (tu) is highlighted, with phone boundaries obtained using CNN-HMM and CNN-HMM with boundary correction.

baseline systems as observed from the spectrograms.

5.6.2 Boundary detection statistics

Table 5.3 shows the number and percentage of the boundaries accurately detected using spectral cues. These are the boundaries that are detected using the rules detailed in Section 4.4.4. It shows that about $5-15\%$ of the total boundaries are detected accurately using the spectral cues[4]. The boundaries detected are used to correct the baseline system's boundaries in an iterative or non-iterative manner as detailed in Section 5.4.

Table 5.3: Boundary detection statistics

Language	**Gender**	**% of boundaries detected by STE**		**% of boundaries detected by SBSF**		**Total boundaries in database**	
		Syllable	**Phone**	**Syllable**	**Phone**	**Syllable**	**Phone**
Bengali	Male	17.72%	7.79%	11.55%	5.07%	80930	184129
	Female	16.21%	7.31%	10.59%	4.77%	84482	187386
Gujarathi	Male	16.31%	7.21%	10.98%	4.85%	59378	134311
	Female	16.91%	7.59%	11.10%	4.98%	44815	99812
Hindi	Male	16.56%	7.46%	10.60%	4.78%	88066	195401
	Female	17.91%	8.32%	10.65%	4.94%	89158	191990
Kannada	Male	10.36%	4.88%	4.90%	2.31%	68677	145779
	Female	12.93%	6.22%	6.80%	3.27%	64756	134516
Malayalam	Male	4.55%	1.99%	2.57%	1.12%	89892	205749
	Female	10.21%	4.39%	4.58%	1.97%	101161	235484
Manipuri	Male	12.15%	6.21%	5.26%	2.69%	59174	115649
	Female	11.54%	5.94%	5.61%	2.89%	61557	119590
Rajasthani	Male	14.27%	6.64%	10.51%	4.89%	69495	149280
	Female	14.19%	6.57%	11.58%	5.36%	64950	140213
Tamil	Male	13.18%	5.96%	5.68%	2.56%	95611	211627
	Female	16.14%	7.34%	6.77%	3.08%	80555	177185
Telugu	Male	10.52%	4.88%	7.55%	3.51%	105404	227113
	Female	16.39%	7.90%	8.85%	4.27%	82073	170328

5.6.3 Text to speech (TTS) systems

A concatenative speech synthesis system based on cluster unit selection synthesis (USS) system and HMM-based parametric speech synthesis systems with STRAIGHT (HTS-

[4]These are not manually verified boundaries. These are the boundaries obtained using rule 1 and 2 detailed in Section 4.4.

STRAIGHT [Kawahara et al., 1999] [5]) are built with various segmentation approaches discussed. The quality of speech synthesized with various segmentation methods is compared.

The TTS systems built are evaluated qualitatively by conducting degradation mean opinion score (DMOS) listening test. In DMOS test, utterances synthesized using various approaches are played randomly to the listeners along with few natural utterances. The listeners are allowed to listen to the sentences only once and are asked to rate the quality of the system based on naturalness and intelligibility on a scale of 1-5, 5 being the best and 1 being the worst. The scores of the synthesized utterances are then normalized with respect to those of the natural utterances. The test is performed by 6-12 native participants of various languages. Individual scores of each participant are averaged to get the final DMOS score. Separate DMOS tests are conducted for USS and HTS-STRAIGHT frameworks. Due to the difficulty in getting people for subjective evaluation, DMOS evaluation of TTS systems is performed only for 4 languages, namely Hindi, Bengali, Kannada, and Malayalam.

The DMOS score for different languages are given in Table 5.5 and Table 5.4 for USS and HTS-STRAIGHT frameworks respectively. It is observed that the quality of synthesized speech improves with the use of signal processing cues in all the proposed frameworks.

Table 5.4: Degradation mean opinion scores for HTS-STRAIGHT systems

Language	Hindi	Bengali	Kannada	Malayalam
GMM-HMM-FS	3.09	2.66	3.06	2.93
GMM-HMM-BC	3.65	3.19	3.27	3.69
DNN-HMM-flat	3.40	2.43	2.55	2.68
CNN-HMM	3.80	2.93	3.50	3.57
CNN-HMM-BC	3.90	3.86	3.65	4.09
CNN-HMM-BC-split	4.00	3.68	3.57	4.16
DNN-HMM	3.45	3.03	3.03	3.31
DNN-HMM-BC	4.13	3.67	3.10	4.39
DNN-HMM-BC-split	4.05	3.92	3.53	4.12

[5]The default parameters of HTS-toolkit except pitch range is used for all HTS-STRAIGHT systems.

Table 5.5: Degradation mean opinion scores for USS systems

Language	Hindi	Bengali	Kannada	Malayalam
GMM-HMM-FS	2.89	2.46	3.50	2.00
GMM-HMM-BC	3.77	3.79	3.71	2.92
DNN-HMM-flat	2.80	2.60	2.45	1.80
CNN-HMM	3.27	2.90	3.36	2.92
CNN-HMM-BC	3.96	3.68	3.85	4.13
CNN-HMM-BC-split	4.15	3.62	3.72	3.67
DNN-HMM	3.15	2.95	3.06	2.84
DNN-HMM-BC	3.83	3.24	3.57	3.50
DNN-HMM-BC-split	3.64	3.45	3.30	3.72

5.7 Summary

Accurate segmentation of speech data is crucial for improving the quality of synthesized speech in both concatenative and parametric speech synthesis framework. Conventionally, GMM-HMM-based flat start segmentation approaches were used for performing speech segmentation. Owing to the low resource nature of these languages, the boundaries obtained were inaccurate, and the speech synthesis quality was poor. Unlike ASR, where the objective is to obtain answers to queries, thus giving leeway for errors in transcription, text to speech synthesis requires accurate boundaries for synthesis as the consumer of the synthesized output is the human being.

In this work, the importance of spectral cues in automatic speech segmentation for Indian language TTS systems is shown. The boundaries given by spectral cues are used in tandem with machine learning techniques such as DNN-HMM, and CNN-HMM to improve the phoneme segmentation. TTS systems are built using the obtained phoneme segments. Results of the listening test show that the quality is improved after using signal processing cues along with machine learning techniques.

CHAPTER 6

Conclusion and Future Work

6.1 Summary

The work carried out in this thesis mainly focuses on unifying TTS synthesis system building process across the languages. Especially in case of Indian languages, which are digitally low resource and rich in linguistic diversity, a unified approach will make it easier to build TTS systems. Two main subsystems of the TTS system, parser, and segmentation, are improved.

A unified parser is created which can parse Indian language text into the CLS. New set of rules are defined to parse words more accurately. Language-specific rules are added to increase the accuracy further. New languages can be added by just creating a mapping from the script to CLS and identifying language-specific rules.

Speech synthesis requires accurate boundaries to get high-quality output. Segmentation of training data into accurate time aligned phones/syllables play a crucial role in TTS systems. Although machine learning methods can give phone boundaries, they are not good enough to train a good TTS system. Moreover, Indian languages are low resource, so depending solely on machine learning is not a good option. Signal processing cues are proven to give accurate boundaries in certain cases. Also, the accuracy of signal processing cues doesn't depend on the size of data. In this work, an attempt is made to improve machine learning frameworks with the help of signal processing cues.

6.2 Criticisms of the thesis

Parsers usually work at the word level. Re-syllabification that occur across words cannot be handled by unified parser.

Only around 10-25% of the phoneme boundaries are detected correctly using signal processing cues.

6.3 Future work

Some of the possible extensions for this thesis work are listed below.

- More language-specific rules for languages can be explored.
- New languages can be added to the unified parser.
- Since the parser uses CLS, TTS systems with code-switching and code-mixing across languages can be created.
- Only around 10-25% of the boundaries are detected correctly using signal processing cues. A detailed acoustic analysis may reveal more rules and techniques.

Appendices

APPENDIX A

Online resources

All the resources developed as part of this thesis are made publicly available at https://www.iitm.ac.in/donlab/tts/index.php. The unified parser is developed in C language using lex and yacc framework. The neural network based segmentation software is created with the help of Kaldi toolkit. The front-end text processor used for TTS is Festival and the back-end is either HTS-STRAIGHT or USS. Instructions for installation and usage of the developed tools are given along with the software package. All these softwares are tested in ubuntu 14.04.

A.1 Unified parser

The resources are hosted at https://www.iitm.ac.in/donlab/tts/unified.php. Language specific rules can be added in the *rules* folder (samples are given). For making use of dictionary, the words can be added in the *dict* folder in the corresponding language file. For adding a new language, the *common* file needs to be updated with the phoneme mapping and the language name should be added accordingly in $unified.y$ file. Flex and bison needs to be installed before compiling the unified-parser source code.

A.2 DNN based segmentation

The resources are hosted at https://www.iitm.ac.in/donlab/tts/hybridSeg.php. This uses TIMIT Kaldi recipe script. Alignment modifications are done after each iteration. These scripts are in Perl, Python, and Bash. Kaldi and Hybrid segmentation are required for this tool to work.

REFERENCES

[Abdel Hamid et al., 2012] Abdel Hamid, O., Mohamed, A. R., Jiang, H., and Penn, G. (2012). Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4277–4280. IEEE.

[Baby et al., 2018] Baby, A., DS, K. P., and Murthy, H. A. (2018). Signal processing cues to improve automatic speech recognition for low resource indian languages. In *Proc. The 6th Intl. Workshop on Spoken Language Technologies for Under-Resourced Languages*, pages 25–29.

[Baby et al., 2016a] Baby, A., Nishanthi, N. L., Thomas, A. L., and Murthy, H. A. (2016a). A unified parser for developing Indian language text to speech synthesizers. In *International Conference on Text, Speech and Dialogue*, pages 514–521.

[Baby et al., 2016b] Baby, A., Thomas, A. L., Nishanthi, N. L., and Consortium, T. (2016b). Resources for Indian languages. In *CBBLR – Community-Based Building of Language Resources*, pages 37–43, Brno, Czech Republic. Tribun EU.

[Black et al., 1998] Black, A., Taylor, P., Caley, R., and Clark, R. (1998). The festival speech synthesis system.

[Black and Kominek, 2009] Black, A. W. and Kominek, J. (2009). Optimizing segment label boundaries for statistical speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3785–3788.

[Brognaux and Drugman, 2016] Brognaux, S. and Drugman, T. (2016). HMM-based speech segmentation: improvements of fully automatic approaches. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 24:5–15.

[Conrad, 1999] Conrad, S. (1999). The importance of corpus-based research for language teachers. *System*, 27(1):1 – 18.

[Copestake and Flickinger, 2000] Copestake, A. and Flickinger, D. (2000). An open source grammar development environment and broad-coverage English grammar using hpsg. In *Language Resources and Evaluation Conference (LREC)*, pages 591–600.

[Cui et al., 2012] Cui, X., Xue, J., Chen, X., Olsen, P. A., Dognin, P. L., Chaudhari, U. V., Hershey, J. R., and Zhou, B. (2012). Hidden Markov acoustic modeling with bootstrap and restructuring for low-resourced languages. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(8):2252–2264.

[Dash and Chaudhuri, 2001] Dash, N. S. and Chaudhuri, B. B. (2001). Why do we need to develop corpora in Indian languages. In *International Conference on SCALLA*, volume 11.

[Deivapalan et al., 2008] Deivapalan, P., Jha, M., Guttikonda, R., and Murthy, H. A. (2008). Donlabel: an automatic labeling tool for Indian languages. In *National Conference on Communications (NCC)*, pages 263–266.

[Dudley, 1939] Dudley, H. (1939). Remaking speech. *The Journal of the Acoustical Society of America*, 11(2):169–177.

[Durling and Lumsden, 2008] Durling, S. and Lumsden, J. (2008). Speech recognition use in healthcare applications. In *6th international conference on advances in mobile computing and multimedia*, pages 473–478. ACM.

[Eskenazi, 2009] Eskenazi, M. (2009). An overview of spoken language technology for education. *Speech Communication*, 51(10):832 – 844. Spoken Language Technology for Education.

[Fant, 1968] Fant, G. (1968). Analysis and synthesis of speech processes. *Manual of Phonetics*, pages 173–276.

[Ganapathiraju et al., 2001] Ganapathiraju, A., Hamaker, J., Picone, J., Ordowski, M., and Doddington, G. R. (2001). Syllable-based large vocabulary continuous speech recognition. *IEEE Transactions on speech and audio processing*, 9(4):358–366.

[Godfrey et al., 1992] Godfrey, J. J., Holliman, E. C., and McDaniel, J. (1992). Switchboard: Telephone speech corpus for research and development. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 517–520, Washington, DC, USA. IEEE Computer Society.

[Golda Brunet and Murthy, 2017] Golda Brunet, R. and Murthy, H. A. (2017). Transcription correction using group delay processing for continuous speech recognition. *Circuits, Systems, and Signal Processing*, pages 1531–5878.

[Golik et al., 2015] Golik, P., Tüske, Z., Schlüter, R., and Ney, H. (2015). Convolutional neural networks for acoustic modeling of raw time signal in LVCSR. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 26–30.

[Greenberg, 1999] Greenberg, S. (1999). Speaking in shorthand - A syllable-centric perspective for understanding pronunciation variation. *Speech Communication*, 29(2-4):159–176.

[Hinton et al., 2012] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

[Hirsch and Pearce, 2000] Hirsch, H. G. and Pearce, D. (2000). The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*, pages 29–32.

[Huang et al., 1993] Huang, X., Alleva, F., Hon, H. W., Hwang, M. Y., Lee, K. F., and Rosenfeld, R. (1993). The SPHINX-II speech recognition system: an overview. *Computer Speech & Language*, 7(2):137–148.

[Hunt and Black, 1996] Hunt, A. J. and Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 373–376. IEEE.

[Imai et al., 1983] Imai, S., Sumita, K., and Furuichi, C. (1983). Mel log spectrum approximation (mlsa) filter for speech synthesis. *Electronics and Communications in Japan (Part I: Communications)*, 66(2):10–18.

[Janakiraman et al., 2010] Janakiraman, R., Kumar, J. C., and Murthy, H. A. (2010). Robust syllable segmentation and its application to syllable-centric continuous speech recognition. In *National Conference on Communications (NCC)*, pages 1–5. IEEE.

[Joy et al., 2014] Joy, N. M., Abraham, B., Navneeth, K., and Umesh, S. (2014). Cross-lingual acoustic modeling for Indian languages based on subspace gaussian mixture models. In *National Conference on Communications (NCC)*, pages 1–5. IEEE.

[Kawahara et al., 2001] Kawahara, H., Estill, J., and Fujimura, O. (2001). Aperiodicity extraction and control using mixed mode excitation and group delay manipulation for a high quality speech analysis, modification and synthesis system straight. In *Second International Workshop on Models and Analysis of Vocal Emissions for Biomedical Applications*, pages 59–64.

[Kawahara et al., 1999] Kawahara, H., Katsuse, I. M., and de Cheveigne, A. (1999). Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds. *Speech Communication*, 27(3-4):187–207.

[Kim and Conkie, 2002] Kim, Y. J. and Conkie, A. (2002). Automatic segmentation combining an HMM-based approach and spectral boundary correction. In *International Conference on Spoken Language Processing*, pages 145–148.

[Kishore et al., 2002] Kishore, P., Kumar, R., and Sangal, R. (2002). A data driven synthesis approach for Indian languages using syllable as basic unit. In *International Conference on Natural Language Processing (ICON)*, pages 311–316.

[Kishore and Black, 2003] Kishore, S. P. and Black, A. W. (2003). Unit size in unit selection speech synthesis. In *European Conference on Speech Communication and Technology (EUROSPEECH)*, pages 1317–1320.

[Kominek et al., 2003] Kominek, J., Bennett, C. L., and Black, A. W. (2003). Evaluating and correcting phoneme segmentation for unit selection synthesis. In *European Conference on Speech Communication and Technology (EUROSPEECH)*, pages 313–316.

[Kominek and Black, 2004] Kominek, J. and Black, A. W. (2004). The CMU arctic speech databases. In *Fifth ISCA Workshop on Speech Synthesis*, pages 223–224.

[Ladefoged and Johnson, 2014] Ladefoged, P. and Johnson, K. (2014). *A course in phonetics*. Nelson Education.

[Lakshmi and Murthy, 2006] Lakshmi, A. and Murthy, H. A. (2006). A syllable based continuous speech recognizer for Tamil. In *International Conference on Spoken Language Processing*, pages 1878–1881.

[Lavanya et al., 2005] Lavanya, P., Kishore, P., and Madhavi, G. T. (2005). A simple approach for building transliteration editors for Indian languages. *Journal of Zhejiang University Science A*, 6(11):1354–1361.

[Lee et al., 1990] Lee, C. H., Rabiner, L. R., Pieraccini, R., and Wilpon, J. G. (1990). Acoustic modeling for large vocabulary speech recognition. *Computer Speech & Language*, 4:127–165.

[Lee, 2006] Lee, K. S. (2006). MLP-based phone boundary refining for a TTS database. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(3):981–989.

[Levine et al., 1992] Levine, J. R., Mason, T., and Brown, D. (1992). *Lex & yacc*. O'Reilly Media, Inc.

[Lim, 1979] Lim, J. (1979). Spectral root homomorphic deconvolution system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(3):223–233.

[Lo and Wang, 2007] Lo, H. Y. and Wang, H. M. (2007). Phonetic boundary refinement using support vector machine. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages IV–933. IEEE.

[Murthy and Yegnanarayana, 1991] Murthy, H. A. and Yegnanarayana, B. (1991). Speech processing using group delay functions. *Signal Processing*, 22(3):259–267.

[Murthy and Yegnanarayana, 2011] Murthy, H. A. and Yegnanarayana, B. (2011). Group delay functions and its applications in speech technology. *Sadhana*, 36(5):745–782.

[Nagarajan and Murthy, 2004] Nagarajan, T. and Murthy, H. A. (2004). Subband-based group delay segmentation of spontaneous speech into syllable-like units. *EURASIP Journal on Advances in Signal Processing*, 2004(17):1–12.

[Nagarajan et al., 2003] Nagarajan, T., Prasad, V. K., and Murthy, H. A. (2003). Minimum phase signal derived from root cepstrum. *Electronics Letters*, 39(12):941–942.

[Ogbureke and Carson Berndsen, 2009] Ogbureke, K. U. and Carson Berndsen, J. (2009). Improving initial boundary estimation for HMM-based automatic phonetic segmentation. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 884–887.

[Patil et al., 2013] Patil, H. A., Patel, T. B., Shah, N. J., Sailor, H. B., Krishnan, R., Kasthuri, G., Nagarajan, T., Christina, L., Kumar, N., Raghavendra, V., et al. (2013). A syllable-based framework for unit selection synthesis in 13 Indian languages. In *Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, pages 1–8. IEEE.

[Paul and Baker, 1992] Paul, D. B. and Baker, J. M. (1992). The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the workshop on Speech and Natural Language*, pages 357–362. Association for Computational Linguistics.

[Plauche et al., 2006] Plauche, M., Nallasamy, U., Pal, J., Wooters, C., and Ramachandran, D. (2006). Speech recognition for illiterate access to information and technology. In *International Conference on Information and Communication Technologies and Development*, pages 83–92.

[Post et al., 2012] Post, M., Callison Burch, C., and Osborne, M. (2012). Constructing parallel corpora for six Indian languages via crowdsourcing. In *Seventh Workshop on Statistical Machine Translation*, pages 401–409. Association for Computational Linguistics.

[Povey et al., 2011] Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., et al. (2011). The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number EPFL-CONF-192584. IEEE Signal Processing Society.

[Pradhan et al., 2015] Pradhan, A., Prakash, A., Shanmugam, S. A., Kasthuri, G., Krishnan, R., and Murthy, H. A. (2015). Building speech synthesis systems for Indian languages. In *National Conference on Communications (NCC)*, pages 1–6. IEEE.

[Pradhan et al., 2013] Pradhan, A., Shanmugam, A., Prakash, A., Veezhinathan, K., and Murthy, H. A. (2013). A syllable based statistical text to speech system. In *European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE.

[Prakash et al., 2014] Prakash, A., Reddy, M. R., Nagarajan, T., and Murthy, H. A. (2014). An approach to building language-independent text-to-speech synthesis for Indian languages. In *National Conference on Communications (NCC)*, pages 1–5. IEEE.

[Prasad et al., 2004] Prasad, V. K., Nagarajan, T., and Murthy, H. A. (2004). Automatic segmentation of continuous speech using minimum phase group delay functions. *Speech Communication*, 42(3):429–446.

[Rabiner and Juang, 1993] Rabiner, L. and Juang, B. H. (1993). *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[Raghavendra et al., 2008a] Raghavendra, E. V., Desai, S., Yegnanarayana, B., Black, A. W., and Prahallad, K. (2008a). Global syllable set for building speech synthesis in Indian languages. In *Spoken Language Technology Workshop*, pages 49–52. IEEE.

[Raghavendra et al., 2008b] Raghavendra, E. V., Yegnanarayana, B., Black, A. W., and Prahallad, K. (2008b). Building sleek synthesizers for multi-lingual screen reader. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1865–1868.

[Raina et al., 2004] Raina, A. M., Mukerjee, A., Goyal, P., and Shukla, P. (2004). A unified computational lexicon for Hindi-English code-switching. In *International Conference on Natural Language Processing*, pages 19–22.

[Ramani et al., 2013] Ramani, B., Christina, S. L., Rachel, G. A., Solomi, V. S., Nandwana, M. K., Prakash, A., Shanmugam, S. A., Krishnan, R., Kishore, S., Samudravijaya, K., et al. (2013). A common attribute based unified hts framework for speech synthesis in Indian languages. In *ISCA Workshop on Speech Synthesis*, pages 311–316.

[Riley, 1991] Riley, M. D. (1991). Tree-based modelling for speech synthesis. In *ESCA Workshop on Speech Synthesis*, pages 229–232.

[Schultz et al., 2013] Schultz, T., Vu, N. T., and Schlippe, T. (2013). Globalphone: A multilingual text & speech database in 20 languages. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8126–8130. IEEE.

[Schwarz et al., 2006] Schwarz, P., Matejka, P., Burget, L., and Glembek, O. (2006). Phoneme recognizer based on long temporal context. *Speech Processing Group, Faculty of Information Technology, Brno University of Technology.[Online]. Available: http://speech.fit.vutbr.cz/en/software*.

[Sethy, 2002] Sethy, A. (2002). Refined speech segmentation for concatenative speech synthesis. In *International Conference on Spoken Language Processing*, pages 149–153.

[Shanmugam, 2015] Shanmugam, S. A. (2015). A hybrid approach to segmentation of speech using signal processing cues and hidden Markov models. M. S. Thesis, Department of Computer Science Engineering, IIT Madras, India.

[Shanmugam and Murthy, 2014a] Shanmugam, S. A. and Murthy, H. A. (2014a). Group delay based phone segmentation for HTS. In *National Conference on Communications (NCC)*, pages 1–6. IEEE.

[Shanmugam and Murthy, 2014b] Shanmugam, S. A. and Murthy, H. A. (2014b). A hybrid approach to segmentation of speech using group delay processing and HMM-based embedded reestimation. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1648–1652.

[Shrishrimal et al., 2012] Shrishrimal, P., Deshmukh, R. R., and Waghmare, V. (2012). Indian language speech database: a review. *International Journal of Computer Application (IJCA)*, 47(5):17–21.

[Singh, 2006] Singh, A. K. (2006). A computational phonetic model for Indian language scripts. In *Constraints on Spelling Changes: Fifth International Workshop on Writing Systems*, pages 1–19.

[Siniscalchi et al., 2013] Siniscalchi, S. M., Reed, J., Svendsen, T., and Lee, C.-H. (2013). Universal attribute characterization of spoken languages for automatic spoken language recognition. *Computer Speech & Language*, 27(1):209–227.

[Stolcke et al., 2014] Stolcke, A., Ryant, N., Mitra, V., Yuan, J., Wang, W., and Liberman, M. (2014). Highly accurate phonetic segmentation using boundary correction models and system fusion. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5552–5556.

[Tachbelie et al., 2014] Tachbelie, M. Y., Abate, S. T., and Besacier, L. (2014). Using different acoustic, lexical and language modeling units for ASR of an under-resourced language–amharic. *Speech Communication*, 56:181–194.

[Tachbelie et al., 2012] Tachbelie, M. Y., Abate, S. T., Besacier, L., and Rossato, S. (2012). Syllable-based and hybrid acoustic models for amharic speech recognition. In *Spoken Language Technologies for Under-Resourced Languages*, pages 5–10.

[Wichmann and Fligelstone, 2014] Wichmann, A. and Fligelstone, S. (2014). *Teaching and language corpora*. Routledge.

[Wikipedia, 2018a] Wikipedia (2018a). Articulatory phonetics — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Articulatory_phonetics&oldid=838454264. [Online; accessed 15-June-2018].

[Wikipedia, 2018b] Wikipedia (2018b). Languages of India - wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Languages_of_India&oldid=831676831. [Online; accessed 22-March-2018].

[Wikipedia, 2018c] Wikipedia (2018c). List of countries by english-speaking population — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=List_of_countries_by_English-speaking_population&oldid=870313444. [Online; accessed 26-November-2018].

[Wikipedia, 2018d] Wikipedia (2018d). Literacy in india — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Literacy_in_India&oldid=870068272. [Online; accessed 26-November-2018].

[Wikipedia, 2018e] Wikipedia (2018e). Manner of articulation — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Manner_of_articulation&oldid=840388650. [Online; accessed 17-June-2018].

[Wilpon et al., 1990] Wilpon, J. G., Rabiner, L. R., Lee, C. H., and Goldman, E. (1990). Automatic recognition of keywords in unconstrained speech using hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(11):1870–1878.

[Young et al., 2002] Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., et al. (2002). The HTK book. *Cambridge university engineering department*, 3:175.

[Young, 1993] Young, S. J. (1993). *The HTK hidden Markov model toolkit: Design and philosophy*. University of Cambridge, Department of Engineering.

[Yuan et al., 2013] Yuan, J., Ryant, N., Liberman, M., Stolcke, A., Mitra, V., and Wang, W. (2013). Automatic phonetic segmentation using boundary models. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2306–2310, Lyon.

[Zen et al., 2009] Zen, H., Tokuda, K., and Black, A. W. (2009). Statistical parametric speech synthesis. *Speech Communication*, 51(11):1039–1064.

[Zue et al., 1990] Zue, V., Seneff, S., and Glass, J. (1990). Speech database development at MIT: Timit and beyond. *Speech Communication*, 9(4):351–356.

LIST OF PAPERS BASED ON THESIS

1. Arun Baby, Nishanthi N L, Anju Leela Thomas, and Hema A Murthy , "A Unified Parser for Developing Indian Language Text to Speech Synthesizers", in *International Conference on Text, Speech, and Dialogue (TSD)*, pp. 514-521, Czech Republic, September 2016.

2. Arun Baby, Jeena J Prakash, Rupak Vignesh and Hema A Murthy , "Deep Learning Techniques in Tandem with Signal Processing Cues for Phonetic Segmentation for Text to Speech Synthesis in Indian Languages", in *INTERSPEECH 2017*, pp. 3817-3821, Stockholm, Sweden, August 2017.

3. Arun Baby, Jeena J Prakash and Hema A Murthy, "A Hybrid Approach to Neural Networks based Speech Segmentation", in *Frontiers of Research in Speech and Music (FRSM)*, Rourkela, India, December 2017.

OTHER PUBLICATIONS

1. Anusha Prakash, Arun Baby, Aswin Shanmugam S., Jeena J Prakash, Nishanthi N. L., Raghava Krishnan K., Rupak Vignesh Swaminathan and Hema A. Murthy, "Blizzard Challenge 2015 : Submission by DONLab, IIT Madras", in *Blizzard Challenge*, Berlin, Germany, September 2015.

2. Arun Baby, Anju Leela Thomas, Nishanthi N L, and TTS Consortium , "Resources for Indian Languages", in *Community-based Building of Language Resources Workshop (CBBLR), International Conference on Text, Speech, and Dialogue (TSD)*, pp. 37-43, Brno, Czech Republic, September 2016.

3. Atish Ghone, Rachana Nerpagar, Pranaw Kumar, Arun Baby, Aswin Shanmugam, Sasikumar Mukundan and Hema A Murthy , "TBT (Toolkit to Build TTS): A High Performance Framework to build Multiple Language HTS Voice", in *INTERSPEECH 2017 (Show and Tell)*, pp. 3427-3428, Stockholm, Sweden, August 2017.

4. Arun Baby, Anju Thomas, Jeena J Prakash, Anusha Prakash and Hema A Murthy, "Speech Synthesis in Indian Languages and Future Perspectives", in *Global Conference on Cyberspace (GCCS)*, Delhi, India, November 2017.

5. Arun Baby, Karthik Pandia DS and Hema A Murthy, "Signal Processing Cues to Improve Automatic Speech Recognition for Low Resource Indian Languages", *SLTU 2018*, Gurugram, India, August 2018.

6. Anju Leela Thomas, Anusha Prakash, Arun Baby, and Hema A Murthy, "Code-switching in Indic Speech Synthesisers", *INTERSPEECH 2018*, Hyderabad, India, September 2018.

GENERAL TEST COMMITTEE

CHAIRPERSON : Dr. Sreenivasa Kumar P
Department of Computer Science and Engineering

GUIDE : Dr. Hema A Murthy
Department of Computer Science and Engineering

MEMBERS : Dr. Rupesh Nasre
Department of Computer Science and Engineering

Dr. S. Umesh
Department of Electrical Engineering

CURRICULUM VITAE

NAME : Arun Baby

DATE OF BIRTH : 23 February, 1990

WEBSITE : www.arunbaby.com

EDUCATIONAL QUALIFICATIONS

2011 Bachelor of Technology

Institute : Rajagiri School of Engineering and Technology, Cochin

Specialization : Computer Science and Engineering

M.S. by Research

Institute : Indian Institute of Technology, Madras

Registration Date : 06-01-2015